Oracle® REST Data Services

Installation and Configuration Guide

Release 26.2
G58045-02
July 2026

Oracle REST Data Services Installation and Configuration Guide, Release 26.2

G58045-02

Copyright © 2011, 2026, Oracle and/or its affiliates.

Primary Authors: Mamata Basapur, Chuck Murray, Tulika Das

Contributors: Kris Rice, Jeff D. Smith, Colm Divilly, Peter J. Obrien, Dermot O'Neill, Elizabeth Saunders, Ashley Chen, Sharon Kennedy, Ganesh Pitchaiah, Jason Straub, Vladislav Uvarov

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.

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.

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Contents

Preface

Audience i
Related Documents i
Conventions i

Changes in Release 26.2 Oracle REST Data Services Installation and Configuration Guide

Changes in Oracle REST Data Services 26.2 i

1 Oracle REST Data Services Installation Checklist

1.1 System Requirements 1
1.1.1 GraalVM Configuration 2
1.2 Supported Java EE Application Servers 3
1.3 Deploying ORDS to a Kubernetes Cluster with Oracle Container Engine (OKE) 3
1.4 Supported Oracle APEX Versions 5
1.5 ORDS Installer Privileges Script 6
1.6 Privileges Granted to the Oracle REST Data Services Users 6
1.7 About the Database Users Used by Oracle REST Data Services 8
1.8 About Command-Line Interface 9
1.9 About Online Help 9

2 Installing and Configuring Oracle REST Data Services

2.1 Installing Oracle REST Data Services 1
2.1.1 Downloading ORDS 2
2.1.2 Setting Up the Configuration Folder Structure 4
2.1.2.1 Specifying the New ORDS Configuration Folder 4
2.1.3 Interactive Command-Line Interface Installation 5
2.1.3.1 Auto Installation of ORDS 6
2.1.4 Non-Interactive Command-Line Interface Installation (Silent) 29
2.1.4.1 Non-interactive (Silent) ORDS Operations Using the Bequeath Connection 30
2.1.4.2 Understanding Command Options for Command-Line Interface Installation 32

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.4.3 Using Input Redirection 34
2.1.5 Configuring ORDS to use with APEX 36
2.2 Repairing the Oracle REST Data Services Installation 36
2.2.1 Command Options for Repair CLI 37
2.3 Upgrading Oracle REST Data Services 39
2.3.1 Migrate Configuration and Upgrade from ORDS 21.4.x or Earlier Releases 39
2.3.1.1 Interactive Upgrade 40
2.3.1.2 Silent Upgrade 42
2.3.1.3 Upgrading the Multiple Database Pools 43
2.4 Uninstalling Oracle REST Data Services 43
2.4.1 Interactive Uninstall CLI 43
2.4.1.1 Entering a Number to Select the Database Pool 45
2.4.1.2 Entering a Number to Select the Database Connection Type 46
2.4.1.3 Entering the Database Host Name 46
2.4.1.4 Entering the Database Listener Port 46
2.4.1.5 Entering the Database Service Name 46
2.4.1.6 Entering the TNS Location 46
2.4.1.7 Entering a Number to Select the TNS Network Alias 46
2.4.1.8 Entering the Custom Database URL 46
2.4.1.9 Entering the Administrator Username 46
2.4.1.10 Entering the Database Password for SYS AS SYSDBA 47
2.4.1.11 Entering an Option for Uninstalling ORDS 47
2.4.2 Non-Interactive Uninstall CLI 47
2.4.2.1 Using Input Redirection 47
2.4.2.2 Command options for Uninstall CLI 48
2.5 Updating the Configuration Settings 49

3 Configuring Model Context Protocol (MCP)

3.1 About ORDS Model Context Protocol Configuration 1
3.2 ORDS MCP Configuration Settings 2
3.3 Enable the ORDS MCP Endpoint 2
3.4 Configure the Global MCP JWT Profile 2
3.5 Configure MCP Databases in Scope Mode 3
3.6 Configure MCP Databases in Role Mode 4
3.7 Verify MCP Configuration 5
3.8 Troubleshoot MCP Configuration 5

4 ORDS Configuration and Deployment Strategies

4.1 Deploying ORDS with Local Configuration 1
4.1.1 About the Request URL 1

Copyright © 2011, 2026, Oracle and/or its affiliates.

4.1.2 Configuring Additional Databases 2
4.1.3 Advanced Routing 2
4.1.4 Routing Based on the Request Path 3
4.1.5 Routing Based on the Request Hostname 4
4.2 Deploying ORDS with the OCI Database API Gateway Configuration Service 5
4.2.1 ORDS RESTful Service Definition Locations 5
4.2.2 Configuration Settings 7
4.2.3 OCI Database API Gateway Configuration Settings 7
4.2.4 OCI Database API Gateway Configuration Global Settings 8
4.2.5 Advanced ORDS Global Settings 9
4.2.6 OCI Database API Gateway Configuration Pool Settings 10
4.2.7 Advanced ORDS Pool Settings 11
4.2.8 Auto API Specifications 11
4.2.8.1 API Specifications 13
4.2.9 OCI Security 33
4.2.10 Deploying ORDS with the OCI Database API Gateway Configuration service 35
4.3 Deploying ORDS with Central Configuration Server 40
4.3.1 OpenAPI Specification 41
4.3.2 Database Mapping 41
4.3.2.1 Request Host Method 41
4.3.2.2 Header Method 42
4.3.2.3 Direct URL Specification 42
4.3.3 Global Configuration Settings 43
4.3.4 Database Pool Configuration Settings 43
4.3.4.1 Using eTag 45
4.3.4.2 Adding Credentials to an Instance Wallet 45
4.3.5 Security 45
4.3.5.1 Demonstration to Create an Entry of a Wallet for OAuth2.0 Authentication 46
4.3.6 Starting ORDS in a Central Configuration Deployment 47
4.3.6.1 Satisfying the HTTPS Requests 47
4.4 Deploying ORDS with ETCD 48
4.4.1 Prerequisites 48
4.4.2 Database mapping 49
4.4.3 Configuration Settings 49
4.4.4 Starting ORDS with ETCD 51

5 Overview of Schema Level Configuration Override

5.1 Features of Schema Level Configuration Override 1
5.2 Overridable Configuration Properties 1
5.3 Use Cases 2

Copyright © 2011, 2026, Oracle and/or its affiliates.

5.4 Scope and Visibility Matrix 3

6 Miscellaneous Configuration Options of Oracle REST Data Services

6.1 Using the REST Enabled SQL Service with MySQL Database 2
6.1.1 Understanding the Database Credentials Source Setting 2
6.1.2 Configuring a Pool for MySQL Database 2
6.1.2.1 Configuring ORDS for Supported Containers 3
6.2 Configuring Jetty in ORDS Standalone Mode 6
6.2.1 Using Java System Properties 6
6.2.2 Java System Properties 7
6.2.3 Using Jetty XML Configuration Files 8
6.3 Support for Oracle RAC Fast Connection Failover 9
6.4 Configuring ORDS with Kerberos Setup 10
6.5 Authorizing Oracle REST Data Services to Access Oracle Data Guard Protected Users 12
6.6 Configuring REST-Enabled SQL Service Settings 12
6.7 Configuring ICAP Server Integration for Virus Scan 12
6.8 Configuring the Custom Error Pages 13
6.9 Managing ORDS Administrator Privilege 14
6.9.1 Provisioning ORDS_ADMINISTRATOR_ROLE to a User 14
6.9.2 Unprovisioning ORDS_ADMINISTRATOR_ROLE from a User 14
6.10 Managing ORDS Runtime Privilege 15
6.10.1 Provisioning ORDS_RUNTIME_ROLE to a User 15
6.10.2 Unprovisioning ORDS_RUNTIME_ROLE from a User 15
6.11 Using OAuth2 in Non-HTTPS Environments 16
6.12 ORDS Performance Considerations 16
6.12.1 Configuring the Maximum Number of Rows Returned from a Query 16
6.12.2 Configuring ORDS Metadata Cache 17
6.13 Configuring ORDS to Connect with OCI Database Tools Connections 17
6.13.1 Using Instance Principal Authentication 18
6.13.2 Using OCI Profile Authentication 19

7 Deploying and Monitoring Oracle REST Data Services

7.1 Deploying Oracle REST Data Services 1
7.1.1 Serve Commands for Running in Standalone Mode 1
7.1.1.1 Non-Interactive Serve CLI 2
7.1.1.2 Serve Command Options 2
7.1.1.3 Ending an ORDS Process 3
7.1.1.4 Converting a Private Key to DER (Linux and Unix) 4
7.1.1.5 Configuring a Doc Root for Non-APEX Static Resources 5

Copyright © 2011, 2026, Oracle and/or its affiliates.

7.1.2 Oracle WebLogic Server 6
7.1.2.1 About Oracle WebLogic Server 6
7.1.2.2 Configuring Oracle APEX Images 6
7.1.2.3 Launching the Administration Server Console 8
7.1.2.4 Deploying ORDS on Oracle WebLogic Server 8
7.1.2.5 Configuring WebLogic to Handle HTTP Basic Challenges Correctly 10
7.1.2.6 Verifying the State and Health of ords and i 11
7.1.3 Apache Tomcat 11
7.1.3.1 About Apache Tomcat 12
7.1.3.2 Configuring Oracle APEX Images 12
7.1.3.3 Deploying ORDS on Apache Tomcat 12
7.1.4 Oracle Cloud Infrastructure 13
7.1.4.1 About Oracle Cloud Infrastructure 14
7.1.4.2 Configuring ORDS for High Availability on Oracle Cloud Infrastructure 14
7.1.4.3 Advantages of Deploying ORDS with High Availability on Oracle Cloud Infrastructure 14
7.2 Monitoring Oracle REST Data Services 15
7.2.1 Using the Instance API 15
7.2.1.1 Enabling the ORDS Instance API 16
7.2.1.2 Authorization for Using the ORDS Instance API 16
7.2.1.3 API Document 16
7.2.2 Using OpenTelemetry with Oracle REST Data Services 16
7.2.3 ORDS CLI Commands for Managing Pools Using JMX 18
7.2.4 Using OCI Monitoring Service with Oracle REST Data Services 19
7.2.4.1 ords-metrics Utility 20
7.2.4.2 Configuring ORDS to Send Metrics to OCI Monitoring Service 21
7.2.4.3 Enable JMX Metrics 21
7.2.4.4 Configuring OCI Alarms 22
7.2.5 Using OCI Logging Service with Oracle REST Data Services 22
7.2.5.1 Configure Java Logging in ORDS 22
7.2.5.2 Create an agent configuration 23

8 Using the Multitenant Architecture with Oracle REST Data Services

8.1 Setting Up ORDS in a CDB Environment 1
8.1.1 Installation Enabling Multiple Releases 2
8.1.1.1 Command Line Installation 3
8.1.1.2 Interactive Installation 3
8.1.1.3 Silent Installation 3
8.1.2 Upgrading Oracle REST Data Services in a CDB Environment 4
8.1.2.1 Migrating Oracle REST Data Services in the CDB to Enable Multiple Releases 4
8.1.3 Uninstalling Oracle REST Data Services in a CDB Environment 4

Copyright © 2011, 2026, Oracle and/or its affiliates.

8.2 Setting Up ORDS in an Application Container 4
8.2.1 Prerequisites for Creating ORDS in an Application Container 6
8.2.1.1 Creating an Application Root Container 6
8.2.2 Installing ORDS in the Application Root Container 7
8.2.3 Creating an Application Seed 8
8.2.4 Creating an Application PDB from the Application Seed 9
8.2.5 Configuring the PLSQL Gateway for APEX 9
8.2.6 ORDS Configuration Files Setup 10
8.2.6.1 Specifying the ORDS Configuration Folder 10
8.2.6.2 Creating the ORDS Configuration Files for the Application Root Container 10
8.2.7 Running ORDS 12
8.2.8 Upgrading ORDS in the Application Container 13
8.2.9 Uninstalling ORDS from the Application Container 14
8.2.10 Verifying ORDS in the Application Container 14
8.3 Making All PDBs Addressable by Oracle REST Data Services (Pluggable Mapping) 15

9 Installing and Configuring Customer Managed ORDS on Autonomous Database

9.1 About Customer Managed Oracle REST Data Services on Autonomous Database 1
9.2 Preinstallation Tasks 2
9.2.1 Downloading the Autonomous AI Database Wallet 3
9.2.2 Oracle APEX Static Resources 3
9.3 ORDS Command-Line Interface for Customer Managed ORDS 3
9.3.1 Interactive Install for ADB Command Line Interface 3
9.3.1.1 Customer Managed ORDS Command Options 4
9.3.1.2 Interactive Installation Prompts 5
9.3.2 Silent Installation of ADB on Command-Line Interface 14
9.3.2.1 Using Input Redirection 14

10 Enabling and Configuring the Oracle Database API for MongoDB

10.1 Getting Started 2
10.2 Requirements 5
10.3 Configurable Settings for MongoDB 5
10.4 Examples 6
10.5 Accessing the Connection Pools 7
10.6 Logging MongoDB API Access 8

Copyright © 2011, 2026, Oracle and/or its affiliates.

11 Migration of Legacy Configuration

11.1 Migrating From Legacy Configurations Using the CLI Installation 1
11.2 Migrating URL Mapping 2

A Setting-up a PL/SQL Gateway User

A.1 Configuring a PL/SQL Gateway User for APEX A-1
A.2 Configuring a PL/SQL Gateway User to Serve PL/SQL Gateway Requests Directly A-2
A.2.1 Installing/Upgrading the PL/SQL Gateway User A-2
A.2.2 Configuring the PL/SQL Gateway User A-4

B Non-Interactive ORDS CLI Commands Examples

B.1 Non-Interactive Install/Upgrade B-1
B.2 Non-Interactive Install/Upgrade (create a proxied PLSQL Gateway User) B-2
B.3 Non-Interactive Install/Upgrade (specify PLSQL Gateway User and not using RESTful Services) B-2
B.4 Non-Interactive Install/Upgrade (disable a configured PLSQL Gateway) B-2
B.5 Non-Interactive Uninstall ORDS B-2
B.6 config CLI Examples B-3
B.7 Resolving 570 Server Error Response Code B-3

C ORDS Legacy Commands and New ORDS CLI Commands

D About the Oracle REST Data Services Configuration Files

D.1 Understanding the Configuration Folder Structure D-1
D.2 Understanding the Configuration File Format D-2
D.3 Understanding the Configurable Settings D-2

E ORDS Central Configuration OpenAPI

Index

Copyright © 2011, 2026, Oracle and/or its affiliates.

List of Examples

2-1	Creating an Additional Database Pool	12
2-2	Basic Connection	30
2-3	TNS Connection	30
2-4	Custom URL Connection	31
2-5		36
4-1	db.wallet.zip	44
4-2	db.customURL	44
4-3	Response for the ORDS Global Settings	49
6-1	Using the {{--java-options}} command line option	7
6-2	Using a specific access log format	8
6-3	Always returning a certain header in the response	9
6-4	Configuring custom error page for “HTTP 404” status code	13
6-5	Using Grant command	14
6-6	Using ORDS_ADMIN package method	14
6-7	Using REVOKE command	14
6-8	Using ORDS_ADMIN package method	14
6-9	Using Grant command	15
6-10	Using ORDS_ADMIN package method	15
6-11	Using REVOKE command	15
6-12	Using ORDS_ADMIN package method	16
8-1	Creating ORDS Configuration	11
8-2	Using a Basic Connection	11

Copyright © 2011, 2026, Oracle and/or its affiliates.

List of Tables

2-1	ORDS CLI Commands	5
2-2	Prompts for Single Database Pool	7
2-3	Prompts for Multiple Database Pools	8
2-4	Prompts for Add/Update Configuration Only	9
2-5	Prompts for Install or Upgrade ORDS in Database Only	10
2-6	Creating an Additional Database Pool Prompts	13
2-7	Prompts for Selecting the Database Connection	15
2-8	Entering the Basic Connection Prompts	16
2-9	TNS Connection Prompts	16
2-10	Custom URL Connection Prompt	17
2-11	Database Pool and Standalone Configuration Prompts	19
2-12	Example of the Password Option Prompts	21
2-13	Example of the Tablespaces prompts	21
2-14	Example of Additional Features prompts	21
2-15	Prompts for standalone configuration	22
2-16	HTTPS Protocol Prompts	24
2-17	Example of Use Self-Signed Certificate	25
2-18	HTTPS Protocol and Use Self-Signed Certificate Prompts	25
2-19	HTTPS Protocol and Use my SSL Certificate Prompts	26
2-20	Prompts for generating a script to upgrade ORDS	27
2-21	Command Options for Command-Line Interface Installation	32
2-22	Command Options for Repair CLI	37
2-23	Interactive Upgrade Commands	40
2-24	Interactive Uninstall Prompts	43
2-25	Command Options for Uninstall CLI	48
4-1	OCI Database API Gateway Configuration Settings	7
4-2	OCI Database API Gateway Configuration Global Settings	8
4-3	OCI Database API Gateway Configuration Pool Settings	10
4-4	OpenAPI fields	13
4-5	ORDS Credential Wallet Storage	46
5-1	Overridable Configuration Properties	2
5-2	Use Cases of Schema Level Configuration Override feature	3
5-3	Scope and Visibility Matrix	4
6-1	Java System Properties	7
6-2	Configuration Properties for ORDS Metadata Cache	17
7-1	ORDS Metric Name	21

Copyright © 2011, 2026, Oracle and/or its affiliates.

7-2	ORDS Metrics Dimensions	21
9-1	Interactive Installation Prompts	6
9-2	HTTPS Protocol Prompts	11
9-3	Example of Use Self-Signed Certificate	12
9-4	HTTPS Protocol and Use Self-Signed Certificate Prompts	12
9-5	HTTPS Protocol and Use my SSL Certificate Prompts	13
10-1	Configuration Settings to Support the Mongo API	6
C-1	ORDS Legacy and New CLI Commands	C-1
D-1	Oracle REST Data Services Configuration Settings	D-3

Copyright © 2011, 2026, Oracle and/or its affiliates.

Preface

Oracle REST Data Services Developer's Guide explains how to develop applications using Oracle REST Data Services. (Oracle REST Data Services was called *Oracle Application Express Listener* before Release 2.0.6.)

Topics:

- Audience
- Related Documents
- Conventions

- Audience
- Related Documents
- Conventions

Audience

This document is intended for system administrators or application developers who are installing and configuring Oracle REST Data Services. This guide assumes you are familiar with web technologies, especially REST (Representational State Transfer), and have a general understanding of Windows and UNIX platforms.

Related Documents

For more information and resources relating to Oracle REST Data Services, see the following the Oracle Technology Network (OTN) site:

`http://www.oracle.com/technetwork/developer-tools/rest-data-services/`

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 is displayed on the screen, or text that you enter.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Changes in Release 26.2 Oracle REST Data Services Installation and Configuration Guide

- Changes in Oracle REST Data Services 26.2
  This section lists the changes in Oracle REST Data Services for 26.2 release.

Changes in Oracle REST Data Services 26.2

This section lists the changes in Oracle REST Data Services for 26.2 release.

New Features

- Configuring Oracle REST Data Services (ORDS) as a Model Context Protocol (MCP) server by enabling the /mcp endpoint, configuring JWT-based authentication, authorizing database access using scope or role-based permissions, exposing eligible database pools, and verifying and troubleshooting the MCP configuration. See Configuring Model Context Protocol (MCP).
- Updated System Requirements section.

Other Changes

- Updated Deploying ORDS with the OCI Database API Gateway Configuration Service section.
- Updated Understanding the Configurable Settings section.
- Updated Deploying ORDS with the OCI Database API Gateway Configuration Service.

Copyright © 2011, 2026, Oracle and/or its affiliates.

1

Oracle REST Data Services Installation Checklist

This chapter provides information about supported platforms, system requirements, and Database privileges required for using ORDS .

> **Warning**
>
> If you are planning to upgrade from version 21.4.x or earlier releases, then refer to Migration of Legacy Configuration section before proceeding. Starting with ORDS version 22.1, changes to both URL mapping and the configuration directory structure were implemented. Ensure you review these changes and then prepare your migration strategy accordingly.

- System Requirements
- Supported Java EE Application Servers
- Deploying ORDS to a Kubernetes Cluster with Oracle Container Engine (OKE)
  This section describes the procedure for deploying ORDS to an Oracle Kubernetes Cluster.
- Supported Oracle APEX Versions
- ORDS Installer Privileges Script
  This section describes about the script file that provides privileges to the user to install, upgrade, repair, and uninstall ORDS.
- Privileges Granted to the Oracle REST Data Services Users
- About the Database Users Used by Oracle REST Data Services
- About Command-Line Interface
- About Online Help

1.1 System Requirements

Oracle REST Data Services system requirements are as follows:

- A currently supported version of Oracle Database as specified in the Oracle Lifetime Support Policy
- Use one of the following:
  - Oracle Java version 17, 21, or 25
  - Oracle GraalVM Enterprise Edition for Java version 17, 21, or 25

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> On Linux system, if the Oracle Java version is lower than 17, then the installation fails with an error message. For example:
>
> ```
> Error: ORDS requires Java 17 and above to run.
>        Found Java version 11.
>        Please set JAVA_HOME to appropriate version and update PATH
> if necessary.
> ```
>
> So, you must set `JAVA_HOME` to the appropriate version and update the PATH if required.
>
> On Windows system, the user is re-directed to the Java download page.

- Web browser requirements: Refer to Oracle Software Web Browser Support Policy for more information.

> **Note**
>
> Oracle APEX is *not* a prerequisite for using Oracle REST Data Services.
>
> If Oracle APEX is installed and if RESTful services have been configured during the installation (see the step Configuring Oracle REST Data Services in Oracle APEX Installation Guide), then Oracle REST Data Services supports it.

Platforms Available

ORDS is distributed and available in the following platforms:

- You can download at Downloads
- From Oracle YUM Repository as an RPM package
- In Oracle Container Registry as a ready to run image
- GraalVM Configuration

1.1.1 GraalVM Configuration

To operate, Oracle REST Data Services requires a Java Runtime Environment. Customers can choose to use a supported Oracle GraalVM for improved performance in their runtime environment. It is not mandatory.

However, certain features in ORDS require the Java Runtime Environment to be a GraalVM with specific Graal components installed. The dependent Graal component for such features is mentioned in the documentation section. The most common dependency is on the `Graal.js` component for JavaScript support.

Use `gu list` command to determine the components available:

```
gu list
ComponentId              Version             Component name
Stability
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
------------------------------------------------------------------------------
----------
graalvm                  23.0.1             GraalVM Core
Supported
icu4j                    23.0.1             ICU4J
Supported
js                       23.0.1             Graal.js
Supported
regex                    23.0.1             TRegex
Supported
```

If the JavaScript component is not listed as an available component, then install it using the `gu install js` command. Refer to GraalVM documentation for more information on the Graal Updater command utility and the capabilities of the various Graal components.

> **Note**
>
> The ORDS image located at ORDS Docker Image Repository is based on GraalVM with the `Graal.js` component.

> **Note**
>
> The GraalVM Updater was removed in GraalVM for JDK version 21. It is not planned to embed the JavaScript component in ORDS. However, it is possible to use the plugin framework to embed Graal JS component.

> **See Also**
>
> - Embedding Graal JavaScript Component
> - Plugin Demostration Example

1.2 Supported Java EE Application Servers

Oracle REST Data Services supports the following Java EE application servers:

Application Server	Supported Release
Oracle WebLogic Server	14.1.2 Release
Apache Tomcat	9.0.x Release

1.3 Deploying ORDS to a Kubernetes Cluster with Oracle Container Engine (OKE)

This section describes the procedure for deploying ORDS to an Oracle Kubernetes Cluster.

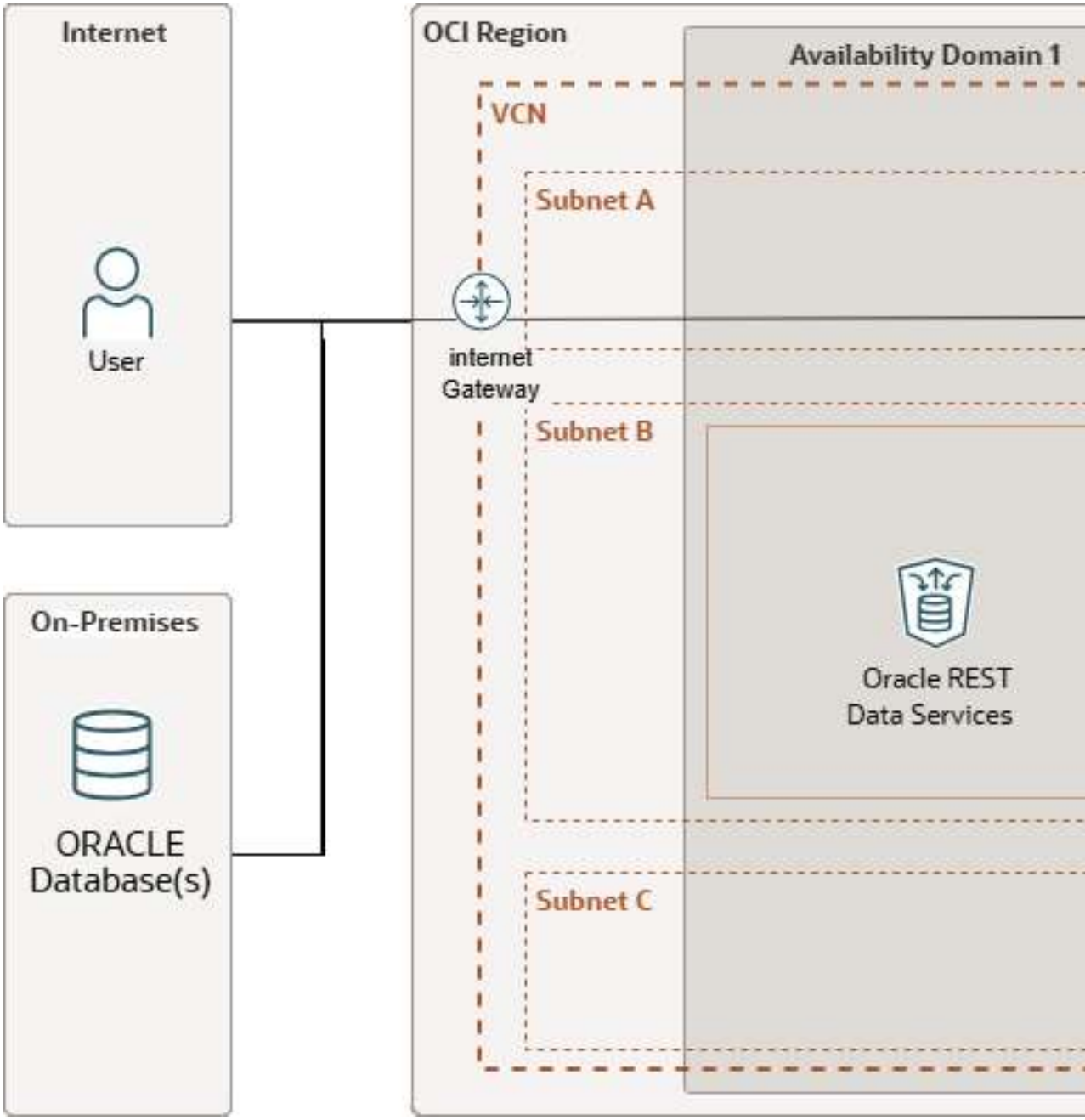
Copyright © 2011, 2026, Oracle and/or its affiliates.

The steps below detail the process of deploying Oracle REST Data Services (ORDS) within an Oracle Kubernetes Cluster. Leveraging Oracle Kubernetes Engine (OKE) enables efficient management of deployments across multiple nodes, ensuring both scalability and reliability:

1. **Accessing the Cluster through kubectl**: Ensure that you have access to your OKE cluster through the `kubectl` command-line interface before moving forward. This enables you to directly interact with the cluster and its components.
2. **Setting Up Configuration Source**: You need to configure a configuration source for ORDS to persist configurations. You can choose from options such as OCI file storage, etcd, or a central configuration server to provide and manage configuration for the ORDS instances.
3. **Configuring ORDS**: Deploy ORDS within the cluster through the creation of an appropriate deployment file. Customize settings such as database connectivity, security protocols, and other configurations to meet your requirements. For example, using a central configuration server.
4. **Managing APEX Images**: For applications developed with APEX, ensure proper management by uploading images to the designated location and configuring permissions as needed. Oracle also recommends using the Oracle APEX Static Resources Content Delivery Network (CDN).
5. **Creating New Pool Configurations:** You can use multiple databases, you can create the pool configuration using ORDS install command.
6. **Migrating or Managing Existing Configurations:** If you are upgrading ORDS versions or modifying the existing configurations, adhere to best practices to prevent disruptions and maintain compatibility.
7. **Deploying ORDS with HTTPS Support:** Establish secure communication by enabling end-to-end HTTPS. This involves acquiring SSL/TLS certificates and configuring them appropriately.
8. **Enabling Autoscaling Features:** Utilize Horizontal Pod Autoscalers (HPA) and Cluster Autoscaler (CA) to dynamically adjust the number of replicas based on demand. HPA monitors CPU usage, while CA manages node scaling automatically.
9. **Managing Logging and Monitoring Services**: Implement robust logging mechanisms to monitor system activities and troubleshoot issues efficiently. Utilize OCI monitoring tools to obtain insights into resource usage and overall system health.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Figure 1-1 Architecture diagram for deploying ORDS on Oracle Kubernetes Cluster



1.4 Supported Oracle APEX Versions

Oracle REST Data Services supports the currently supported versions of APEX.

> **See Also**
>
> SUPPORTED VERSIONS OF APEX

Copyright © 2011, 2026, Oracle and/or its affiliates.

1.5 ORDS Installer Privileges Script

This section describes about the script file that provides privileges to the user to install, upgrade, repair, and uninstall ORDS.

> **Note**
>
> This script is used when you do not want to use SYS AS SYSDBA to install, upgrade, repair, and uninstall ORDS for Oracle PDB or Oracle 11g.

The Oracle REST Data Services contains a script, `ords_installer_privileges.sql` which is located in the `scripts/installer` folder. The script provides the assigned database user the privileges to install, upgrade, repair, and uninstall ORDS in Oracle Database.

Perform the following steps:

1. Using SQLcl or SQL*Plus, connect to the Oracle Database instance. You must have a Database account with appropriate privileges for installing ORDS.
2. Execute the following script providing the database user:

```
SQL> @/path/to/scripts/installer/ords_installer_privileges.sql exampleuser
SQL> exit
```

You must use the specified database user to install, upgrade, repair, and uninstall ORDS.

1.6 Privileges Granted to the Oracle REST Data Services Users

As part of the Oracle REST Data Services installation, privileges are granted to several users and roles:

- `ORDS_RUNTIME_ROLE` role
  - `ORDS_RUNTIME_ROLE` is granted `EXECUTE` on the following packages if these packages are not granted `EXECUTE` to `PUBLIC`:
    - `SYS.DBMS_LOB`
    - `SYS.DBMS_SESSION`
    - `SYS.DBMS_UTILITY`
    - `SYS.WPIUTL`
  - `ORDS_RUNTIME_ROLE` is granted the necessary `ORDS_METADATA` object privileges to determine the repository version and to access the connection pool configurations.
- `ORDS_PUBLIC_USER` user
  - `ORDS_PUBLIC_USER` is granted connect to allow connection to the database.
  - `ORDS_PUBLIC_USER` is granted role, `ORDS_RUNTIME_ROLE` to allow the user to act as an ORDS runtime user
- `ORDS_ADMINISTRATOR_ROLE` role
  - `ORDS_ADMINISTRATOR_ROLE` is granted EXECUTE on `ORDS_METADATA.ORDS_ADMIN` PL/SQL package.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- PUBLIC
  - PUBLIC is granted EXECUTE on ORDS_METADATA.ORDS_REPVERSION view to allow the repository version to be queried by anyone.
  - PUBLIC is granted SELECT on many ORDS_METADATA views.
  - PUBLIC is granted EXECUTE on ORDS_METADATA PL/SQL packages that are available for developer users.
- ORDS_METADATA schema
  - ORDS_METADATA schema is granted on the following packages if these packages are not granted EXECUTE on PUBLIC:
    - SYS.DBMS_ASSERT
    - SYS.DBMS_LOB
    - SYS.DBMS_OUTPUT
    - SYS.DBMS_SCHEDULER
    - SYS.DBMS_SESSION
    - SYS.DBMS_UTILITY
    - SYS.DEFAULT_JOB_CLASS
    - SYS.HTP
    - SYS.OWA
    - SYS.WPG_DOCLOAD
  - ORDS_METADATA is granted SELECT (11g) or READ (12*c* or later) on the following view if it is not granted SELECT or READ to PUBLIC:
    - SYS.SESSION_PRIVS
  - ORDS_METADATA schema is granted EXECUTE on the following packages:
    - SYS.DBMS_CRYPTO
    - SYS.DBMS_METADATA
  - ORDS_METADATA schema is granted SELECT (11g) or READ (12c or later) on the following views:
    - SYS.DBA_OBJECTS
    - SYS.DBA_ROLE_PRIVS
    - SYS.DBA_TAB_COLUMNS
  - ORDS_METADATA schema is granted SELECT including WITH GRANT OPTION on the following views:
    - SYS.USER_CONS_COLUMNS
    - SYS.USER_CONSTRAINTS
    - SYS.USER_OBJECTS
    - SYS.USER_PROCEDURES
    - SYS.USER_TAB_COLUMNS
    - SYS.USER_TABLES
    - SYS.USER_VIEWS

Copyright © 2011, 2026, Oracle and/or its affiliates.

  - ORDS_METADATA schema is granted the following system privileges:
    - ALTER USER
    - CREATE ANY TRIGGER
    - CREATE JOB
    - CREATE VIEW
    - CREATE PUBLIC SYNONYM
    - DROP PUBLIC SYNONYM
  - ORDS_METADATA schema is granted the necessary object privileges to migrate APEX REST data to ORDS_METADATA tables.
  - ORDS_METADATA schema is granted ORDS_ADMINISTRATOR_ROLE, ORDS_RUNTIME_ROLE roles with administrator option.
- PUBLIC is granted SELECT on many ORDS_METADATA tables and views.
- PUBLIC is granted EXECUTE on PL/SQL packages that are available for users to invoke.
- ORDS_METADATA is granted EXECUTE on the following packages if these packages are not granted EXECUTE to PUBLIC:
  - SYS.DBMS_ASSERT
  - SYS.DBMS_LOB
  - SYS.DBMS_OUTPUT
  - SYS.DBMS_SCHEDULER
  - SYS.DBMS_SESSION
  - SYS.DBMS_UTILITY
  - SYS.DEFAULT_JOB_CLASS
  - SYS.HTP
  - SYS.OWA
  - SYS.WPG_DOCLOAD
- ORDS_METADATA is granted the necessary object privileges to migrate APEX REST data to ORDS_METADATA tables.

1.7 About the Database Users Used by Oracle REST Data Services

Oracle REST Data Services uses the following database users:

User Name	Required	Description
ORDS_METADATA	Yes	Owner of the PL/SQL packages used for implementing many Oracle REST Data Services capabilities. ORDS_METADATA is where the metadata about Oracle REST Data Services-enabled schemas is stored. It is not accessed directly by Oracle REST Data Services; the Oracle REST Data Services application never creates a connection to the ORDS_METADATA schema.

Copyright © 2011, 2026, Oracle and/or its affiliates.

User Name	Required	Description
ORDS_PUBLIC_USER	Yes	User for Oracle REST Data Services to make an initial database connection for handling requests. Depending on the request, the user will proxy to the relevant Oracle REST Data Services-enabled schema or PL/SQL Gateway user to complete the requested action in the database.

1.8 About Command-Line Interface

ORDS release 22.1.0 introduces the ORDS Command-Line Interface that provides the following benefits:

- Create, update, and list your ORDS configuration
- Add additional database pools to your configuration Install/upgrade
- Repair, or uninstall ORDS in the database
- Run ORDS in standalone mode

The preceding functions can be accomplished interactively through prompts, or run silently (non-interactively) using the ORDS commands.

1.9 About Online Help

The online help provides information about the commands along with the available options and arguments. To show the list of ORDS commands, execute the following command:

```
ords --help
```

To show the details of an ORDS command, specify the ORDS command followed by `--help`. If you want additional details for a sub-command, then specify the `--help` after the sub-command.

Syntax:

```
ords <command> --help
```

```
ords <command> <sub-command> --help
```

Examples:

- `ords install --help`

Copyright © 2011, 2026, Oracle and/or its affiliates.

- `ords config --help`

```
ords config set --help
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

2

Installing and Configuring Oracle REST Data Services

This chapter describes how to install, configure, and upgrade Oracle REST Data Services (ORDS).

- Installing Oracle REST Data Services
- Repairing the Oracle REST Data Services Installation
- Upgrading Oracle REST Data Services
  This section describes how to upgrade to Oracle REST Data Services latest release.
- Uninstalling Oracle REST Data Services
  This section describes the interactive and non-interactive uninstallation CLI commands used to uninstall the ORDS schema.
- Updating the Configuration Settings
  The ORDS `config` command is used to update and edit the contents of the global and database pool specific settings in the configuration files.

> **See Also**
>
> - Deploying ORDS with High Availability on Oracle Cloud Infrastructure
> - Using the Multitenant Architecture with Oracle REST Data Services
> - About the Database Users Used by Oracle REST Data Services
> - Configuring a PL/SQL Gateway User to Serve PL/SQL Gateway Requests Directly

2.1 Installing Oracle REST Data Services

You must complete the following steps before installing ORDS.

> **Note**
>
> The procedures in this topic apply to installing Oracle REST Data Services in a non-CDB Database or a PDB. For setting up ORDS in a CDB environment, refer to Using the Multitenant Architecture with Oracle REST Data Services section.
>
> If you are planning to use the Oracle API for MongoDB, then you must install Oracle REST Data Services in a non-CDB or PDB. See Enabling and Configuring the Oracle Database API for MongoDB for more information.

- Downloading ORDS
- Setting Up the Configuration Folder Structure

Copyright © 2011, 2026, Oracle and/or its affiliates.

You can then install ORDS using one of the following options:

- Interactive Command-Line Interface Installation
- Non-Interactive Command-Line Interface Installation (Silent)

- Downloading ORDS
  This section describes how to download ORDS.
- Setting Up the Configuration Folder Structure
  This section describes how to set up the configuration folder structure.
- Interactive Command-Line Interface Installation
- Non-Interactive Command-Line Interface Installation (Silent)
- Configuring ORDS to use with APEX
  APEX requires PL/SQL Gateway access to ORDS.

> **See Also**
>
> - Oracle REST Data Services Downloads
> - Oracle REST Data Services SODA for REST Developer's Guide

2.1.1 Downloading ORDS

This section describes how to download ORDS.

Downloading ORDS Zip file

Download the `ords_<latest>.zip` file from the Oracle REST Data Services (ORDS) download page, and then unzip `ords_<latest>.zip` file into a folder of your choice. The folder you choose to unzip the file is referred to as the ORDS product folder. The ORDS product folder contains a `bin` folder and other folders and files required to run ORDS.

Adding ORDS `bin` folder to your Operating System PATH
Oracle recommends to add the ORDS `bin` folder to your operating system PATH environment variable.

Example for LINUX/UNIX Operating System

```
echo -e 'export PATH="$PATH:/<ords product folder>/bin"' >> ~/.bash_profile
```

Start a new shell to pick up this change.

Example for WINDOWS Operating System
Use a command prompt with administrator privileges.

```
SetX PATH "%PATH%;<ords product folder>\bin"
```

Start a new command prompt to pick up this change.

Example for MAC Operating System

```
echo -e 'export PATH="$PATH:/<ords product folder>/bin"' >> ~/.zprofile
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Start a new terminal to pick up this change.

Downloading ORDS Using yum

ORDS is published in Oracle Linux (OL) 8 and 9 repositories. System administrators can add this repository to a yum configuration in order to install ORDS and its dependencies.

- OL8 Repo
- OL9 Repo

If the oracle software repository has been added to your yum configuration, then you can download and install ORDS using the following command:

```
sudo yum install ords
```

The preceding command handles all the ORDS dependencies and place the ORDS program in `/usr/local/bin/ords` folder to set up your ORDS configuration and install or upgrade ORDS in the database.

Oracle recommends you to use `/etc/ords/conf` configuration directory. The following comand is used to configure ORDS:

```
ords --config /etc/ords/config install
```

If you want to use a different directory to place your ORDS configurations and you want to use the Linux system service management as system d or system v, then configure the ORDS `config` directory in `/etc/ords.conf` folder.

Installing ORDS Using RPM

If your system where you want to install is not using yum or is not connected to the internet, then you can download the ORDS RPM from the links mentioned in the preceding section (Download ORDS using yum) and install the RPM manually. This does not handle the ORDS dependencies, you must install the java JDK manually. After downloading the RPM file you can install it using the following command:

```
sudo rpm -i ords-22.1.X-X.elX.noarch.rpm
```

This places the ORDS program in `/usr/local/bin/ords` folder to set up your ORDS configuration and install or upgrade ORDS in the database.

To configure your ORDS, we recommend using the configuration directory in `/etc/ords/conf` folder with the following command:

```
ords --config /etc/ords/config install
```

If you want to use a different directory to place your ORDS configurations and you want to use the LINUX system service management as systemd or system v, then configure the ORDS configuration directory in `/etc/ords.conf` folder.

Using ORDS image from Oracle Container Registry

Oracle REST Data Services is available as a Docker image (which also supports Podman) in the Oracle Container Registry. This ORDS image includes GraalVM and an ORDS production release configured for standalone deployment.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Oracle REST Data Services Docker image

2.1.2 Setting Up the Configuration Folder Structure

This section describes how to set up the configuration folder structure.

- Specifying the New ORDS Configuration Folder

2.1.2.1 Specifying the New ORDS Configuration Folder

New installation has the ORDS configuration files placed in the configuration folder. Configuration files from ORDS release 21.4.x or earlier are migrated to the configuration folder.

You can obtain the configuration folder location using one of the following options:

1. **Command option**:
   Use the `--config` option to specify your configuration folder.

   Example:

   ```
   ords --config /path/to/conf install
   ```

2. **Environment variable**:

   > **Note**
   >
   > This option is preferred if you forget to include the `--config` option when you are using the ORDS command-line Interface.

   - `ORDS_CONFIG`: Create the `ORDS_CONFIG` environment variable.
     Example for LINUX or UNIX operating system:

     ```
     Shell script containing ORDS_CONFIG
     cat example_env
     export ORDS_CONFIG=/path/to/conf

     echo $ORDS_CONFIG
     /path/to/conf
     ```

3. **Current Working Directory**:
   If the `--config <configuration_folder>` option is not specified, and the `ORDS_CONFIG` environment variable is not defined, then your current working directory is used as the configuration folder.

   Example:

   If your current working directory is `/path/to/conf`, then it uses that location for your configuration folder.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Note

- Oracle highly recommends you to ensure that ORDS does not create the configuration directory in the ORDS product folder. For example, you can have a configuration folder in the following location:

  ```
  /Users/<user_name>/work/dbtools-dev/config/
  ```

  The best practice is to have your configuration files separate from the application files, this makes maintenance and upgrades easier and more reliable.

- The following command is no longer valid:

  Example: `java -jar ords.war install`

  If you specify the legacy commands using `java -jar ords.war`, you get the following warning message:

  Warning: Support for executing: `java -jar ords.war` has been deprecated. Please add ords to your PATH and use the ords command instead. Run the following command to add ORDS to your `PATH`:

  <Displays an example of adding the bin folder to your PATH>

  Start a new terminal to pick up this change. Oracle recommends to add the ORDS product `bin` folder to your path.

2.1.3 Interactive Command-Line Interface Installation

This section explains various scenarios that use the auto-installation feature found in the interactive Command-Line Interface (CLI) installation prompts.

To be prompted for the scenarios discussed in the following sections, you must execute one of the following commands:

Table 2-1 ORDS CLI Commands

ORDS CLI Command	Description
`ords install`	Creates a database pool or upgrades ORDS in existing database pool(s). Generates an upgrade script if more than one database pool exists in the ORDS configuration.
`ords install -i --db-only`	Install or upgrade ORDS that are in the database pools, or specify the database connection. Generates upgrade script if more than one database pool exists in the ORDS configuration. With this command, ORDS configuration is neither created nor updated.
`ords install -i --config-only`	Creates or updates a database pool. This command does not install or upgrade ORDS in the database.

Configuration Folder

Copyright © 2011, 2026, Oracle and/or its affiliates.

Specify the install command without specifying the `--config <CONFIG FOLDER>` option. This assumes that the configuration directory is specified through the environment variable, or you are defaulting to the current working directory.

Example:

- `ords install`
- `ords install -i --config-only`
- `ords install -i --db-only`

Specify the install command using the `--config <CONFIG FOLDER>` option. Depending on the ORDS CLI commanda used, ORDS uses that configuration folder to create or update the configuration or read the existing configuration from that `<CONFIG FOLDER>` location.

Example:

- `ords --config /path/to/myconfig install`
- `ords --config /path/to/myconfig install -i --config-only`
- `ords --config /path/to/myconfig install -i --db-only`
- Auto Installation of ORDS
  Starting with Oracle REST Data Services (ORDS) release 24.1, auto installation feature is introduced.

2.1.3.1 Auto Installation of ORDS

Starting with Oracle REST Data Services (ORDS) release 24.1, auto installation feature is introduced.

Auto-Install operation, discovers the database connections and display them to the user. Creating a database pool is much simpler and default settings are used to minimize the prompting to the user. If a bequeath connection is available, then it is used to retrieve the information from the database, and install or upgrade ORDS in the database without using the credentials of the administrator user.

> **See Also**
>
> Setup Bequeath Connection for Interactive Install

Following are some of the scenarios demonstrating auto-installation feature:

- Scenario 1: New ORDS Configuration and ORDS Installation
- Scenario 2: Single Database Pool (Default Pool)
- Scenario 3: Multiple Database Pools
- Scenario 4: Add or Update Configuration Only
- Scenario 5: Install or Upgrade ORDS in Database Only
- Scenario 1: New ORDS Configuration and ORDS Installation
  This section explains the interactive command-line interface (CLI) installation prompts for creating a new ORDS configuration.
- Scenario 2: Single Database Pool (Default Pool)
- Scenario 3: Multiple Database Pools

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Scenario 4: Add or Update Configuration Only
- Scenario 5: Install or Upgrade ORDS in Database Only
- Creating an Additional Database Pool
- Multiple Options for Database Pool and Standalone Configuration
- Setup Bequeath Connection for Interactive Install

2.1.3.1.1 Scenario 1: New ORDS Configuration and ORDS Installation

This section explains the interactive command-line interface (CLI) installation prompts for creating a new ORDS configuration.

Run the following ORDS CLI command:

```
ords install
```

When you run the preceding command, the interactive inataller detects that an ORDS configuration does not exist. A default database pool is created. Refer to Creating an Additional Database Pool, you are prompted to create the default database pool and to install ORDS in the database.

2.1.3.1.2 Scenario 2: Single Database Pool (Default Pool)

Run the following ORDS CLI command:

```
ords install
```

When you create a new ORDS configuration, a default pool is created. This scenario indicates that there is only one database pool, known as the `default` database pool. You have the option to select the default database pool and upgrade ORDS in the database. You also have the option to create an additional database pool.

Table 2-2 Prompts for Single Database Pool

Prompt	Description

`Enter a number to select the database pool to upgrade ORDS or create an additional database pool`
`[1] default jdbc:oracle:thin:@DB1?TNS_ADMIN=/path/to/myfolder`
`[C] Create an additional database pool`
	Refer to
	• Entering a Number to Select the Database Pool
	• Creating an Additional Database Pool

2.1.3.1.3 Scenario 3: Multiple Database Pools

In this scenario, if more than one database pools exist in your configuration, you have the option to upgrade ORDS in an existing database pool, generate an upgrade script, specify the database pool to upgrade ORDS or create an additional database pool.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Displays a list of the database pools in your configuration. The default database pool is displayed first followed by the other database pools in an alphabetical order.

Run the following ORDS CLI command:

```
ords install
```

Table 2-3 Prompts for Multiple Database Pools

Prompt	Description

`Enter a number to select the database pool to upgrade ORDS. Total database pools is 12.`
`Showing only the first 9 database pools. Additional options to Generate script, Specify a database pool,`
`or Create additional database pool.`
`[1] default jdbc:oracle:thin:@MYPDB?TNS_ADMIN=/path/to/my/folder`
`[2] exampledb1 jdbc:oracle:thin:@DB1?TNS_ADMIN=/path/to/my/folder`
`[3] exampledb2 jdbc:oracle:thin:@DB2?TNS_ADMIN=/path/to/my/folder`
`[4] exampledb3 jdbc:oracle:thin:@DB3?TNS_ADMIN=/path/to/my/folder`
`[5] fxdb50 jdbc:oracle:thin:@//localhost:1521/fxdb50`
`[6] hrdb105 jdbc:oracle:thin:@//myhr.xyzcorp.com:1521/hrdb`
`[7] sampledb7 jdbc:oracle:thin:@//my-dev.sample.com:1523/db7`
`[8] sampledb8 jdbc:oracle:thin:@//my-dev.sample.com:1523/db8`
`[9] sampledb9 jdbc:oracle:thin:@//my-dev.sample.com:1523/db9`
`[G] Generate script to upgrade ORDS in all the database pools`
`[S] Specify the database pool name to upgrade ORDS`
`[C] Create an additional database pool`
`Choose [1]:`
	If the total number of database pools is greater than 9, then the option `Specify the database pool name to upgrade ORDS` is displayed.
	Refer to:
	• Entering a Number to Select the Database Pool
	• Generate an Upgrade Script
	• Specify the Database Pool Name to Upgrade ORDS
	• Creating an Additional Database Pool

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.3.1.4 Scenario 4: Add or Update Configuration Only

Run the following ORDS CLI command:

```
ords install -i --config-only
```

Using this scenario, you can update an existing database pool or create a database pool. The `--config-only` option does not perform the install or upgrade ORDS in the database operation.

Table 2-4 Prompts for Add/Update Configuration Only

Prompt	Description

`Enter a number to select the database pool to update, or create an additional database pool.`
`[1] default jdbc:oracle:thin:@MYPDB?TNS_ADMIN=/path/to/my/folder`
`[2] exampledb1 jdbc:oracle:thin:@DB1?TNS_ADMIN=/path/to/my/folder`
`[3] sampledb7 jdbc:oracle:thin:@//my-dev.sample.com:1523/db7`
`[C] Create an additional database pool`
`Choose [1]:`
	Refer to:
	• Multiple Options for Database Pool and Standalone Configuration
	• Creating an Additional Database Pool

2.1.3.1.5 Scenario 5: Install or Upgrade ORDS in Database Only

Run the following ORDS CLI command:

```
ords install -i --db-only
```

Using this scenario, you can only install or upgrade ORDS in the database. If more than one database pool exists in your configuration, you have the option to:

- upgrade ORDS in an existing database pool
- generate an upgrade script
- specify the database connection or database pool to install or upgrade ORDS in the database

Displays a list of the database pools in your configuration.

The default database pool is displayed first, followed by the other database pools in alphabetical order. A maximum of 9 database pools are displayed. If the maximum number of database pools is reached, the option "`Specify the database pool name to upgrade ORDS`" is displayed.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-5 Prompts for Install or Upgrade ORDS in Database Only

Prompt	Description

`Enter a number to select the database pool to upgrade ORDS. Total database pools is 12.`
`Showing only the first 9 database pools. Additional options to Generate script,`
`Specify a database pool or database connection to upgrade ORDS.`
`[1] default jdbc:oracle:thin:@MYPDB?TNS_ADMIN=/path/to/my/folder`
`[2] exampledb1 jdbc:oracle:thin:@DB1?TNS_ADMIN=/path/to/my/folder`
`[3] exampledb2 jdbc:oracle:thin:@DB2?TNS_ADMIN=/path/to/my/folder`
`[4] exampledb3 jdbc:oracle:thin:@DB3?TNS_ADMIN=/path/to/my/folder`
`[5] fxdb50 jdbc:oracle:thin:@//localhost:1521/fxdb50`
`[6] hrdb105 jdbc:oracle:thin:@//myhr.xyzcorp.com:1521/hrdb`
`[7] sampledb7 jdbc:oracle:thin:@//my-dev.sample.com:1523/db7`
`[8] sampledb8 jdbc:oracle:thin:@//my-dev.sample.com:1523/db8`
`[9] sampledb9 jdbc:oracle:thin:@//my-dev.sample.com:1523/db9`
`[G] Generate script to upgrade ORDS in all the database pools`
`[S] Specify the database connection to install/upgrade ORDS`
`[P] Specify the database pool name to upgrade ORDS`
`Choose [1]:`
	If the total number of database pools is greater than 9, then the option `Specify the database pool name to upgrade ORDS` is displayed. Refer to:
	• Entering a Number to Select the Database Pool
	• Generate an Upgrade Script
	• Entering a Number to Select the Database Connection
	• Specify the Database Pool Name to Upgrade ORDS

- Entering a Number to Select the Database Pool

2.1.3.1.5.1 Entering a Number to Select the Database Pool

The database pool(s) and its connections are displayed. Enter the number to select the database pool that you want to install or upgrade ORDS in the database. The installer installs

Copyright © 2011, 2026, Oracle and/or its affiliates.

ORDS in the database if ORDS does not exist in the database. The installer upgrades ORDS in the database if the ORDS schema version is less than the ORDS product version that you are using.

2.1.3.1.6 Creating an Additional Database Pool

Create an additional database pool option

The Create an additional database pool option, prompts you for the following information:

- Database pool name
  - If a database pool already exists in your ORDS configuration, then you are prompted for the database pool name.

> **Note**
>
> If you are creating a new ORDS configuration, then you are not prompted for the database pool name since it creates the default database pool.

- Connection information
  - Shows the TNS Net service names from the `tnsnames.ora` file. You have the option to select the TNS Net service name.
    - Automatically locates the `tnsnames.ora` file in your folders
  - Specify the database connection (`Basic`, `TNS` or `Custom URL`).
- Administrator username and password

> **Note**
>
> If you are using Bequeath Connection and the host and servicename or sid exists in the database, then you are not prompted for the administrator username and password.

- Multiple options to update the database pool and standalone configuration
  - Connection information
  - Runtime user `ORDS_PUBLIC_USER` password
  - Default and temporary tablespaces for the ORDS runtime user and the ORDS schema
  - Additional ORDS Feature
  - Standalone Mode Configuration
    - HTTP/HTTPS Protocol
    - Protocol port
    - Certificate Type for HTTPS Protocol
      - Self-Signed Certificate
      - User provides the Certificate
    - APEX Static Resources Location

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> If APEX exists in the database, then the `APEX static resources location` option is displayed.

The multiple options for the database pool and standalone configuration are displayed and enables you to make any updates. If you are satisfied with the values that are displayed, then select option `A` to accept and continue. If the option contains Accept and Continue - Create configuration and Install ORDS in the database, then the database pool is created and ORDS is installed in the database. If you enable the option `Configure and start ORDS in Standalone Mode`, then the standalone settings are created or updated, and ORDS runs after the installation completes.

Example 2-1 Creating an Additional Database Pool

```
Enter the database pool name: database2

Enter a number to select the TNS Net Service Name to use or specify the
database connection
    [1] TNS name: DB2  SERVICE_NAME=pdb2
    [2] TNS name: DB3  SERVICE_NAME=pdb3
    [3] TNS name: DB4  SERVICE_NAME=pdb4
    [S] Specify the database connection
Choose [1]: 1

Connecting to administrator user: SYS AS SYSDBA for PDB service name: pdb2
using bequeath connection

ORDS is not installed in the database. ORDS installation is required.

Enter a number to update the value or select option A to Accept and Continue
  [1] Connection Type: TNS
  [2] TNS Connection: TNS_NAME=DB2 TNS_FOLDER=/path/to/myfolder
         Administrator User: SYS AS SYSDBA via Bequeath Connect
  [3] Database password for ORDS runtime user (ORDS_PUBLIC_USER): <generate>
  [4] ORDS runtime user and schema tablespaces:  Default: SYSAUX Temporary
TEMP
  [5] Additional Feature: Database Actions
  [6] Configure and start ORDS in Standalone Mode: Yes
  [7]    Protocol: HTTPS
  [8]       HTTPS Port: 8443
  [9]    Certificate Type: Use Self-Signed Certificate
  [10]      SSL Hostname: myhost
  [11]   APEX static resources location: /path/to/apex/images
  [A] Accept and Continue - Create configuration and Install ORDS in the
database
  [Q] Quit - Do not proceed. No changes
Choose [A]:
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-6 Creating an Additional Database Pool Prompts

Prompt	Description
`Enter the database pool name: database2`	Prompts for the database pool name if your ORDS configuration already exists and contains a database pool. **Note** If you are creating a new configuration, you are not prompted for the database pool name because it creates the database pool named `default`. Refer to Entering the Database Pool Name

`Enter a number to select the TNS Net Service Name to use or specify the database connection`
`[1] TNS name: DB2 SERVICE_NAME=pdb2`
 	
`[2] TNS name: DB3 SERVICE_NAME=pdb3`
 	
`[3] TNS name: DB4 SERVICE_NAME=pdb4`
 	
`[S] Specify the database connection`
`Choose [1]:`
	Refer to
	• Entering a Number to Select the TNS Net Service Name to Use or Specify the Database Connection
	• Entering a Number to Select the Database Connection

`Provide database user name with administrator privileges.`
`Enter the administrator username: SYS`
`Enter the database password for SYS AS SYSDBA:`
	Note
	If you are using Bequeath Connection and the host and servicename and sid exists in the database, then you are not prompted for the administrator username or password.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-6 (Cont.) Creating an Additional Database Pool Prompts

Prompt	Description

`Enter a number to update the value or select option A to Accept and Continue`
`[1] Connection Type: TNS`
`[2] TNS Connection: TNS_NAME=DB2 TNS_FOLDER=/path/to/myfolder`
`Administrator User: SYS AS SYSDBA via Bequeath Connect`
`[3] Database password for ORDS runtime user (ORDS_PUBLIC_USER): <generate>`
`[4] ORDS runtime user and schema tablespaces: Default: SYSAUX Temporary TEMP`
`[5] Additional Feature: Database Actions`
`[6] Configure and start ORDS in Standalone Mode: Yes`
`[7] Protocol: HTTPS`
`[8] HTTPS Port: 8443`
`[9] Certificate Type: Use Self-Signed Certificate`
`[10] SSL Hostname: myhost`
`[11] APEX static resources location: /path/to/apex/images`
`[A] Accept and Continue - Create configuration and Install ORDS in the database`
`[Q] Quit - Do not proceed. No changes`
`Choose [A]:`
	Displays the default values highlighted in blue for the database pool.
	Displays the Standalone settings that already exists in the ORDS configuration. These settings are displayed after the option `Configure and start ORDS in Standalone Mode`.
	Refer to Multiple Options for Database Pool and Standalone Configuration

- Entering the Database Pool Name
- Entering a Number to Select the TNS Net Service Name to Use or Specify the Database Connection
- Entering a Number to Select the Database Connection

2.1.3.1.6.1 Entering the Database Pool Name

The database pool name displays when you choose the option `Create an Additional Database Pool`. Specify the database pool name carefully since it sets the mapping pattern.

If you select the option `c` when prompted, then the database pool name prompt is displayed. Enter the database pool name. For example, if the pool name you enter is `database1`, then all the REST requests to that pool starts with `/ords/database1/`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> - The database pool name must only contain lowercase alphabets a-z, digits 0-9, and the characters “-“, “_“ and “.“ .
> - When the database pool is created, the database pool folder is also created along with the pool specific settings, and a wallet to store the database user credentials.
> - The database pool name also sets the mapping pattern.
>
> If you are creating a new ORDS configuration, then you are not prompted for the database pool name.

2.1.3.1.6.2 Entering a Number to Select the TNS Net Service Name to Use or Specify the Database Connection

A list of the TNS net service name and its service name or SID displays if the `tnsnames.ora` file is found in your folder. If the `tnsnames.ora` does not exist in your folder, then the database connection type options are shown.

> **See Also**
>
> Entering a Number to Select the Database Connection

2.1.3.1.6.3 Entering a Number to Select the Database Connection

Select the database connection type.

Table 2-7 Prompts for Selecting the Database Connection

Prompt	Description

`Enter a number to select the database connection type to use`
`  [1] Basic (host name, port, service name)`
`  [2] TNS (TNS alias, TNS directory)`
`  [3] Custom database URL`
`Choose [1]:`
	Refer to:
	• Entering the Basic Connection
	• Entering the TNS Connection
	• Entering the Custom URL Connection

- Entering the Basic Connection
- Entering the TNS Connection
- Entering the Custom URL Connection

2.1.3.1.6.3.1 Entering the Basic Connection

Specify the database host name, database listener port and the service name.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-8 Entering the Basic Connection Prompts

Prompt	Description
`Enter the database host name [localhost]:`	Refer to Entering the Database Host Name
`Enter the database listen port [1521]:`	Refer to Entering the Database Listener Port
`Enter the database service name [orcl]:`	Refer to Entering the Database Service Name

- Entering the Database Host Name
- Entering the Database Listener Port
  Specify the database port for a basic connection.
- Entering the Database Service Name

2.1.3.1.6.3.1.1 Entering the Database Host Name
Specify the database host name for the basic connection.

2.1.3.1.6.3.1.2 Entering the Database Listener Port
Specify the database port for a basic connection.

2.1.3.1.6.3.1.3 Entering the Database Service Name
Specify the service name. The default value for the service name is obtained from one of the following options in the given sequence:

- The service name setting in the ORDS configuration file. If it is not present, then
- It checks for `ORACLE_PDB` environment variable. If it is undefined, then
- It checks for `ORACLE_SID` environment variable. If it is undefined, then
- It uses `orcl` as the default value.

2.1.3.1.6.3.2 Entering the TNS Connection
Specify the TNS folder location and select the TNS network alias.

Table 2-9 TNS Connection Prompts

Prompt	Description
`Enter the TNS location: /path/to/ myfolder`	Refer to Entering the TNS Location

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-9 (Cont.) TNS Connection Prompts

Prompt	Description

`Enter a number to select the TNS Network alias to use`
`[1] DB1`
`SERVICE_NAME=pdb1`
 	
`[2] DB2`
`SERVICE_NAME=pdb2`
 	
`[3] FREEPDB1`
`SERVICE_NAME=freepdb1`
 	
`Choose [1]:`
	Refer to Entering a Number to Select the TNS Network Alias

- Entering the TNS Location
- Entering a Number to Select the TNS Network Alias

2.1.3.1.6.3.2.1 Entering the TNS Location

Specify a valid folder of where the `tnsnames.ora` file is located.

2.1.3.1.6.3.2.2 Entering a Number to Select the TNS Network Alias

Select the TNS alias name from the list of TNS network alias names displayed.

2.1.3.1.6.3.3 Entering the Custom URL Connection

Specify the Custom URL connection.

Table 2-10 Custom URL Connection Prompt

Prompt	Description

`Enter the Custom database URL:`
`jdbc:oracle:thin:@//`
`localhost:1521/orcl`
	Refer to Entering the Custom Database URL

- Entering the Custom Database URL

2.1.3.1.6.3.3.1 Entering the Custom Database URL

Specify the custom database URL for custom database URL connection.

Example URL formats:

```
jdbc:oracle:<driver>:@//<host>:<port>/<servicename>
jdbc:oracle:thin:@//localhost:1521/orcl

jdbc:oracle:<drivertype>:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<host>)
(PORT=<port>))(CONNECT_DATA=(SERVICE_NAME=<servicename>)))
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=orcl)))
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.3.1.7 Multiple Options for Database Pool and Standalone Configuration

If you are creating or updating a database pool, the multiple options display the database pool and standalone configuration prompts. The standalone configuration options display after the `Configure ORDS and start in Standalone Mode` prompt.

> **Note**
>
> If a database pool already exists in your ORDS configuration, then the existing standalone settings are displayed.

Enter the option number to change the value of the setting. When you are satisfied with the values, press the return key to accept and continue.

- Multiple Options with Default Values
- Standalone Configuration
  This section explains how to configure Oracle REST Data Services (ORDS) and run in a standalone mode.

2.1.3.1.7.1 Multiple Options with Default Values

For a new database pool, the following default values are highlighted in blue:

- Generate the Runtime user (`ORDS_PUBLIC_USER`) password
- If you are installing ORDS in the database, the default values for the default tablespace is `SYSAUX` and the temporary tablespace is `TEMP`.

  > **Note**
  >
  > If any of these tablespaces do not exist, then it uses the database default tablespaces.

- The additional feature defaults to `Database Actions`.
- If you are installing ORDS in standalone mode, then `Configure and Start ORDS in Standalone Mode` defaults to `Yes`
- Protocol defaults to `HTTP`
- The default value for Choose is `[A] Accept and Continue`

Copyright © 2011, 2026, Oracle and/or its affiliates.

Figure 2-1 Example of Multiple Options with Default Values

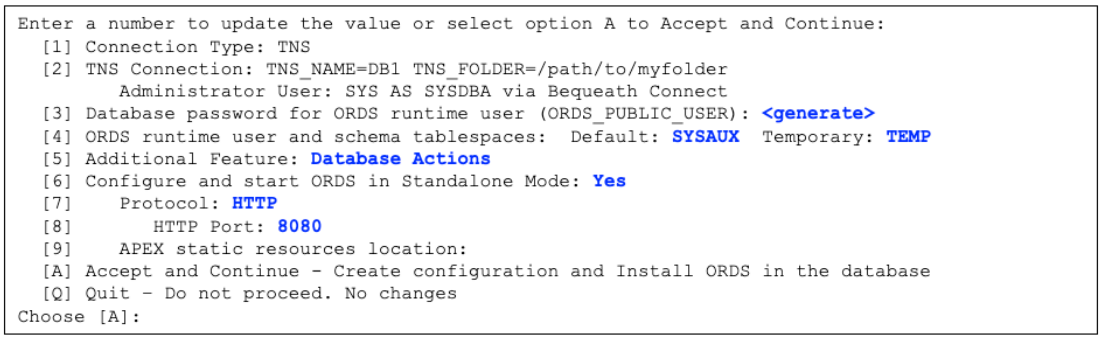


Table 2-11 Database Pool and Standalone Configuration Prompts

Prompt	Description
`Connection Type: TNS`	The type of connection. Displays either Basic, TNS or Custom URL. Refer to Entering a Number to Select the Database Connection
`Basic Connection: HOST=localhost PORT=1521 SERVICE_NAME=DB1`	Displays Basic Connection if connection type is Basic. Refer to Entering the Basic Connection
`TNS Connection: TNS_NAME=DB1 TNS_FOLDER=/path/to/myfolder`	Displays TNS Connection if connection type is TNS. Refer to Entering the TNS Connection
`Custom URL Connection: CUSTOM_URL=jdbc:oracle:thin:@// localhost:1521/DB1`	Displays Custom URL Connection if connection type is Custom URL. Refer to Entering the Custom URL Connection
`Database password for ORDS runtime user (ORDS_PUBLIC_USER): <generate>`	Defaults to generate password. Refer to Entering the Password Option
`ORDS runtime user and schema tablespaces: Default: SYSAUX Temporary TEMP`	Displays Tablespace option only if installing ORDS in the database. Refer to Entering the Tablespaces
`Additional Feature: Database Actions`	Defaults Additional Feature to Database Actions. Refer to Entering a Number to Select the Additional Feature

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-11 (Cont.) Database Pool and Standalone Configuration Prompts

Prompt	Description
`Configure and start ORDS in Standalone Mode: Yes`	Toggles to `Yes` or `No`. Displays option if installing ORDS in the database and not using the Bequeath Connection. Refer to Standalone Configuration
`Configure ORDS in Standalone Mode: Yes`	Toggles to Yes or No Displays option if using the Bequeath Connection or the configuration only (`--config-only`) option. Refer to Standalone Configuration
`Protocol: HTTP`	Defaults Protocol to HTTP. Refer to Entering a Number to Use HTTP or HTTPS Protocol
`HTTP Port: 8080`	Defaults HTTP Port to 8080. Displays HTTP Port if protocol is HTTP. Refer to Entering a Number to Use HTTP or HTTPS Protocol
`APEX static resources location: /path/to/apex/images`	Refer to Entering the APEX Static Resources Location
`Accept and Continue - Create configuration and Install ORDS in the database`	Refer to Accept and Continue Option

- Accept and Continue Option
- Entering the Password Option
- Entering the Tablespaces
- Entering a Number to Select the Additional Feature

2.1.3.1.7.1.1 Accept and Continue Option

Once you are done updating and reviewing the values, select option `[A] Accept and Continue`. Accept and Continue includes a message on what is performed. For example, Accept and Continue - `Create configuration and Install ORDS in the database` option creates the database pool and ORDS is installed in the database. If configure ORDS in standalone mode is enabled, then the standalone configuration is also created or updated.

2.1.3.1.7.1.2 Entering the Password Option

The runtime user password for `ORDS_PUBLIC_USER` defaults to generate password. If you want to change the password and specify your own password, select option `S` to specify the password. You are prompted twice for the password.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-12 Example of the Password Option Prompts

Prompt
Enter your choice for the runtime user password (ORDS_PUBLIC_USER) [S] Specify your own password [G] Generate password [C] Cancel - No Changes Choose [G]: S Enter the database password for ORDS_PUBLIC_USER: Confirm password:

2.1.3.1.7.1.3 Entering the Tablespaces

The tablespace option only displays if installing ORDS in the database. Specify the default and temporary tablespaces for the ORDS database runtime user and the ORDS schema. The default value for the default tablespace is `SYSAUX` and the default value for the temporary tablespace is `TEMP`.

Table 2-13 Example of the Tablespaces prompts

Prompts
Enter the default tablespace for ORDS_METADATA and ORDS_PUBLIC_USER [SYSAUX]: MYTABLESPACE
Enter the temporary tablespace for ORDS_METADATA and ORDS_PUBLIC_USER [TEMP]:

2.1.3.1.7.1.4 Entering a Number to Select the Additional Feature

Specify the additional feature that you want to enable.

Table 2-14 Example of Additional Features prompts

Prompt
Enter a number to select additional feature(s) to enable: [1] Database Actions (Enables all features) [2] REST Enabled SQL and Database API [3] REST Enabled SQL [4] Database API [5] None Choose [1]:

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.3.1.7.2 Standalone Configuration

This section explains how to configure Oracle REST Data Services (ORDS) and run in a standalone mode.

Table 2-15 Prompts for standalone configuration

Prompt	Description
`Configure and start ORDS in Standalone Mode:`	Toggle the option to `Yes` or `No`. A `Yes` indicates to configure ORDS in a standalone mode. In addition, ORDS runs in a standalone mode after completing the ORDS install or upgrade process. A `No` indicates, do not make any standalone configuration changes and do not run ORDS in a standalone mode. **Note** The `Configure and start ORDS in Standalone Mode` option is displayed only if you are installing ORDS and you are not using the bequeath connection.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-15 (Cont.) Prompts for standalone configuration

Prompt	Description
`Configure ORDS in Standalone Mode`	You can toggle the option to `Yes` or `No`. A `Yes` indicates to configure ORDS in a standalone mode. A `No` indicates not to make any standalone configuration changes.

Note

- The Configure ORDS in Standalone Mode option displays only if you are using the bequeath connection or the configuration only (`--config-only`) option.
- If you have setup your OS environment to use the bequeath connection, then ensure that the `JDK_JAVA_OPTIONS` does not contain the `useOracleHome=true` prior to using the `ords serve` command. Remove `useOracleHome` from the environment variable `JDK_JAVA_OPTIONS` or unset `JDK_JAVA_OPTIONS`.

If ORDS is installed in the database, and you want to run ORDS in a standalone mode, then use the `ords serve` command.

- Entering a Number to Use HTTP or HTTPS Protocol
- Entering the HTTPS Port
- Entering a Number to Select the Certificate Type

Copyright © 2011, 2026, Oracle and/or its affiliates.

- HTTPS Protocol and Use Self-Signed Certificate
- HTTPS Protocol and Use my SSL Certificate
- Entering the APEX Static Resources Location
- Generate an Upgrade Script
- Specify the Database Pool Name to Upgrade ORDS

2.1.3.1.7.2.1 Entering a Number to Use HTTP or HTTPS Protocol

Displays the HTTP/HTTPS Protocol prompts. Select the protocol (HTTP or HTTPS) that you want to use.

> **Note**
>
> Oracle highly recommends to run ORDS in a standalone mode with HTTPS versus HTTP.

- Entering the HTTP Port

2.1.3.1.7.2.1.1 Entering the HTTP Port

Specify the HTTP port. The default HTTP port is 8080.

2.1.3.1.7.2.2 Entering the HTTPS Port

Specify the HTTPS port. The default HTTPS port is 8443.

Table 2-16 HTTPS Protocol Prompts

Prompt	Description

`Enter a number to select the protocol`
`[1] HTTP`
`[2] HTTPS`
`Choose [1]:`
	Select the protocol (HTTP or HTTPS) that you want to use.

`Enter the HTTPS port [8443]:`	Specify the HTTPS port. The default HTTPS port is 8443.

`Enter a number to select the certificate type`
`[1] Use self-signed certificate (generates automatically)`
`[2] Use my SSL certificate (requires SSL certificate and SSL certificate private key)`
`Choose [1]: 1`
	Refer to:
	• Entering a Number to Select the Certificate Type
	• HTTPS Protocol and Use Self-Signed Certificate
	• HTTPS Protocol and Use my SSL Certificate

2.1.3.1.7.2.3 Entering a Number to Select the Certificate Type

You have the option to select the self-signed certificate that generates automatically when running ORDS in standalone mode, or you provide the SSL certificate which requires both the SSL Certificate and its private key.

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.3.1.7.2.4 HTTPS Protocol and Use Self-Signed Certificate

Selecting the prompt `Use self-signed certificate` prompts you for the SSL hostname.

Table 2-17 Example of Use Self-Signed Certificate

Prompt

`Enter a number to select the certificate type`
`[1] Use self-signed certificate (generates automatically)`
`[2] Use my SSL certificate (requires SSL certificate and SSL certificate private key)`
`Choose [1]: 1`
 	
`Enter the SSL hostname: myhost`

ORDS generates the Self-Signed Certificate when running ORDS in a standalone mode.

The value entered here is stored in the `standalone.https.host` configuration setting. The Self-Signed Certificate is generated only once and the Common Name value for that certificate is taken from the `standalone.https.host` configuration setting. Changing the `standalone.https.host` after the Self-Signed Certificate is generated does not change the Common Name in that certificate.

Table 2-18 HTTPS Protocol and Use Self-Signed Certificate Prompts

Prompt	Description
`Certificate Type: Use Self-Signed Certificate`	
`SSL Hostname: myhost`	Specify the SSL Hostname.

2.1.3.1.7.2.5 HTTPS Protocol and Use my SSL Certificate

Selecting Use my SSL Certificate requires you to provide an existing SSL certificate and its private key.

Example of Use my SSL Certificate

```
Enter a number to select the certificate type
   [1] Use self-signed certificate (generates automatically)
   [2] Use my SSL certificate (requires SSL certificate and SSL certificate private key)
Choose [1]: 2

 Enter the path for the SSL Certificate: /path/to/mycertificate.crt
 Enter the path for the SSL Certificates private key: /path/to/mycert.der
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-19 HTTPS Protocol and Use my SSL Certificate Prompts

Prompt	Description
`Certificate Type: Use my SSL Certificate`	Refer to Entering a Number to Select the Certificate Type
`Path for the SSL Certificate: /path/to/mycert.crt`	Refer to Entering the Path for the SSL Certificate
`Path for the SSL Certificate private key: /path/to/mycert.der`	Refer to Entering the Path for the SSL Certificate's Private Key

- Entering the Path for the SSL Certificate
- Entering the Path for the SSL Certificate's Private Key

2.1.3.1.7.2.5.1 Entering the Path for the SSL Certificate

Specifying the SSL certificate file location and filename.

Example:

```
/path/to/mycert.crt
```

2.1.3.1.7.2.5.2 Entering the Path for the SSL Certificate's Private Key

Specifying the SSL certificate's private key file location and filename.

Example

```
/path/to/mycert.der
```

2.1.3.1.7.2.6 Entering the APEX Static Resources Location

ORDS detects if APEX is installed in the database and if the database user `APEX_PUBLIC_USER` exists. If found, then it displays the APEX static resources location option. Provide the folder location of the APEX images.

2.1.3.1.7.2.7 Generate an Upgrade Script

Selecting the option `[G] Generate script to upgrade ORDS in all the database pools`, prompts you for the script location and generates the upgrade script. The upgrade script contains ORDS non-interactive (silent) command to upgrade ORDS in the database and includes the database pool to use. You can run this script in your shell environment.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-20 Prompts for generating a script to upgrade ORDS

Prompts
Enter a number to select the database pool to upgrade ORDS. Total database pools is 12. Showing only the first 9 database pools. Additional options to Generate script, Specify a database pool, or Create additional database pool. [1] default jdbc:oracle:thin:@MYPDB?TNS_ADMIN=/path/to/my/folder [2] exampledb1 jdbc:oracle:thin:@DB1?TNS_ADMIN=/path/to/my/folder [3] exampledb2 jdbc:oracle:thin:@DB2?TNS_ADMIN=/path/to/my/folder [4] exampledb3 jdbc:oracle:thin:@DB3?TNS_ADMIN=/path/to/my/folder [5] fxdb50 jdbc:oracle:thin:@//localhost:1521/fxdb50 [6] hrdb105 jdbc:oracle:thin:@//myhr.xyzcorp.com:1521/hrdb [7] sampledb7 jdbc:oracle:thin:@//my-dev.sample.com:1523/db7 [8] sampledb8 jdbc:oracle:thin:@//my-dev.sample.com:1523/db8 [9] sampledb9 jdbc:oracle:thin:@//my-dev.sample.com:1523/db9 [G] Generate script to upgrade ORDS in all the database pools [S] Specify the database pool name to upgrade ORDS [C] Create an additional database pool Choose [1]: G Enter the location to store the ORDS upgrade script [/path/to/scripts]: Generating upgrade script... Completed generating upgrade script at /path/to/scripts/ords_upgrade_db_pools_2024-03-24_161653_46492

If you are using the bequeath connection and ORDS exist in the database based on your database pool connection information such as host, port, servicename/sid, then the non-interactive (silent) command includes the `--bequeath-connect` option.

Example:

```
ords --config /path/to/myconfig install --db-pool exampledb1 --bequeath-
connect --db-only
```

Otherwise, you must replace the `--admin-user <username>` with the administrator user, and the `--password-stdin < <file>` with a file that contains the administrator password.

Example:

```
ords --config /path/to/myconfig install --db-pool <pool> --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool <pool> --admin-user ADMIN --
db-only --password-stdin < /path/to/special.txt
```

Example of the Generated Upgrade Script

```
# -------------------------------------------------------------
# Date       : 24 Mar 2024 16:27:05
# Release    : Oracle REST Data Services 24.1.0.r0822130
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
# Type       : Generated Upgrade Script
# ------------------------------------------------------------

# ------------------------------------------------------------
# Database   : Oracle Database 23c Free
# DB Version : 23.3.0.23.09
# Install/upgrade command using --bequeath-connect option
# ------------------------------------------------------------

ords --config /path/to/myconfig install --bequeath-connect --db-only
ords --config /path/to/myconfig install --db-pool exampledb1 --bequeath-
connect --db-only
ords --config /path/to/myconfig install --db-pool exampledb2 --bequeath-
connect --db-only
ords --config /path/to/myconfig install --db-pool exampledb3 --bequeath-
connect --db-only
ords --config /path/to/myconfig install --db-pool fxdb50 --bequeath-connect --
db-only

# ------------------------------------------------------------
# Install/upgrade command using --db-only option
# ------------------------------------------------------------

ords --config /path/to/myconfig install --db-pool hrdb105 --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool sampledb7 --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool sampledb8 --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool sampledb9 --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool testdb111 --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool testdb222 --admin-user
<username> --db-only --password-stdin < <file>
ords --config /path/to/myconfig install --db-pool testdb333 --admin-user
<username> --db-only --password-stdin < <file>
```

2.1.3.1.7.2.8 Specify the Database Pool Name to Upgrade ORDS

The option `Specify the database pool name to upgrade ORDS` prompts you for the database pool name. If the database pool name exists in your ORDS configuration, then ORDS is upgraded in the database.

Example:

```
Enter the database pool name: db1
```

2.1.3.1.8 Setup Bequeath Connection for Interactive Install

You can use the bequeath connection to install and upgrade Oracle REST Data Services in the database.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> For bequeath connections, if you access the database using a fully qualified domain name (FQDN), then the target database must have a domain that matches the DNS domain of the host machine. Ensure that the instance parameter `db_domain` is set to the same domain. A mismatch may prevent ORDS from establishing a bequeath connection.

The bequeath connection enables ORDS to connect directly to an Oracle database bypassing the network listener. You can then execute ORDS on the same server as the database.

Administrator username and password are not required for using the bequeath connection. To use the bequeath connection, set the following environment variables:

- `ORACLE_HOME`
- `ORACLE_SID`
- `LD_LIBRARY_PATH (points to the $ORACLE_HOME/lib)`

You can then execute one of the following ORDS CLI commands to install ORDS:

> **Note**
>
> `--java-options "-DuseOracleHome=true"` option is mandatory for using the bequeath connection to install/upgrade, repair or uninstall ORDS.

- `ords --java-options "-DuseOracleHome=true" install -i --bequeath-connect`
  Creates a database pool or upgrades ORDS that are in the database pool(s).
- `ords --java-options "-DuseOracleHome=true" install -i --bequeath-connect --db-only`
  Install or upgrade ORDS that are in the database pools, or specify the database connection. Here, the configuration is not created or updated.

2.1.4 Non-Interactive Command-Line Interface Installation (Silent)

The non-interactive command-line interface (CLI) installation or silent installation, performs the following tasks:

- Creates or updates the ORDS configuration files
- Installs or upgrades the ORDS schema
- Creates the ORDS proxy user and the related objects in the Database

The non-interactive installation command does not run ORDS in standalone mode. If you want to run ORDS in standalone mode, then it must be followed by the `ords serve` command.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> If you have setup your OS environment to use the bequeath connection, then ensure that the `JDK_JAVA_OPTIONS` does not contain the `useOracleHome=true` prior to using the `ords serve` command. Remove `useOracleHome` from the environment variable `JDK_JAVA_OPTIONS` or unset `JDK_JAVA_OPTIONS`.

If you provide the command-line options for the non-interactive installation and if you are missing a required option (for example: Basic connection requires the `db-port` option), then an error message is displayed.

The `--db-pool <pool_name>` option specifies the database pool to create or update. If the `--db-pool` option is omitted, then it defaults to the `default` database pool.

- Non-interactive (Silent) ORDS Operations Using the Bequeath Connection
- Understanding Command Options for Command-Line Interface Installation
- Using Input Redirection

2.1.4.1 Non-interactive (Silent) ORDS Operations Using the Bequeath Connection

You can use the bequeath connection to install or upgrade, repair and uninstall Oracle REST Data Services in the database.

The bequeath connection enables ORDS to connect directly to an Oracle database bypassing the network listener. You can then execute ORDS on the same server as the database.

Using the bequeath connection, administrator username, and password are not required. Set the following environment variables to use the bequeath connection:

- `ORACLE_HOME`
- `ORACLE_SID`
- `LD_LIBRARY_PATH (points to the $ORACLE_HOME/lib)`

> **Note**
>
> `--java-options "-DuseOracleHome=true"` option is mandatory for using the bequeath connection to install/upgrade, repair or uninstall ORDS.

Examples using ORDS install command using the `--bequeath` option:

Example 2-2 Basic Connection

```
ords --java-options "-DuseOracleHome=true" --config <PATH TO CONFIG> install
--db-pool <POOL NAME> --
bequeath-connect --db-hostname <DB HOST> --db-port <DB PORT> --
db-servicename <DB SERVICENAME> --proxy-user --feature-sdw true
--log-folder <LOG FOLDER> --password-stdin < <PATH TO FILE>
```

Example 2-3 TNS Connection

```
ords --java-options "-DuseOracleHome=true" --config <PATH TO CONFIG> install
--db-pool <POOL NAME> --
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
bequeath-connect --db-tns-alias <TNS ALIAS> --db-tns-dir <TNS
FOLDER> --proxy-user --feature-sdw true --log-folder <LOG
FOLDER> --password-stdin < <PATH TO FILE>
```

Example 2-4 Custom URL Connection

```
ords --java-options "-DuseOracleHome=true" --config <PATH TO CONFIG> install
--db-pool <POOL NAME> --
bequeath-connect --db-custom-url <Custom URL> --proxy-user --
feature-sdw true --log-folder <LOG FOLDER> --password-stdin <
<PATH TO FILE>
```

> **Note**
>
> If you omit `--db-pool` option, then the settings are written to the `default` pool settings file.

Example of the ords install `--db-only` command to upgrade an existing database pool.

```
ords --java-options "-DuseOracleHome=true" --config <PATH TO CONFIG> install
--db-pool <POOL NAME> --bequeath-connect --db-only
```

> **Note**
>
> If you omit `--db-pool` option, the upgrade occurs using the `default` pool.

Example of the ords install repair command using an existing database pool:

```
ords --java-options "-DuseOracleHome=true" --config <PATH TO CONFIG> install
repair --db-pool <POOL NAME> --bequeath-connect
```

> **Note**
>
> If you omit `--db-pool` option, the repair occurs using the `default` pool.

Example using ords uninstall command using an existing database pool:

```
ords --java-options "-DuseOracleHome=true" --config <PATH TO CONFIG>
uninstall --db-pool <POOL NAME> --bequeath-connect
```

> **Note**
>
> If you omit `--db-pool` option, the uninstall occurs using the `default` pool.

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.4.2 Understanding Command Options for Command-Line Interface Installation

Table 2-21 Command Options for Command-Line Interface Installation

Options	Description
`--admin-user <USERNAME>`	Specifies the user with privileges to install or upgrade ORDS in the database, or obtains the ORDS status information.
`--bequeath-connect`	Specifies that bequeath connect that enables clients to connect directly to an Oracle database bypassing the network listener. This is when the client is on the same server as the database. Refer to Oracle REST Data Services documentation to setup your environment to use the --bequeath-connect option.
`--config <CONFIG_PATH>`	Specifies the path to the folder containing the configuration files.
`--config-only`	Specifies to create or update the configuration only.
`--db-only`	Specifies to install or upgrade ORDS in the database and not to create or update the configuration files.
`--db-pool <POOL_NAME>`	Specifies the name of the database pool to create, update or use. If the `db-pool` option is omitted, then the `default` database pool is used.
`--db-custom-url <URL>`	Specifies the custom database url for the database connection.
`--db-hostname <HOST>`	Specifies the database host name.
`--db-port <PORT>`	Specifies the database port.
`--db-servicename <SERVICENAME>`	Specifies the database service name.
`--db-sid <SID>`	Specifies the database SID.
`--db-tns-alias <ALIAS_NAME>`	Specifies the TNS alias name that exists in the `tnsnames.ora` file.
`--db-tns-dir <TNS_DIR>`	Specifies the folder where the `tnsnames.ora` file is located.
`--feature-db-api <BOOLEAN>`	Specifies if you want to enable DB API feature. Possible values are: `true` or `false`. • If the value is set to `true`, then DB API feature is enabled. • If the value is set to `false`, then DB API feature is disabled. Returns an error if the specified options are `--feature-sdw true` and `--feature db-api false`.
`--feature-rest-enabled-sql <BOOLEAN>`	Specifies if you want to enable REST-Enabled SQL feature. Possible values are: `true` or `false`. • If the value is set to `true`, then the REST-Enabled SQL feature is enabled. • If the value is set to `false`, then the REST-Enabled SQL feature is disabled. Returns an error if the specified options are `--feature-sdw true` and `--feature-rest-enabled-sql false`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-21 (Cont.) Command Options for Command-Line Interface Installation

Options	Description
`--feature-sdw <BOOLEAN>`	Specifies if you want to enable Database Actions feature. Possible values are: `true` or `false`. • If the value is set to `true`, then the Database Actions feature is enabled. • If the value is set to `false`, then the Database Actions feature is disabled. If the option is set to `true`, then the following settings are set to `true` in the configuration file: • `database.api.enabled` • `restEnabledSql.active` Returns an error if `--feature-sdw true` and any of following options are specified, and are set to `false`: • `--feature-db-api` • `--feature-rest-enabled-sql`
`--gateway-mode <MODE>`	Specifies the PL/SQL gateway mode. Possible values are: `proxied`, `direct`, or `disabled`. Default value is `disabled`. Where: • `disabled`- PL/SQL gateway is not allowed. • `direct` - The PL/SQL gateway call is performed directly as the user identified by `db.username` (located in the `pool.xml` setting). • `proxied` - The PL/SQL gateway call is performed by proxying from the run-time user (`db.username` is located in the `pool.xml`) to the relevant schema configured by `ords_admin.config_plsql_gateway`.
`--gateway-user <USER>`	Specifies the user that exists in the Oracle database and has privileges to access the stored procedures.
`--interactive, -i`	Prompts user for all the required information.
`--help, -h`	Shows how to use the command.
`--legacy-config <FOLDER>`	Specify the legacy configuration folder to migrate to the revised configuration structure.
`--legacy-context <STRING>`	Specifies the context (example: `ords`, `apex`, and so on.) that was used in the legacy configuration.
`--log-folder`	Writes the logs from the install, upgrade, repair or uninstall to the log folder. If this option is omitted, then the output is written to standard output.
`--password-stdin`	To run the ORDS install, upgrade, repair, or uninstall command non-interactively, use this option to read the password value from standard input when redirecting input to a file or here document.
`--pdb-exclude <(PDB...)>`	Specifies the option applicable for CDB. List of PDBs that are excluded from the ORDS install/ upgrade, or repair.
`--pdb-open-readwrite-all`	Specifies the option applicable for CDB. Specifies whether to open all PDBs in read write mode if their status is either closed or read only.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-21 (Cont.) Command Options for Command-Line Interface Installation

Options	Description
`--pdb-open-readwrite <(PDB...)>`	Specifies the option applicable for CDB. List of PDBs that are read-only or closed to be open for read/write to install, upgrade, repair, or uninstall ORDS.
`--pdb-skip-readonly`	Specifies the option applicable for CDB. Skips PDBs that are read-only. If this option is omitted, an error message is returned informing the user that the PDB(s) are read-only for install or upgrade ORDS. Excludes `PDB$SEED` since it is open for read/write.
`--pdb-skip-closed`	Specifies the option applicable for CDB. Skips PDBs that are closed (mounted). If this option is omitted, an error message is returned informing the user that the PDB(s) are closed for installation/upgradation of ORDS.
`--proxy-user`	Specifies the password of the ORDS database user (`ORDS_PUBLIC_USER`) that is in a file or in a `here` document when redirecting the standard input.
`--proxy-user-tablespace <TABLESPACE>`	Specifies the option applicable for ORDS install. The default tablespace for proxy user (For example: `ORDS_PUBLIC_USER`). If this option is omitted, then the defaulted database default tablespace is used.
`--proxy-user-temp-tablespace <TABLESPACE>`	Specifies the option applicable for an ORDS install. The temporary tablespace for proxy user (For example: `ORDS_PUBLIC_USER`). If this option is omitted, then the defaulted database temporary tablespace is used.
`--schema-tablespace <TABLESPACE>`	Specifies the default tablespace for `ORDS_METADATA` schema. If this option is omitted, then the defaulted database default tablespace is used.
`--schema-temp-tablespace <TABLESPACE>`	Specifies the option applicable for an ORDS install. The temporary tablespace for `ORDS_METADATA` schema. If this option is omitted, then the defaulted database temporary tablespace is used.

2.1.4.3 Using Input Redirection

This section describes how to redirect the standard input using the here command or to a file.

Redirect STDIN to a file

Redirect STDIN to a file that contains the password. In the following example, the file must contain two passwords. Each password must be on a separate line.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example:

```
$ cat password.txt
firstpassword
secondpassword
```

The first password belongs to the first user on the command line (for example: `SYS`). The second password belongs to the second user on the command line (for example: `proxy-user`).

```
ords --config <configuration_folder> install --db-pool <pool_name> --admin-
user <username> --proxy-user
--db-hostname <host> --db-port <port_number> --db-servicename <service_name>
--log-folder <log_folder> --feature-sdw <boolean> --password-stdin <
<filename>
```

Example:

```
ords --config /path/to/conf install --db-pool db1 --admin-user SYS --proxy-
user --db-hostname localhost
--db-port 1521 --db-servicename orcl --log-folder /path/to/logs --feature-sdw
true
--password-stdin < password.txt
```

Redirect Standard Input Using Here Document

Redirect STDIN using the Here document (also known as heredoc) for the password(s). The heredoc consists of the '<<' redirection operator followed by a delimiter token.

Each password must be on a separate line and it is ended by the delimiter token.

Example:

The first password belongs to the first user on the command line (for example: `SYS`). The second password belongs to the second user on the command line (for example: `proxy-user`).

```
ords --config <configuration_folder> install --db-pool <pool_name> --admin-
user <username> --proxy-user --db-hostname
 <host> --db-port <port_number> --db-servicename <service_name> --log-folder
<log_folder> --feature-sdw <boolean>
--password-stdin << EOF
<password1>
<password2>
EOF
```

```
ords --config /path/to/conf install --db-pool db1 --admin-user SYS --proxy-
user --db-hostname
localhost --db-port 1521 --db-servicename orcl --log-folder /path/to/logs --
feature-sdw true
--password-stdin << EOF
<password1>
<password2>
EOF
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

2.1.5 Configuring ORDS to use with APEX

APEX requires PL/SQL Gateway access to ORDS.

To configure ORDS to use with APEX, once APEX installation is completed, run the following command:

> **Note**
>
> If you have a single database pool (for example: pool named default), then omit the `--db-pool` option.

```
ords --config <config path> config --db-pool <pool name> set
plsql.gateway.mode proxied
```

Example 2-5

```
ords --config /path/to/config config set plsql.gateway.mode proxied
```

```
ords --config /path/to/config config --db-pool db1 set plsql.gateway.mode
proxied
```

Restart ORDS, you can then access APEX from the ORDS landing page:

```
http://<host>:<port>/ords
```

2.2 Repairing the Oracle REST Data Services Installation

If you want to check that the Oracle REST Data Services installation is valid, specify the repair command.

> **Note**
>
> To repair ORDS in the database, the ORDS product version must be the same version as the ORDS installed schema version in the database.

Example of repair command to prompt for information:

```
ords [--config <folder>] install repair --interactive [--log-folder <folder>]
```

If you have an existing configuration, you may choose to select from a list of database pools. You also have the option to specify the database connection for ORDS to be installed. You are prompted to provide the administrator username and password.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example

```
Oracle REST Data Services - Interactive Repair

  Enter a number to select the database pool to use or specify the database
connection
    [1] default      jdbc:oracle:thin:@//localhost:1521/orcl
    [2] sales          jdbc:oracle:thin:@//localhost:1521/pdb1
    [S] Specify the database connection
  Choose [1]:
  Provide database user name with administrator privileges.
    Enter the administrator username: SYS
    Enter the database password for SYS AS SYSDBA:
```

> **Note**
>
> If you install Oracle REST Data Services before APEX run the following command to ensure that the ORDS configuration for your database pool contains the PL/SQL gateway mode setting and is set to `proxied`.
>
> If you have a single database pool (for example: pool named `default`), then omit the `--db-pool` option.
>
> ```
> ords --config <config_path> --db-pool <pool_name> get
> plsql.gateway.mode
> ```
>
> Run the following command to set `plsql.gateway.mode` value to `proxied`:
>
> ```
> ords --config <config_path> config --db-pool <pool_name> set
> plsql.gateway.mode proxied
> ```

- Command Options for Repair CLI
  This section describes the interactive and non-interactive install repair CLI commands used to repair the ORDS schema in the database.

2.2.1 Command Options for Repair CLI

This section describes the interactive and non-interactive install repair CLI commands used to repair the ORDS schema in the database.

Table 2-22 Command Options for Repair CLI

Command	Description
`--admin-user <string>`	Specifies the user with privileges to install, upgrade, or uninstall ORDS in the database.
`--bequeath-connect`	Specifies that the bequeath connect enables clients to connect directly to an Oracle database bypassing the network listener. This occurs if client is on the same server as the database.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-22 (Cont.) Command Options for Repair CLI

Command	Description
`--db-custom-url <url>`	Specifies the custom database url for the database connection.
`--db-hostname <string>`	Specifies the database host name.
`--db-pool <string>`	Specifies the name of the database connection pool.
`--db-port <int>`	Specifies the database port.
`--db-servicename <string>`	Specifies the database service name.
`--db-sid <string>`	Specifies the system identifier which is a unique name that identifies the database.
`--db-tns-alias <string>`	Specifies the TNS alias name that exist in the `tnsnames.ora` file.
`--db-tns-dir <folder>`	Specifies the folder where the `tnsnames.ora` file is located.
`-h, --help`	Shows how to use and provides information of the command.
`-i, --interactive`	Prompts for the required information.
`--log-folder <folder>`	Writes the logs from the install, upgrade or uninstall to the log folder. If this option is omitted, then the output is written to standard output.
`--password-stdin`	Specifies to use this option to read the password value from standard input when redirecting input to a file or `here` document when you run the ords install or uninstall command non-interactively, If the `--password-stdin` option and the redirection symbol ("<" or "<<") are omitted, then you are prompted for the password(s) for the command option(s) that require the password.
`--pdb-exclude <string...>`	Specifies the list of PDBs that are excluded from the ORDS install, upgrade, or repair. This option only applies to the CDB.
`--pdb-open-readwrite <string...>`	Specifies the list of PDBs that are read only or closed to be open for read write for ORDS install, upgrade, repair or uninstall. This option only applies to the CDB.
`--pdb-open-readwrite-all`	Specifies to open all PDBs in read write mode if their status is either closed or read only. This option only applies to the CDB.
`--pdb-skip-closed`	Skips PDBs that are closed (mounted). This option only applies to the CDB.
`--pdb-skip-readonly`	Skips PDBs that are read-only. This option only applies to the CDB.

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.3 Upgrading Oracle REST Data Services

This section describes how to upgrade to Oracle REST Data Services latest release.

Upgrading from ORDS 21.4.x or earlier releases

If you are planning to upgrade from version 21.4.x or earlier releases, then refer to the Migration of Legacy Configuration section before proceeding. Starting with ORDS version 22.1, changes to both URL mapping and the configuration directory structure were implemented. Ensure you review these changes and then prepare your migration strategy accordingly.

Upgrading from ORDS 22.1.x or later

If you are upgrading from ORDS 22.1.x or later releases, then refer to the following sections:

- Interactive Command-Line Interface Installation
- Non-Interactive Command-Line Interface Installation (Silent)
- Migrate Configuration and Upgrade from ORDS 21.4.x or Earlier Releases
  This section describes how to upgrade from ORDS 21.4.x or earlier releases.

2.3.1 Migrate Configuration and Upgrade from ORDS 21.4.x or Earlier Releases

This section describes how to upgrade from ORDS 21.4.x or earlier releases.

To upgrade to ORDS 22.x from ORDS 21.4.x or earlier releases:

Provide the location of your configuration files that were used from your previous ORDS release (21.4.x or earlier). If you do not know the location, then specify the command using your previous ORDS release (21.4.x or earlier).

```
java -jar /path/to/earlierRelease/ords.war configdir
```

Example:

```
java -jar /path/to/earlierRelease/ords.war configdir

INFO   The config.dir value is /path/to/legacy/conf
```

In the preceding example `/path/to/legacy/conf` is configuration location

You can upgrade to ORDS 22.x release from ORDS 21.4.x or earlier releases in the following two ways:

- Interactive Upgrade
- Silent Upgrade
- Interactive Upgrade
- Silent Upgrade
- Upgrading the Multiple Database Pools

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.3.1.1 Interactive Upgrade

Interactive Upgrade

Specify the following install command to migrate your configuration, and to upgrade ORDS in your specified database:

```
ords --config <folder> install -i --legacy-config <folder> --log-folder
    <folder>
```

Table 2-23 Interactive Upgrade Commands

Command Options	Description
`--config <folder>`	Specifie the configuration folder to store your migrated configuration files. Your configuration folder must not be in the same location as your legacy configuration folder. Otherwise, an error occurs.
`-i` or `--interactive`	Specifies to run the command in interactive mode. You are then prompted for the information.
`--legacy-config <folder>`	Specifies to provide the location of your configuration files that you used for ORDS 21.4.x or earlier releases.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-23 (Cont.) Interactive Upgrade Commands

Command Options	Description
`--log-folder <folder>`	Specifies to provide the location to store your log file.

```
ords --config /path/to/new/conf install -i --legacy-config /path/to/legacy/
conf --log-folder /path/to/logs

Oracle REST Data Services 24.x.x.rNNNNNN - Migrate Configuration
Migrating ORDS Configuration files located at /path/to/legacy/conf
...

Legacy configuration files located at /path/to/legacy/conf/ords are no longer
being used
to configure ORDS or its connection pools.
Your migrated configuration files are now located at /path/to/new/conf

Oracle REST Data Services - Interactive Install

  Enter a number to select the database pool to upgrade ORDS.
  Additional options to Generate script, or Create an additional database
pool.
    [1] default      jdbc:oracle:thin:@//localhost:1521/orcl
    [2] db1          jdbc:oracle:thin:@//localhost:1521/pdb1
    [3] db2          jdbc:oracle:thin:@//localhost:1521/pdb2
    [G] Generate script to upgrade ORDS in all the database pools
    [C] Create an additional database pool
  Choose [1]:
```

A list of the migrated database pools are displayed. The legacy configuration database pool named apex is migrated, and the name is changed to default.

If you have multiple database pools to upgrade, you can either select the option Generate a script to upgrade ORDS in all the database pools, or repeat the install command again excluding the `--legacy-config` option since you have already migrated the configuration.

Example:

```
ords --config /path/to/new/conf install -i --log-folder /path/to/logs
```

> **See Also**
>
> Scenario 3: Multiple Database Pools

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.3.1.2 Silent Upgrade

To do a silent upgrade you must provide the following:

- Legacy configuration location
- The database password for the administrator user

The database password can be in a text file or using a `Here` document. If you are using a text file, then ensure to delete the text file when the upgradation is complete. The legacy configuration database pool named `apex` is migrated, and the name is changed to `default`.

If you omit the `--database-pool <name>` option, then it uses the default database pool to upgrade.

Using a Text File

You can redirect standard input using a text file that contains the password. Delete the text file once you complete the upgrade operation.

```
ords --config <folder> install --admin-user <user> --legacy-config <folder> --
log-folder <folder>
      --password-stdin <  <text-file>
```

Example of using a <text-file> with 1 password

```
The text file contains 1 password for --admin-user <user>

ords --config /path/to/new/conf install --admin-user SYS --legacy-config /
path/to/legacy/conf
--log-folder /path/to/logs --password-stdin < adminpwd.txt
```

Using a Here Document

You can redirect Standard Input using the `Here` document (also known as `heredoc`) for the password(s). The `heredoc` consists of the << redirection operator followed by a delimiter token once the upgrade operation completes. If you are using a script that contains the password(s), then delete the script.

```
ords --config <folder> install --admin-user <user> --legacy-config <folder> --
log-folder <folder> --password-stdin <<  <delimiter-token>
> <password>
> <delimiter-token>


ords --config /path/to/new/conf install --admin-user SYS --legacy-config /
path/to/legacy/conf --log-folder
        /path/to/logs --password-stdin << EOF
> <password for admin-user>
> EOF
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

2.3.1.3 Upgrading the Multiple Database Pools

Oracle highly recommends that if you have multiple database pools that uses a PL/SQL gateway user, then you must also upgrade those database pools using the `ords` install command so that the PL/SQL gateway user is configured in the ORDS schema.

2.4 Uninstalling Oracle REST Data Services

This section describes the interactive and non-interactive uninstallation CLI commands used to uninstall the ORDS schema.

- Interactive Uninstall CLI
- Non-Interactive Uninstall CLI

2.4.1 Interactive Uninstall CLI

The interactive uninstallation CLI prompts you for the necessary information to uninstall the ORDS schema, ORDS proxy user and related database objects from the database.

To be prompted, you must execute one of the following commands:

- Specify the uninstall command only. For example: `$ ords uninstall`

> **Note**
>
> Assumes that the configuration folder is specified through the environment variable or you are defaulting to the current working directory.

- Specify the option `--config <configuration folder>` followed by the uninstall command. For example: `$ ords --config /path/to/config uninstall`
- Specify the `--interactive` option. For example: `$ ords --config /path/to/config uninstall --interactive`

Table 2-24 Interactive Uninstall Prompts

Prompt Number	Prompt	Description
1.	`Enter a number to select the database pool to use or create the database pool [1] default jdbc:oracle:thin:@//localhost:1521/orcl [S] Specify the database connection Choose [1]:`	Refer to Entering a Number to Select the Database Pool

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-24 (Cont.) Interactive Uninstall Prompts

Prompt Number	Prompt	Description

2.	`Enter a number to select the database connection type to use`
	`[1] Basic (host name, port, service name)`
	`[2] TNS (TNS alias, TNS directory)`
	`[3] Custom database URL`
	`Choose [1]:`
		Refer to Entering a Number to Select the Database Connection Type

3.	`Enter the database host name [localhost]:`	Refer to Entering the Database Host Name
4.	`Enter the database listen port [1521]:`	Refer to Entering the Database Listener Port
5.	`Enter the database service name [orcl]:`	Refer to Entering the Database Service Name
6.	`Enter the TNS location: /path/to/tns`	Refer to Entering the TNS Location

7.	`Enter a number to select the TNS Network alias to use`
	`[1]`
	`DB1 ...CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=pdb1)))`
	`[2]`
	`DB2 ...CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=pdb2)))`
	`[3]`
	`DB3 ...CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=pdb3)))`
	`Choose [1]:`
		Refer to Entering a Number to Select the TNS Network Alias

8.	`Enter the Custom database URL:`	Refer to Entering the Custom Database URL
9.	`Provide database username with administrator privileges.` `Enter the administrator username: SYS`	Refer to Entering the Administrator Username

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-24 (Cont.) Interactive Uninstall Prompts

Prompt Number	Prompt	Description
10.	`Enter the database password for SYS AS SYSDBA:`	Refer to Entering the Database Password for SYS AS SYSDBA

11.	`ORDS contains 1 enabled/disabled schema: HR`
	
	`Uninstall ORDS in the database`
	`[1] Yes`
	`[2] No`
	`Choose [2]:`
		Refer to Entering an Option for Uninstalling ORDS

- Entering a Number to Select the Database Pool
- Entering a Number to Select the Database Connection Type
  Select the database connection type.
- Entering the Database Host Name
  Specify the database host name for basic connection.
- Entering the Database Listener Port
  Specify the database port for a basic connection.
- Entering the Database Service Name
- Entering the TNS Location
- Entering a Number to Select the TNS Network Alias
- Entering the Custom Database URL
- Entering the Administrator Username
- Entering the Database Password for SYS AS SYSDBA
  Specify the administrator user password.
- Entering an Option for Uninstalling ORDS

2.4.1.1 Entering a Number to Select the Database Pool

You get this prompt only if the configuration pool(s) exists in the configuration folder.

- If you select a database pool, then it bypasses the database connection type and prompts you for the administrator username and password.
- If you select option 2, then you get the prompt number 2.

> **Note**
>
> If the configuration pool(s) does not exist in your configuration folder, then it prompts you for the database connection type.

Copyright © 2011, 2026, Oracle and/or its affiliates.

2.4.1.2 Entering a Number to Select the Database Connection Type

Select the database connection type.

2.4.1.3 Entering the Database Host Name

Specify the database host name for basic connection.

2.4.1.4 Entering the Database Listener Port

Specify the database port for a basic connection.

2.4.1.5 Entering the Database Service Name

Specify the service name. The default value for the service name is obtained from one of the following in the given sequence:

1. The service name setting in the ORDS configuration file. If not present, then
2. It checks for `ORACLE_PDB` environment variable. If undefined, then
3. It checks for `ORACLE_SID` environment variable. If undefined, then
4. It uses `orcl` as the default value.

2.4.1.6 Entering the TNS Location

Specify a valid folder of where the `tnsnames.ora` file is located.

2.4.1.7 Entering a Number to Select the TNS Network Alias

Select the TNS alias name from the list of TNS network alias names displayed.

2.4.1.8 Entering the Custom Database URL

Specify the custom database URL for custom database URL connection.

Example URL formats:

```
jdbc:oracle:<driver>:@//<host>:<port>/<servicename>
jdbc:oracle:thin:@//localhost:1521/orcl

jdbc:oracle:<drivertype>:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<host>)
(PORT=<port>))(CONNECT_DATA=(SERVICE_NAME=<servicename>)))
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=orcl)))
```

2.4.1.9 Entering the Administrator Username

Specify a user who has installer privileges to install or upgrade ORDS in the database.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Privileges Granted to the Oracle REST Data Services Users

2.4.1.10 Entering the Database Password for SYS AS SYSDBA

Specify the administrator user password.

2.4.1.11 Entering an Option for Uninstalling ORDS

If the ORDS enabled or disabled schemas exist, and the `--force` command option is not specified, then it displays the enabled or disabled schemas and prompts for a confirmation to uninstall ORDS in the database.

If the `--force` command option is specified, then the uninstall confirmation is not shown,

2.4.2 Non-Interactive Uninstall CLI

The non-interactive uninstall CLI uninstalls the ORDS schema, ORDS proxy user, and related database objects from the database.

Provide the command options for the non-interactive uninstall CLI. If you are missing a required option (for example, `db-port` for a connection), then you are not prompted. Instead, an error is returned.

- Using Input Redirection
- Command options for Uninstall CLI

2.4.2.1 Using Input Redirection

This section describes how to redirect standard input using Here document or to a file.

Redirect Standard Input to a File

Redirect STDIN to a file that contains the password. In the following example, the file contains one password.

Example:

```
$ cat adminpwd.txt
password
```

The password belongs to the administrator user on the command line (for example: `SYS`). This example assumes that the config folder and the db1 pool already exists.

```
ords --config /path/to/conf uninstall --db-pool db1 --admin-user SYS --log-
folder /path/to/logs --password-stdin < adminpwd.txt
```

Redirect Standard Input using Here Document

Redirect STDIN using the `Here` document (also known as heredoc) for the password. The heredoc consists of the << redirection operator followed by a delimiter token.

The password must be on a separate line and it is ended by the delimiter token.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example:

```
ords --config <configuration_folder> uninstall --db-pool <pool_name> --admin-
user <username>
--db-hostname <host> --db-port <port_number> --db-servicename <service_name>
--log-folder <log_folder>
--password-stdin << EOF
<password>
EOF
```

The password belongs to the the administrator user on the command line. (for example: SYS).

```
ords --config /path/to/config uninstall --db-pool db1 --admin-user SYS --db-
hostname
      localhost --db-port 1521 --db-servicename orcl --log-folder /path/to/
logs << EOF
<password>
EOF
```

2.4.2.2 Command options for Uninstall CLI

Table 2-25 Command Options for Uninstall CLI

Options	Description
`--admin-user <USERNAME>`	Specifies the user with privileges to install, upgrade, repair, or uninstall ORDS in the database.
`--bequeath-connect`	Specifies that the bequeath connect enables clients to connect directly to an Oracle database bypassing the network listener. This occurs if client is on the same server as the database.
`--config <CONFIG_PATH>`	Specifies the path to the folder containing the configuration files.
`--db-pool <POOL_NAME>`	Specifies the name of the database pool.
`--db-custom-url <URL>`	Specifies the user specified custom url.
`-db-hostname <HOST>-`	Specifies the database host name.
`--db-port <PORT>`	Specifies the database port.
`--db-servicename <SERVICENAME>`	Specifies the database service name.
`--db-sid <SID>`	Specifies the database SID.
`--db-tns-alias <ALIAS_NAME>`	Specifies the tns alias name that exists in the `tnsnames.ora` file.
`-db-tns-dir <TNS_DIR>-`	Specifies the folder where the `tnsnames.ora` file is located.
`--force`	Specifies that you can uninstall ORDS from the database regardless of the number of ORDS enabled/disabled schemas.
`--help, -h`	Shows how to use the command.
`--interactive, -i`	Prompts the user for all the required information.
`--log-folder`	Writes the logs from the install, upgrade, repair, or uninstall to the log folder. If this option is omitted, then the output is written to the standard output.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 2-25 (Cont.) Command Options for Uninstall CLI

Options	Description
`-password-stdin`	Specifies when you want to run the ORDS install or uninstall command non-interactively, you can use this option to read the password value from standard input when redirecting input to a file or `here` document. If the `--password-stdin` option and the redirection symbol ("<" or "<<") are omitted, you are prompted for the password(s) for the command option(s) that requires the password.
`--pdb-open-readwrite-all`	Specifies the option applicable for CDB. Specifies whether to open all PDBs in read write mode if their status is either closed or read only.
`--pdb-open-readwrite <(PDB...)>`	Specifies the option applicable for CDB. Specifies the list of PDBs that are readonly or closed to be open for readwrite to uninstall ORDS.

2.5 Updating the Configuration Settings

The ORDS `config` command is used to update and edit the contents of the global and database pool specific settings in the configuration files.

The `config` command comprises a list of sub-commands. The config command has the `--db-pool <POOL_NAME>` option that identifies the database pool to which you want to apply the setting. If this option is omitted, then the default database pool settings get updated.

```
ords config [OPTIONS] [SUB-COMMAND]
```

Where:

- OPTIONS:
  - `--db-pool <string>`: The name of the database connection pool to configure
  - `-h, --help`: Show usage information for the command
- SUB-COMMAND: `delete`, `get`, `info`, `list`, `secret`, `set`, and `user`

Following is the list of sub-commands along with the descriptions:

`info`

Shows the description of the specified key, explaining its purpose. Indicates if it is a sensitive value and what is the default value.

Example: `ords config info <KEY>`

`set`

Sets the `config` key to a value. If you wish the setting to be set at a global level, then use the `--global` option.

Example: `ords config set [--global] <KEY> <VALUE>`

`secret`

Sets the sensitive values (for example: connection pools password). Use `ords config secret` to set the sensitive values. Some values are sensitive (example: `db.password`). Per Oracle

Copyright © 2011, 2026, Oracle and/or its affiliates.

Secure Coding Standards such values must not be passed as command line arguments as this leads to leakage of the value (appears in `ps` output).

Example: `ords config secret --password-stdin <KEY> < <file>`

> **Note**
>
> If an attempt is made to do `ords config set` on a sensitive value, an error is returned.

Non-Interactive example:

If the `--password-stdin` option is specified, then the value is read from `stdin`.

Example:

```
ords config secret --password-stdin db.password < secret.txt
```

Interactive Example

If `--password-stdin` is not specified, then the user is prompted to enter and confirm the secret value.

Example:

```
ords config secret db.password
```

`get`

Echo the config value to stdout. If the value is a secret then it's value will be replaced by the text `******`. If you wish to see the plain text of the secret, then use the `--secret` argument.

Example: `ords config get [--global] [--secret] <KEY>`

`list`

Show all explicitly configured values for the pool settings and global settings. Secret values are replaced with the text `******`. Use `--include-defaults` to list all the settings including those with the default values.

Example: `ords config list [--include-defaults]`

The information lists the name of the setting, value, and the source location (global configuration, pool specific configuration, or global/pool wallet. The table includes the columns such as

- `Setting`: Name of the setting
- `Value`: The value of the setting
- `Source`: Location of the global/pool specific setting

For example the output is similar to the following:

```
Configuration: /path/to/conf/

Database pool: default

Setting                        Value                  Source
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
-------------------         ---------------         ------
db.connectionType           basic                   Pool
db.hostname                 localhost               Pool
db.password                 ******                  Pool Wallet
db.port                     1521                    Pool
db.servicename              orcl                    Pool
db.username                 ORDS_PUBLIC_USER        Pool
feature.sdw                 true                    Pool
restEnabledSql.active       true                    Pool
```

The column Source indicates the origin of the setting value which can be any one of the following:

- `Default`: Uses the default value for the setting
- `Global`: The value is specified in the global/settings.xml configuration file
- `Pool`:The value is explicitly configured in the pool
- `Pool Wallet`:The (sensitive) value is explicitly configured in the pool wallet

`user delete`

Deletes the key from the configuration.

Example: `delete [--global]`

`user add`

Add a user and its password and role(s) in the credentials file. If the user already exists, then it is updated.

Non-interactive

Include option `--password-stdin` to read from STDIN. Requires the password file of the user.

Example:

```
ords config user add <USERNAME> <ROLES>
Enter the password:
Confirm password:
```

Interactive

Prompts for the password if `--password-stdin` option is omitted.

Example:

```
ords config user add <USERNAME> <ROLES>
Enter the password:
Confirm password:
```

`user delete`

```
ords config user delete <USERNAME>
```

Delete the specified user.

Copyright © 2011, 2026, Oracle and/or its affiliates.

user get

```
ords config user get <USERNAME>
```

Retrieve the specified user.

user list

```
ords config user list
```

Lists the users in the credential file.

> **See Also**
>
> - Understanding the Configurable Settings
> - Setting Up the Configuration Folder Structure

Copyright © 2011, 2026, Oracle and/or its affiliates.

3

Configuring Model Context Protocol (MCP)

This chapter explains how to enable the `/mcp` endpoint, configure JWT validation for authentication, and specify the ORDS database pools that are accessible through MCP.

Oracle REST Data Services (ORDS) can provide a Model Context Protocol (MCP) server that enables authenticated AI clients and developer tools to access database resources. By default, ORDS MCP is disabled. After it is enabled, ORDS uses a global MCP JWT profile for authentication and provides the MCP Client access to the database connections

The ORDS MCP runtime is supported only in ORDS standalone deployments. When ORDS is deployed on Tomcat or WebLogic, the `/mcp` endpoints are not available.

- About ORDS Model Context Protocol Configuration
- ORDS MCP Configuration Settings
- Enable the ORDS MCP Endpoint
- Configure the Global MCP JWT Profile
- Configure MCP Databases in Scope Mode
- Configure MCP Databases in Role Mode
- Verify MCP Configuration
- Troubleshoot MCP Configuration

> **See Also**
>
> Using ORDS Model Context Protocol (MCP)

3.1 About ORDS Model Context Protocol Configuration

ORDS MCP is available at the `/mcp` endpoint on the ORDS host. It is not served under the `/ords` REST-enabled schema path.

When an MCP client first accesses `/mcp`, ORDS MCP returns `401 Unauthorized` with a `WWW-Authenticate` challenge pointing to the well-known metadata endpoint, which the client uses to discover where to obtain a JWT access token and continue the OAuth 2.0 flow. ORDS validates the bearer token using the global MCP JWT Profile, including the configured issuer, audience, and JWK-backed signature validation. After the token is accepted, ORDS exposes only the MCP-enabled database connection pools that the caller is authorized to access, based on the token's configured scopes or roles.

In MCP client responses, ORDS uses the term database for a configured MCP target. In ORDS configuration, each MCP database is backed by a direct database pool with pool credentials.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Only direct database pools are eligible for ORDS MCP. A database pool configured with `mcp.scope` or `mcp.role` is reserved exclusively for MCP and is not used for REST URL mapping.

Direct connection pools are defined by the actual user to be used for the connection. This contrasts with proxy connection pools that you may used for ORDS REST APIs with `ORDS_PUBLIC_USER` that is used to proxy to REST enabled schemas. This means that the user you select for your MCP database pool will be the user used to execute all MCP Tool requests.

3.2 ORDS MCP Configuration Settings

ORDS MCP uses global settings to control endpoint availability and JWT validation:

- `feature.mcp`: Enables or disables the `/mcp` endpoint. The default value is `false`.
- `mcp.security.jwt.profile.issuer`: Specifies the expected JWT issuer.
- `mcp.security.jwt.profile.audience`: Specifies the expected JWT audience, typically the `/mcp` resource URI.
- `mcp.security.jwt.profile.jwk.url`: Specifies the JSON Web Key Set (JWKS) URL used to validate JWT signatures.
- `mcp.security.jwt.profile.authorization.server.url`: Optionally specifies the authorization server URL to include in OAuth protected resource metadata.
- `mcp.security.jwt.profile.role.claim.name`: Optionally specifies the JSON Pointer to the JWT role claim. When this setting is configured, ORDS MCP uses role mode for authorization. If it is not configured, ORDS MCP uses scope mode.

ORDS MCP uses the following pool-specific settings to expose a direct database pool through MCP:

- `mcp.scope`: Specifies the scope required to access the database pool when ORDS MCP operates in scope mode.
- `mcp.role`: Specifies the role required to access the database pool when ORDS MCP operates in role mode.
- `db.description`: Optionally specifies the database pool description returned by the `database_list` operation when the client requests detailed information.

3.3 Enable the ORDS MCP Endpoint

Enable the ORDS MCP endpoint by setting the global `feature.mcp` configuration property:

```
ords --config /path/to/conf config set --global feature.mcp true
```

If feature.`mcp` is `false` or not set, then the `/mcp` endpoint is disabled and requests to `/mcp` are not served.

3.4 Configure the Global MCP JWT Profile

To configure the JWT profile that ORDS uses to validate MCP bearer tokens, run the following ORDS CLI commands:

```
ords --config /path/to/conf config set --global
mcp.security.jwt.profile.issuer https://idp.example.com
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
ords --config /path/to/conf config set --global
mcp.security.jwt.profile.audience https://ords.example.com/mcp
ords --config /path/to/conf config set --global
mcp.security.jwt.profile.jwk.url https://idp.example.com/.well-known/jwks.json
ords --config /path/to/conf config set --global
mcp.security.jwt.profile.authorization.server.url https://idp.example.com
```

The authorization server URL is optional. When configured, ORDS includes it in the OAuth protected-resource metadata for the `/mcp` endpoint, enabling MCP clients to automatically discover the authorization server. If the authorization server URL is not configured, then ORDS MCP continues to function as long as the issuer, audience, and JWK URL settings are configured. In this case, ORDS omits the authorization_servers field from the protected-resource metadata. This enables administrators to keep the authorization server private, but requires MCP clients to be configured with the authorization server explicitly, since it cannot be discovered automatically from ORDS.

The caller token must contain the global MCP access scope:

```
urn:oracle:dbtools:ords:mcpserver:all
```

If `mcp.security.jwt.profile.role.claim.name` is not configured, ORDS MCP uses scope mode for pool authorization. If it is configured, then ORDS MCP uses role mode.

3.5 Configure MCP Databases in Scope Mode

In scope mode, each MCP database pool must specify the scope required to access the pool.

Run the following ORDS CLI commands to expose the `mcp-hr` database pool through MCP with the global MCP access scope configured as the pool scope:

```
ords --config /path/to/conf config --db-pool mcp-hr set mcp.scope
urn:oracle:dbtools:ords:mcpserver:all
ords --config /path/to/conf config --db-pool mcp-hr set db.description "HR
reporting database"
```

Run the following ORDS CLI command to require a custom pool scope:

```
ords --config /path/to/conf config --db-pool finance set mcp.scope
urn:example:ords:mcp:finance
```

For a custom pool scope, the caller token must contain both the global MCP access scope and the custom pool scope.

Example JWT Payload

```
{
  "iss": "https://idp.example.com",
  "sub": "auth0|6a3d7f722913da150ac58f3c",
  "aud": "https://ords.example.com/mcp",
  "iat": 1782493244,
  "exp": 1782579644,
  "scope": "urn:oracle:dbtools:ords:mcpserver:all",
  "azp": "tpc_5J8M7hyRHBriMaf9Zo68w9"
}
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Note

- The JWT `iss` claim matches the ORDS global setting `mcp.security.jwt.profile.issuer`
- The JWT `aud` claim matches the ORDS global setting `mcp.security.jwt.profile.audience`
- The `scope` claim includes the scope required for MCP access `"urn:oracle:dbtools:ords:mcpserver:all"`
- The `scope` claim also includes the value configured for the database connection pool's `mcp.scope` setting. In this example, the required value is also `urn:oracle:dbtools:ords:mcpserver:all`
- The JWT `iat`, `exp` and `signature` must all be valid

3.6 Configure MCP Databases in Role Mode

In role mode, configure the JWT claim that contains roles, then assign a required role to each MCP database pool.

Run the following ORDS CLI commands to read the roles from the JWT `/roles` claim and expose the `mcp-hr` database pool only to callers with the `ORDS_MCP_HR` role:

```
ords --config /path/to/conf config set --global
mcp.security.jwt.profile.role.claim.name /roles2
ords --config /path/to/conf config --db-pool mcp-hr set mcp.role ORDS_MCP_HR
```

In **role mode**, only database pools configured with `mcp.role` are exposed through MCP.

```
{
  "iss": "https://idp.example.com",
  "sub": "auth0|6a3d7f722913da150ac58f3c",
  "aud": "https://ords.example.com/mcp",
  "iat": 1782493244,
  "exp": 1782579644,
  "scope": "urn:oracle:dbtools:ords:mcpserver:all",
  "roles2": [
    "ORDS_MCP_HR"
  ],
  "azp": "tpc_5J8M7hyRHBriMaf9Zo68w9"
}
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Note

- The JWT `iss` claim matches the ORDS global setting `mcp.security.jwt.profile.issuer`
- The JWT `aud` claim matches the ORDS global setting `mcp.security.jwt.profile.audience`
- The `scope` claim provides the required scope for MCP `"urn:oracle:dbtools:ords:mcpserver:all"`
- The `roles2` claim is matched with the global setting `mcp.security.jwt.profile.role.claim.name`. The JWT `roles2` claim must provide a role that matches the `mcp.role` pool setting to access that database connection pool. In this example, the required role is `ORDS_MCP_HR`.
- The JWT `iat`, `exp` and `signature` must all be valid

3.7 Verify MCP Configuration

Start or restart ORDS after updating the configuration.

Run the following command to confirm that `/mcp` is protected:

```
curl -i https://ords.example.com/mcp
```

When ORDS MCP is enabled and the global JWT profile is fully configured, ORDS returns a Bearer authentication challenge. The challenge includes a resource metadata reference that MCP clients use for authorization discovery.

Run the following command to verify the protected-resource metadata:

```
curl -i https://ords.example.com/.well-known/oauth-protected-resource/mcp
```

If the global MCP JWT profile is incomplete, ORDS does not enable access to the `/mcp` endpoint.

3.8 Troubleshoot MCP Configuration

If the `/mcp` endpoint is not available, verify that `feature.mcp` is set to `true` in the global configuration.

If the `/mcp` endpoint returns an authentication or service error, verify that the global MCP JWT profile is fully configured with the expected issuer, audience, and JWK URL. Also verify that the identity provider is accessible and can serve the configured JWK URL.

If the MCP client authenticates successfully but no databases are returned, verify the following:

- The caller token contains the global MCP access scope
- In scope mode, the target database pool is configured with `mcp.scope`, and the caller token contains the required scope.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- In role mode, `mcp.security.jwt.profile.role.claim.name` is configured with the correct JWT role claim, the target database pool is configured with `mcp.role`, and the token contains the required role.
- The target database pool is a direct database pool configured with pool credentials.

If a database pool is no longer accessible through the standard ORDS REST URL mapping after MCP configuration, verify whether `mcp.scope` or `mcp.role` was intentionally configured for that pool. Pools with those settings are reserved for MCP.

Copyright © 2011, 2026, Oracle and/or its affiliates.

4

ORDS Configuration and Deployment Strategies

Oracle REST Data Services supports the ability to connect to more than one database. This section describes different strategies for routing requests to the appropriate database.

- Deploying ORDS with Local Configuration
- Deploying ORDS with the OCI Database API Gateway Configuration Service
  This section describes how to deploy Oracle REST Data Services (ORDS) using the OCI Database API Gateway Configuration Service.
- Deploying ORDS with Central Configuration Server
  This section describes how to deploy ORDS using a central configuration server.
- Deploying ORDS with ETCD
  This section describes how to deploy ORDS using ETCD.

4.1 Deploying ORDS with Local Configuration

By default, ORDS configuration is persisted in the ORDS configuration directory files. These files are best managed using the `ords config` command line command and can be modified manually.

- About the Request URL
- Configuring Additional Databases
- Advanced Routing
- Routing Based on the Request Path
- Routing Based on the Request Hostname

4.1.1 About the Request URL

Oracle REST Data Services supports a number of different strategies for routing requests to the appropriate database. All of these strategies rely on examining the request URL and choosing the database based on some kind of match against the URL. It is useful to recap the pertinent portions of a request URL. Consider the following URL:

```
https://www.example.com/ords/sales/f?p=1:1
```

This URL consists of the following sections:

- Protocol: `https`
- Host Name: `www.example.com`
- Context Root: `/ords`

  The context root is the location at which Oracle REST Data Services is deployed on the application server.
- Request Path: `/sales/f?p=1.1`

Copyright © 2011, 2026, Oracle and/or its affiliates.

This is the portion of the request URL relative to the context root.

For different applications, it may be important to route requests based on certain prefixes in the request path or certain prefixes in the full request URL.

There are two steps to configuring multiple databases:

1. Configuring the database connection information
2. Configuring which requests are routed to which database

4.1.2 Configuring Additional Databases

When you first configure Oracle REST Data Services, you configure a default database connection named: `default`. You can create additional database connections using the `install` command.

> **Tip**
>
> To see full help for the `install` command type:
>
> ```
> ords install --help
> ```

For adding additional database connection pool, enter the following command:

```
ords --config <configuration folder> install
```

Refer to Interactive Command-Line Interface Installation to create an additional database connecion pool.

> **See Also**
>
> - Interactive Command-Line Interface Installation
> - Installing and Configuring Oracle REST Data Services
> - Routing Based on the Request Path
> - Routing Based on the Request Hostname

4.1.3 Advanced Routing

By default, additional databases are mapped using the database pool name specified while installing ORDS. Alternatively, custom mapping can be performed based on a path prefix or a hostname as described in the following sections. Both *Routing Based on the Request Path* and *Routing Based on the Request Hostname* can be used at the same time in the same database pool configuration. Use of paths and hostname files are not mutually exclusive.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> If custom mapping is configured, then default mapping with database pool name is not enabled.

> **See Also**
>
> Configuring Additional Databases

4.1.4 Routing Based on the Request Path

You can create the request routing rules based on the request path using the paths file. To do this, perform the following steps:

1. Open or create a file `<ords_config_folder>/databases/<database_name>/paths`
2. Enter the paths that you want to be served by this database.

Example location of the paths file

```
ords_conf/
+-- databases/
|   +-- db1/
|   |   +-- wallet/
|   |   +-- paths
|   |   +-- pool.xml
|   +-- db2/
|       +-- wallet/
|       +-- paths
|       +-- pool.xml
+-- globals/
```

Example: `ords_conf/databases/db1/paths`

```
sales
```

Example: `ords_conf/databases/db2/paths`

```
support
/finance/department1
```

The previous rules match all the following requests:

```
http://example.com/ords/sales --> db1
http://example.com/ords/sales/leads --> db1
https://www.example.com/ords/sales/forecasting.report?month=jan --> db1 (If
www.example.com resolves to the same system as example.com.)
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
http://example.com/ords/support --> db2
http://example.com/ords/finance/department1 --> db2
```

4.1.5 Routing Based on the Request Hostname

You can create request routing rules based on the hostname using the hostnames file. To do this, perform the following steps:

1. Open or create a file `<ords_config_folder>/databases/<database_name>/hostnames`.
2. Add the hostnames you want to be served by this database.

Example location of hostnames files:

```
ords_conf/
+-- databases/
|   +-- db1/
|   |   +-- wallet/
|   |   +-- hostnames
|   |   +-- pool.xml
|   +-- db2/
|       +-- wallet/
|       +-- hostnames
|       +-- pool.xml
+-- globals/
```

Example: `ords_conf/databases/db1/hostnames`

```
www.example.com
example.org
```

Example `ords_conf/databases/db2/hostnames`

```
foo.bar.com
```

The preceding rules match the requests to their corresponding database connection as follows:

```
http://www.example.com/ords/ --> db1
http://example.org:8080/ords/f?p=1:1 --> db1
https://foo.bar.com/ords/myschema/resource --> db2
```

The preceding rules do not match following requests:

```
http://example.com/ords/ (The hostname is missing the www.)
http://foo.bar.net/ords/myschema/resource --> db2 (hostname is different
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

4.2 Deploying ORDS with the OCI Database API Gateway Configuration Service

This section describes how to deploy Oracle REST Data Services (ORDS) using the OCI Database API Gateway Configuration Service.

The OCI Database API Gateway Configuration Service (Configuration Service) is included with the Oracle Database Tools Service in Oracle Cloud Infrastructure (OCI). Oracle Database Tools enables users to create connections to any Oracle Database in OCI, and these connections can now be used to define ORDS pools. The Configuration Service is used to specify how ORDS is configured, including global settings and pool-level configuration. It also supports defining user-enabled resources (AutoREST objects) and user-defined resources such as modules, templates, and handlers enabling the users to create RESTful resources for their Oracle Database without installing ORDS in the database.

This OCI-managed service is designed to integrate seamlessly with ORDS, and provides all required configuration through a centralized, managed, and secured approach.

- ORDS RESTful Service Definition Locations
  This section describes the ORDS RESTful service definition locations.
- Configuration Settings
  This section outlines how configuration is structured when using the OCI Database API Gateway Configuration Service to deploy ORDS.
- OCI Database API Gateway Configuration Settings
  This section lists the OCI Database API Gateway Configuration settings.
- OCI Database API Gateway Configuration Global Settings
  This section lists the settings defined in an OCI Database API Gateway Configuration Global Settings.
- Advanced ORDS Global Settings
  This section describes the advanced ORDS global settings.
- OCI Database API Gateway Configuration Pool Settings
  This section describes the OCI Database API Gateway Configuration Pool Settings.
- Advanced ORDS Pool Settings
  This section describes the advanced ORDS pool settings.
- Auto API Specifications
  This section defines the Auto REST Objects.
- OCI Security
  This section covers the security model for the OCI Database Tools Database API Gateway Configuration Service.
- Deploying ORDS with the OCI Database API Gateway Configuration service
  This section provides the steps to deploy ORDS using the OCI Database API Gateway Configuration service.

4.2.1 ORDS RESTful Service Definition Locations

This section describes the ORDS RESTful service definition locations.

The OCI Database API Gateway Configuration Service supports the following two locations for ORDS RESTful service definitions:

Copyright © 2011, 2026, Oracle and/or its affiliates.

- **DATABASE**: RESTful Service definitions continue to be stored in the database, while ORDS operates in a more traditional configuration
- **CLOUD**: User-defined resources, including modules, templates, and handlers, along with user-enabled resources such as REST-enabled SQL and REST APIs, are configured entirely through the Database API Gateway.

In standard ORDS deployments (Local, Central, or ETCD), RESTful service definitions for user-defined and user-enabled resources are stored in the ORDS_METADATA schema in the database. ORDS uses this metadata to provide features such as AutoREST, REST-enabled SQL, and REST APIs. With the OCI Database API Gateway Configuration Service, these definitions can instead be managed directly in the gateway configuration, outside the database.

Following are the two key benefits of the OCI Database API Gateway Configuration Service:

- Automatic runtime detection of configuration changes, including new pool configurations
- No ORDS restart is required when the configuration changes are applied

DATABASE

Configures ORDS so that all pool connections use an ORDS Runtime Database user (for example: `ORDS_PUBLIC_USER`). ORDS must be installed in these databases, and all RESTful service definitions must be sourced from the database.
When service definitions are stored in the database, the following options can also be configured:

- All available ORDS configuration properties
- Database Actions
- Oracle APEX

Figure 4-1 DATABASE ORDS RESTful Service Definition Location

CLOUD

In this configuration, all RESTful service definitions including user-defined resources such as modules, templates, and handlers, as well as user-enabled resources such as AutoREST APIs are managed exclusively through the configuration. This deployment model offers maximum flexibility and enables ORDS to be used with databases where installing ORDS is not permitted.

In this configuration:

- ORDS pools are defined using OCI Database Tools Connections for enhanced security
- ORDS pool connections use a non-ORDS Runtime Database user.
- ORDS installation is not required for these databases (that is, will not reference `ORDS_METADATA` for RESTful service implementation).

Figure 4-2 CLOUD ORDS RESTful Service Definition Location

Copyright © 2011, 2026, Oracle and/or its affiliates.

4.2.2 Configuration Settings

This section outlines how configuration is structured when using the OCI Database API Gateway Configuration Service to deploy ORDS.

In an OCI Database API Gateway Configuration Service deployment, global settings and pool settings are consolidated into a single document called the ORDS Instance Configuration. If the ORDS RESTful Service Definition Location is set to CLOUD, then the ORDS Instance Configuration can also include API specifications and Auto API specifications. Users are not required to manage the underlying document structure directly. All settings can be configured through the OCI Console.

Configuration settings are logically divided into two categories:

- **First-class properties**: Core settings required for ORDS to function correctly
- **Advanced properties**: Optional ORDS settings that provide fine-grained control

Global and Pool First Class Properties can be found in Table 3-2 OCI Database API Gateway Configuration Global Settings and Table 3-3 OCI Database API Gateway Configuration Pool Settings, respectively. Any property not classified as a first-class property must be specified as an advanced property.

See Also

- About the Oracle REST Data Services Configuration Files
- OCI Database API Gateway Configuration Global Settings
- OCI Database API Gateway Configuration Pool Settings

4.2.3 OCI Database API Gateway Configuration Settings

This section lists the OCI Database API Gateway Configuration settings.

Table 4-1 OCI Database API Gateway Configuration Settings

OCI Setting Name	Description
ORDS RESTful service definition location	Specifies the ORDS RESTful service definition location. Supported values: • **DATABASE**: ORDS must be installed in all the pools databases and the pools `databaseToolsConnectionId` must connect to an `ORDS RUNTIME USER`. • **CLOUD**: All resources are defined within the `autoApiSpecs` and `apiSpecs` sections of the cloud configuration. The databases connected through the pools do not require ORDS to be installed, and all pool connections are handled as direct connections.
Name	Specifies an user-friendly name to help easily identify this configuration.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-1 (Cont.) OCI Database API Gateway Configuration Settings

OCI Setting Name	Description
Compartment	Specifies the comparment where the Database API Gateway Configuration will reside.

4.2.4 OCI Database API Gateway Configuration Global Settings

This section lists the settings defined in an OCI Database API Gateway Configuration Global Settings.

Table 4-2 OCI Database API Gateway Configuration Global Settings

OCI Setting Name	Description
`Pool route`	Specifies how the target pool is selected for an HTTP request. **PATH**: The `poolRouteValue` is specified in the request path. This is the default API value. **HEADER**: The `poolRouteValue` is provided in a request header.
`Pool routing header`	Specifies the request header name that provides the pool route value when the `poolRouteValue` is set as `HEADER`.
`Enable Database API feature`	Enables the database API feature. Supported Values: • `ENABLED` • `DISABLED`
`Http access`	Specifies the http access. `0` means it is intentionally disabled. By default the value is `0` . Ports can range from 1–65535. Binding to ports below 1024 requires elevated (root) privileges and is generally discouraged. Deployment on non-privileged ports (1024–65535) is recommended. **Note** These settings can be overridden through the ORDS command line.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-2 (Cont.) OCI Database API Gateway Configuration Global Settings

OCI Setting Name	Description
`Https access`	Specifies the https access. A value of `0` indicates the feature is intentionally disabled. By default, it is set to 8443. Ports can range from 1–65535. Binding to ports below 1024 requires elevated (root) privileges and is generally discouraged. Deployment on non-privileged ports (1024–65535) is recommended. If this is set without specifying `httpsCertificatePath` and `httpsCertificateKeyPath`, ORDS generates and uses a self-signed certificate. This self signed certificate would be persisted on the file system based on the ORDS home.
`Document root`	Specifies the document root used to serve static content. this setting is required if the ORDS instances serve static content (for example: APEX images). APEX images are resolved using the document root together with the APEX image context path. Default value is `/i`. If the value is not set, then ORDS does not serve any static resources from disk. **Note** Oracle recommends the use of the Oracle CDN for APEX images.
`Certificate Bundle`	Specifies whether to use a self-signed certificate (for testing only) or to provide certificate details to enable HTTPS.
`SSL Public certificate path`	Specifies the path to the certificate.
`SSL public certificate format`	Specifies the SSL public certificate format. `PEM` is the only current valid format.
`SSL Private key path`	Specifies SSL Private key path
`SSL private key format`	Specifies SSL private key format. Supported values: • `DER` • `PEM`

4.2.5 Advanced ORDS Global Settings

This section describes the advanced ORDS global settings.

All global settings described in About the Oracle REST Data Services Configuration Files section can be used as advanced properties, except the following:

- The settings that are already defined as first-class properties

Copyright © 2011, 2026, Oracle and/or its affiliates.

- The database-related settings that are restricted. For example: `db.username`, `db.password`, and `db.*`

4.2.6 OCI Database API Gateway Configuration Pool Settings

This section describes the OCI Database API Gateway Configuration Pool Settings.

Table 4-3 OCI Database API Gateway Configuration Pool Settings

OCI Setting Name	Description
`Name`	Specifies a user-friendly name to easily identify this pool.
`Connection In Compartment`	Specifies the Compartment where the Database Tool Connection resides.
`Pool route value`	Specifies the string used by ORDS to identify the target pool. ORDS can be globally configured to inspect the path or a header for the value.
`Connection`	Specifies the Database Tools Connection OCID used to create the connection pool.
`Enable Database Actions`	Enables Database Actions. Database Actions, is a web-based interface that provides development, data tools, administration and monitoring features for Oracle Database.
`Enable REST-Enabled SQL Service`	Enables REST-Enabled SQL Service. REST-Enabled SQL is a REST API that lets you run ad-hoc SQL and SQL scripts. You can POST one or more SQL statements to the service. **Note** Database Actions requires this feature.
`Max pool size`	Specifies the maximum number of database connections a pool can create at a time.
`Min pool size`	Specifies the minimum number of database connections that a pool can have.
`Initial pool size`	Specifies the initial number of database connections required to create a pool.
`JWKS URL`	Specifies the JWKS URL used to validate JWT token signatures. **Note** This setting is required for Bearer Authentication.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-3 (Cont.) OCI Database API Gateway Configuration Pool Settings

OCI Setting Name	Description
`Allowed Issuer`	Specifies the issuer for accepted JWTs. The `iss` claim of the token must match this value. **Note** This setting is required for Bearer Authentication.
`Allowed Audience`	Specifies the audience for accepted JWTs. The `aud` claim of the token match this value. **Note** This setting is required for Bearer Authentication.
`Role Claim Name`	This setting is optional. If this value is not provided, Scope-Based Access Control is used. If it is provided, Role-Based Access Control is used. It specifies the JSON pointer to the claim in the JWT that contains the roles of the user. The claim name must be a valid JSON pointer and must start with a forward slash (for example: `/roles`).

4.2.7 Advanced ORDS Pool Settings

This section describes the advanced ORDS pool settings.

All global settings described in About the Oracle REST Data Services Configuration Files section can be used as advanced properties, except the following:

- The settings that are already defined as first-class properties
- The database-related settings that are restricted. For example: `db.username`, `db.password`, and `db.*`

4.2.8 Auto API Specifications

This section defines the Auto REST Objects.

> **Note**
>
> This section is applicable only for CLOUD connections. Auto API specifications are supported only when the OCI Database API Gateway configuration for the ORDS RESTful Service Definition Location is set to CLOUD.

Copyright © 2011, 2026, Oracle and/or its affiliates.

The `autoApiSpec` section defines Auto REST Objects. Auto REST Objects must be present in the schema associated with the connection pool. Auto API specifications are supported only when the OCI Database API Gateway configuration for ORDS RESTful Service Definition Location is set to CLOUD.

Option	Description
`Name`	Specifies a user-friendly name to easily identify the resource.
`Description`	Specifies the description to help document the resource.
`Database object name`	Specifies the name of the database object.
`Auto REST Objects`	Specifies the type of the Object. Supported values: • `FUNCTION` • `MVIEW` • `PACKAGE` • `PROCEDURE` • `TABLE` • `VIEW` • `DUALITYVIEW`
`Alias`	Specifies the URI path element. Defaults to the lowercase object name. Must be unique within the pool.This option is optional.
`Operations`	Supports READ and WRITE operations. Supported values: • `READ` • `WRITE`
`Security Schemes`	Defines the security schemes for the resource. Supported values: • `BASIC` • `BEARER` If none of the values are specified, then the resource is public. If value `BASIC` is specified, then the resource is accessible using basic authentication. If value `BEARER` is specified, then the resource is accessible by valid Bearer Tokens (the resource is accessible by any authorized user) . A JWT Profile has to be defined for the pool..
`Scope`	Specifies the scope that protects this resource. The JWT must provide this scope to access this resource. This requires the BEARER to be listed in the Security Schemes. Used with Scope-Based JWT profiles.
`Role`	Specifies the role that protects the resource. The JWT must include the specified role to grant access to the resource. One or more (max 20) roles can be specified. This requires the BEARER to be listed in the Security Schemes. Used with Role-Based JWT profiles.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- API Specifications
  This section applies exclusively to CLOUD configurations of the OCI Database API Gateway. It defines user-created REST resources, similar to the Module > Template > Handler hierarchy used in standard ORDS configurations.

4.2.8.1 API Specifications

This section applies exclusively to CLOUD configurations of the OCI Database API Gateway. It defines user-created REST resources, similar to the Module > Template > Handler hierarchy used in standard ORDS configurations.

These user-defined REST resources are distinct from traditional ORDS REST modules and are supported only for OCI Database API Gateway configurations.

User-defined REST resources conform to the OpenAPI 3.1.x specification and must satisfy the following requirements:

- The definition must be in JSON format.
- The definition must comply with OpenAPI 3.1.x standards.

> **See Also**
>
> OpenAPI Specification Extensions the "extensions" are a subset of the entire specification.

> **Note**
>
> User-defined REST resources support OpenAPI specification extensions. These extension properties are identified by the `x-` prefix. For more information, see the OpenAPI Specification Extensions documentation.

Table 4-4 OpenAPI fields

Field	Optional/Mandatory	Type	Description
`servers`	optional	Not applicable	Specifies the base path for all API specification paths. Only relative path is supported. These form the base path of all `apiSpec` paths.
`security`	optional	Not applicable	Specifies the security required to access this API
`components.securitySchemes."<securitySchemeName>".type`	optional	Not applicable	Specifies the security scheme type used by ORDS. `"http"` : Support Basic Authentication in ORDS `"oauth2"` and `"openIdConnect"` : Supports JWT Profile in ORDS

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-4 (Cont.) OpenAPI fields

Field	Optional/Mandatory	Type	Description
components.securitySchemes."<securitySchemeName>".scheme	optional	Not applicable	Specifies the scheme of security definition. "basic", "bearer". Used to identify if JWT Profile or Basic Authentication is required
paths."<path>"	Not applicable	string	Specifies the path to the resource.
paths."<path>"."<operation>"	Not applicable	object	Specifies the operation for a resource.
paths."<path>"."<operation>".security	optional	object	Specifies the security required to access the resource.
x-dbtools-properties	Not applicable	Not applicable	ORDS extension of OpenAPI specification with information on how ORDS provides the resources.
x-dbtools-properties.originsAllowed	optional	array	Specifies the permitted origins for the apiSpec
x-dbtools-properties.itemsPerPage	optional	number	Specifies the default pagination for a resource handler HTTP operation GET method. That is, the number of rows to return on each page of a JSON format result set based on a database query.
x-dbtools-properties.published	optional	string	Specifies the publication status. Supported values: • "PUBLISHED" • "NOT_PUBLISHED" Default value: "PUBLISHED"
x-dbtools-properties.description	optional	string	Specifies the developer description of the apiSpec definition.
paths."<path>".x-dbtools-etag	optional	Not applicable	Specifies the ORDS extension of the Open API specification with the information on how ORDS provides this paths resource.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-4 (Cont.) OpenAPI fields

Field	Optional/Mandatory	Type	Description
`paths."<path>".x-dbtools-etag.httpEntityTagType`	Not applicable	string	Specifies the type of entity tag to be used by the resource path. An entity tag is an HTTP header that acts as a version identifier for a resource. Use entity tag headers to avoid retrieving previously retrieved resources and to perform optimistic locking while updating the resources. Supported values: • `"HASH"` • `"QUERY"` • `"NONE"` Default value: `"HASH"`
`paths."<path>".x-dbtools-etag.eTagQuery`	Not applicable	string	Specifies the query used to generate the ETag If the `httpEntityTagType` value is `"QUERY"`, then the query to generate the ETag can be provided here.
`paths."<path>".x-dbtools-etag.description`	optional	Not applicable	Specifies the developer description of the `apiSpec` definition.
`paths."<path>"."<operation>".x-dbtools-operation`	Not applicable	object	Specifies the ORDS extension of the Open API specification with information on how ORDS provides this resource operation.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-4 (Cont.) OpenAPI fields

Field	Optional/Mandatory	Type	Description
`paths."<path>"."<operation>".x-dbtools-operation.sourceType`	Not applicable	string	Specifies the HTTP request method for this handler. Refer to ORDS.CREATE_SERVICE. Supported values: • `"collection_feed"` • `"collection_item"` • `"media"` • `"plsql"` • `"query"` • `"csv_query"` • `"query_one_row"` • `"feed"` • `"mle_javascript"`
`paths."<path>"."<operation>".x-dbtools-operation.source`	Not applicable	string	Specifies the source implementation for the selected HTTP method.
`paths."<path>"."<operation>".x-dbtools-operation.mleEnvName`	optional	string	Specifies the multilingual Engine (MLE) environment available for this operation.
`paths."<path>"."<operation>".x-dbtools-operation.mimesAllowed`	optional	string	Applies to PUT and POST operations only. Comma separated list of MIME types that the operation accepts.
`paths."<path>"."<operation>".x-dbtools-operation.description`	optional	string	Specifies the developer comments.
`paths."<path>"."<operation>".x-dbtools-operation.parameters`	optional	Not applicable	Specifies an array of parameter definitions that ORDS can interpret.
`paths."<path>"."<operation>".x-dbtools-operation.parameters[0].name`	Not applicable	string	Specifies the name of the parameter as it is named in the URI Template or the HTTP Header. Used to map the names that are not valid SQL parameter names.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 4-4 (Cont.) OpenAPI fields

Field	Optional/Mandatory	Type	Description
`paths."<path>"."<operation>".x-dbtools-operation.parameters[0].bindVariable`	Not applicable	string	Specifies the name of the parameter as it is referred to in the SQL. If a `NULL` value is specified, then the parameter is unbound.
`paths."<path>"."<operation>".x-dbtools-operation.parameters[0].sourceType`	Not applicable	string	Specifies the type that is identified if the parameter originates in the URI Template or an HTTP Header. Supported values: • `"HEADER"` • `"RESPONSE"` • `"URI"`
`paths."<path>"."<operation>".x-dbtools-operation.parameters[0].accessMethod`	Not applicable	string	Specifies the access method of the parameter. Supported values: • `"IN"` • `"OUT"` • `"INOUT"`
`paths."<path>"."<operation>".x-dbtools-operation.parameters[0].parameterType`	Not applicable	string	Specifies the native type of the parameter. Supported values: • `"STRING"` • `"INT"` • `"DOUBLE"` • `"BOOLEAN"` • `"LONG"` • `"TIMESTAMP"` • `"RESULTSET"`
`paths."<path>"."<operation>".x-dbtools-operation.parameters[0].description`	Not applicable	string	Specifies the developer comments.

API Specifications

The following table compares a typical ORDS PL/SQL Resource Module export (first column), its corresponding OpenAPI 3.0.x specification (second column), and a sample OpenAPI 3.1.x specification (third column).

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> This examples use the HR sample schema available from the Oracle Database Sample Schemas repository: https://github.com/oracle-samples/db-sample-schemas. To reproduce the example, create the `HR` schema in your database and populate the `REGIONS` table. You can then upload the 3.1.x OpenAPI specification and test the service using the sample data.

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

DECLARE
 	
  l_roles
OWA.VC_ARR;
  l_modules
OWA.VC_ARR;
  l_patterns
OWA.VC_ARR;
 	
BEGIN
  ORDS.DEFINE_MODULE(
  p_module_name
=> 'hr.regions',
  p_base_path
=> '/regions/',
  p_items_per_page
=> 25,
  p_status
=> 'PUBLISHED',
  p_comments
=> NULL);
 	
  ORDS.DEFINE_TEMPLATE(
  p_module_name
=> 'hr.regions',
  p_pattern
=> 'v1/:id',
  p_priority
=> 0,
  p_etag_type
=> 'HASH',
  p_etag_query
=> NULL,
  p_comments
=> NULL);
 	
  ORDS.DEFINE_HANDLER(
  p_module_name
=> 'hr.regions',
  p_pattern
=> 'v1/:id',
  p_method
=> 'POST',
  p_source_type
=> 'plsql/block',
  p_mimes_allowed
=> NULL,
  p_comments
=> NULL,
  p_source
=>
	{
	  "openapi": "3.0.0",
	  "info": {
	  "title": "ORDS
	generated API for
	hr.regions",
	  "version":
	"1.0.0"
	  },
	  "servers": [
	  {
	  "url":
	"http://localhost:8080/
	ords/hr/regions"
	  }
	  ],
	  "paths": {
	  "/v1/{id}": {
	  "delete": {
	
	"description": "Remove
	a record from
	hr.regions",
	
	"responses": {
	
	"204": {
	
	"description": "Deleted
	result."
	  }
	  },
	
	"parameters": [
	  {
	
	"name": "id",
	
	"in": "path",
	
	"required": true,
	
	"schema": {
	
	  "type": "string",
	
	  "pattern": "^[^/]+$"
	
	},
	
	"description":
		{
		  "openapi": "3.1.0",
		  "info": {
		  "title": "HR
		Regions API Spec",
		  "version": "1.0.0"
		  },
		  "paths": {
		  "/regions/v1/{id}":
		{
		  "get": {
		  "operationId":
		"get_regions_v1_id",
		  "parameters": [
		  {
		  "name":
		"id",
		  "in":
		"path",
		  "required":
		true,
		  "schema": {
		  "type":
		"string"
		  }
		  }
		  ],
		  "responses": {
		  "200": {
		
		"description":
		"Response",
		  "content": {
		
		"application/json": {
		
		"schema": {
		
		"type": "array",
		
		"items": {
		
		"type": "object",
		
		"additionalProperties":
		true
		  }
		  }
		  }
		  }
		  }

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

'BEGIN
IF :region_id <= 50 THEN
:status_code := 422;
:status := ''error'';
:message := ''REGION_ID must be greater than 50.'';
ELSE
INSERT INTO regions (region_id, region_name)
VALUES (:region_id, :region_name);
 	
:status_code := 201;
:status := ''success'';
:message := ''Region was created.'';
END IF;
END;');
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'POST',
 	
p_name => 'region_id',
 	
p_bind_variable_name => 'region_id',
 	
p_source_type => 'HEADER',
 	
p_param_type => 'INT',
 	
p_access_method =>
	"implicit"
	},
	{
	"name": "l_id",
	
	"in": "header",
	
	"schema": {
	
	"type": "string"
	
	}
	}
	]
	},
	"post": {
	
	"description": "Create a new record on hr.regions",
	
	"responses": {
	
	"201": {
	
	"description": "The successfully created record.",
	
	"content": {
	
	"application/json": {
	
	"schema": {
	
	"type": "object",
	
	"properties": {
	
	"message": {
	
	"type": "string"
	
	},
		},
		"x-dbtools-operation": {
		"moduleName": "hr.regions",
		"pattern": "v1/:id?",
		"method": "GET",
		"sourceType": "json/collection",
		"source": "SELECT r.region_name,\n c.country_name\n FROM regions r\n JOIN countries c\n ON c.region_id = r.region_id\n WHERE r.region_id = :id",
		"parameters": []
		}
		},
		"post": {
		"operationId": "post_regions_v1_id",
		"parameters": [
		{
		"name": "id",
		"in": "path",
		"required": true,
		"schema": {
		"type": "string"
		}
		},
		{
		"name": "region_id",
		"in": "header",
		"required": false,
		"schema": {
		"type": "integer"
		}

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

'IN',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'POST',
 	
p_name => 'region_name',
 	
p_bind_variable_name => 'region_name',
 	
p_source_type => 'HEADER',
 	
p_param_type => 'STRING',
 	
p_access_method => 'IN',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'POST',
 	
p_name => 'status',
 	
p_bind_variable_name => 'status',
	"status": {
	
	"type": "string"
	}
	}
	}
	}
	}
	},
	"parameters": [
	{
	
	"name": "id",
	
	"in": "path",
	
	"required": true,
	
	"schema": {
	"type": "string",
	
	"pattern": "^[^/]+$"
	
	},
	"description": "implicit"
	},
	{
	
	"name": "region_id",
	
	"in": "header",
	
	"schema": {
	"type": "integer"
	
	}
	},
	{
		},
		{
		"name": "region_name",
		"in": "header",
		"required": false,
		"schema": {
		"type": "string"
		}
		}
		],
		"responses": {
		"200": {
		
		"description": "Response",
		"headers": {
		"X-ORDS-STATUS-CODE": {
		
		"schema": {
		
		"type": "integer"
		}
		}
		},
		"content": {
		
		"application/json": {
		
		"schema": {
		
		"$ref": "#/components/schemas/RegionActionResponse"
		}
		}
		}
		},
		"201": {
		
		"description": "Region was created.",
		"content": {
		
		"application/json": {
		
		"schema": {

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

p_source_type => 'RESPONSE',
 	
p_param_type => 'STRING',
 	
p_access_method => 'OUT',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'POST',
 	
p_name => 'message',
 	
p_bind_variable_name => 'message',
 	
p_source_type => 'RESPONSE',
 	
p_param_type => 'STRING',
 	
p_access_method => 'OUT',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'POST',
	"name": "region_name",
	
	"in": "header",
	
	"schema": {
	
	"type": "string"
	
	}
	
	}
	]
	},
	"put": {
	
	"description": "Create or update a record on hr.regions",
	
	"responses": {
	
	"200": {
	
	"description": "The successfully updated record.",
	
	"content": {
	
	"application/json": {
	
	"schema": {
	
	"type": "object",
	
	"properties": {
	
	"message": {
	
	"type": "string"
	
	},
	
	"status": {
		"$ref": "#/components/schemas/RegionActionResponse"
		}
		}
		}
		},
		"422": {
		
		"description": "Validation error.",
		"content": {
		
		"application/json": {
		
		"schema": {
		
		"$ref": "#/components/schemas/RegionActionResponse"
		}
		}
		}
		}
		},
		"x-dbtools-operation": {
		"moduleName": "hr.regions",
		"pattern": "v1/:id?",
		"method": "POST",
		"sourceType": "plsql/block",
		"source": "BEGIN IF :region_id <= 50 THEN :status_code := 422; :status := ''error''; :message := ''REGION_ID must be greater than 50.''; ELSE INSERT INTO regions (region_id, region_name) VALUES (:region_id, :region_name); :status_code := 201; :status := ''success''; :message := ''Region was

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

p_name => 'X-ORDS-STATUS-CODE',
p_bind_variable_name => 'status_code',
p_source_type => 'HEADER',
p_param_type => 'INT',
p_access_method => 'OUT',
p_comments => NULL);
 	
ORDS.DEFINE_HANDLER(
p_module_name => 'hr.regions',
p_pattern => 'v1/:id',
p_method => 'DELETE',
p_source_type => 'plsql/block',
p_mimes_allowed => NULL,
p_comments => NULL,
p_source =>
'DECLARE
child_exists EXCEPTION;
PRAGMA EXCEPTION_INIT(child_exists, -2292);
 	
existing_count INTEGER;
BEGIN
DELETE FROM regions
WHERE region_id > :id;
 	
IF SQL%ROWCOUNT = 0 THEN
:status_code :=
	"type": "string"
	}
	}
	}
	}
	"201": {
	"description": "The successfully created record.",
	"content": {
	"application/json": {
	"schema": {
	"type": "object",
	"properties": {
	"message": {
	"type": "string"
	},
	"status": {
	"type": "string"
	}
	}
	}
		created.''; END IF;
		END;",
		"parameters": [
		{
		"name": "region_id",
		"bindVariable": "region_id",
		"sourceType": "HEADER",
		"parameterType": "INT",
		"accessMethod": "IN",
		"description": ""
		},
		{
		"name": "region_name",
		"bindVariable": "region_name",
		"sourceType": "HEADER",
		"parameterType": "STRING",
		"accessMethod": "IN",
		"description": ""
		},
		{
		"name": "status",
		"bindVariable": "status",
		"sourceType": "RESPONSE",
		"parameterType": "STRING",
		"accessMethod": "OUT",
		"description": ""

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

404;
:status :=
''error'';
:message :=
''Region '' || :id ||
'' was not found.'';
ELSE
:status_code :=
200;
:status :=
''success'';
:message :=
''Region '' || :id ||
'' was deleted.'';
END IF;
 	
COMMIT;
 	
EXCEPTION
WHEN child_exists
THEN
SELECT COUNT(*)
INTO
existing_count
FROM countries
WHERE region_id
> :id;
 	
:status_code :=
422;
:status :=
''error'';
:message :=
''This region still has
'' || existing_count ||
''
countries assigned,
cannot delete.'';
END;');
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name =>
	}
	
	}
	}
	},
	"parameters": [
	{
	
	"name": "id",
	
	"in": "path",
	
	"required": true,
	
	"schema": {
	
	"type": "string",
	
	"pattern": "^[^/]+$"
	
	},
	
	"description":
	"implicit"
	},
	{
	
	"name": "region_id",
	
	"in": "header",
	
	"schema": {
	
	"type": "string"
	
	}
	},
	{
	
	"name": "region_name",
	
	"in": "header",
	
	"schema": {
	
	"type": "string"
	
	}
		},
		{
		"name":
		"message",
		
		"bindVariable":
		"message",
		
		"sourceType":
		"RESPONSE",
		
		"parameterType":
		"STRING",
		
		"accessMethod": "OUT",
		
		"description": ""
		},
		{
		"name":
		"X-ORDS-STATUS-CODE",
		
		"bindVariable":
		"status_code",
		
		"sourceType": "HEADER",
		
		"parameterType": "INT",
		
		"accessMethod": "OUT",
		
		"description": ""
		}
		]
		}
		},
		"put": {
		"operationId":
		"put_regions_v1_id",
		"parameters": [
		{
		"name":
		"id",
		"in":
		"path",
		"required":
		true,
		"schema": {
		"type":
		"string"
		}

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

p_name => 'status',
 	
p_bind_variable_name => 'status',
 	
p_source_type => 'RESPONSE',
 	
p_param_type => 'STRING',
 	
p_access_method => 'OUT',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'DELETE',
 	
p_name => 'message',
 	
p_bind_variable_name => 'message',
 	
p_source_type => 'RESPONSE',
 	
p_param_type => 'STRING',
 	
p_access_method => 'OUT',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
	}
	]
	},
	"get": {
	
	"description": "Retrieve records from hr.regions",
	
	"responses": {
	
	"200": {
	
	"description": "The queried record.",
	
	"content": {
	
	"application/json": {
	
	"schema": {
	
	"type": "object",
	
	"properties": {
	
	"items": {
	
	"type": "array",
	
	"items": {
	
	"type": "object",
	
	"properties": {
	
	"country_name": {
	
	"$ref": "#/components/schemas/VARCHAR2"
		},
		{
		"name": "region_name",
		"in": "header",
		"required": false,
		"schema": {
		"type": "string"
		}
		},
		{
		"name": "region_id",
		"in": "header",
		"required": false,
		"schema": {
		"type": "string"
		}
		}
		],
		"responses": {
		"200": {
		
		"description": "Response",
		"headers": {
		"X-ORDS-STATUS-CODE": {
		
		"schema": {
		
		"type": "integer"
		}
		}
		},
		"content": {
		
		"application/json": {
		
		"schema": {
		
		"$ref": "#/components/schemas/RegionActionResponse"
		}

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

p_pattern => 'v1/:id',
p_method => 'DELETE',
p_name => 'l_id',
p_bind_variable_name => 'l_id',
p_source_type => 'HEADER',
p_param_type => 'STRING',
p_access_method => 'IN',
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
p_module_name => 'hr.regions',
p_pattern => 'v1/:id',
p_method => 'DELETE',
p_name => 'X-ORDS-STATUS-CODE',
p_bind_variable_name => 'status_code',
p_source_type => 'HEADER',
p_param_type => 'INT',
p_access_method => 'OUT',
	},
	"region_name": {
	"$ref": "#/components/schemas/VARCHAR2"
	}
	}
	}
	}
	}
	}
	},
	"parameters": [
	{
	"name": "id",
	"in": "path",
	"required": true,
	"schema": {
	"type": "string",
	"pattern": "^[^/]+$"
	},
	"description": "implicit"
		}
		}
		},
		"404": {
		"description": "Region was not found.",
		"content": {
		"application/json": {
		"schema": {
		"$ref": "#/components/schemas/RegionActionResponse"
		}
		}
		}
		}
		},
		"x-dbtools-operation": {
		"moduleName": "hr.regions",
		"pattern": "v1/:id?",
		"method": "PUT",
		"sourceType": "plsql/block",
		"source": "BEGIN UPDATE regions SET region_name = :region_name WHERE region_id = :region_id AND region_id > 50; IF SQL%ROWCOUNT = 0 THEN :status_code := 404; :status := ''error''; :message := ''Region was not found, or REGION_ID is not greater than 50.''; ELSE :status_code := 200; :status := ''success''; :message := ''Region was updated.''; END IF; END;",
		"parameters":

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

p_comments => NULL);
 	
ORDS.DEFINE_HANDLER(
p_module_name => 'hr.regions',
p_pattern => 'v1/:id',
p_method => 'GET',
p_source_type => 'json/collection',
p_items_per_page => 25,
p_mimes_allowed => NULL,
p_comments => NULL,
p_source =>
'SELECT r.region_name,
c.country_name
FROM regions r
JOIN countries c
ON c.region_id = r.region_id
WHERE r.region_id = :id');
 	
ORDS.DEFINE_HANDLER(
p_module_name => 'hr.regions',
p_pattern => 'v1/:id',
p_method => 'PUT',
p_source_type => 'plsql/block',
p_items_per_page => 25,
p_mimes_allowed => NULL,
p_comments => NULL,
p_source =>
'BEGIN
UPDATE regions
SET region_name = :region_name
WHERE region_id
	}
	]
	}
	}
	},
	"components": {
	"schemas": {
	"VARCHAR2": {
	"type": "string"
	}
	}
	}
	}
		[
		{
		"name": "region_name",
		"bindVariable": "region_name",
		"sourceType": "HEADER",
		"parameterType": "STRING",
		"accessMethod": "IN",
		"description": ""
		},
		{
		"name": "region_id",
		"bindVariable": "region_id",
		"sourceType": "HEADER",
		"parameterType": "STRING",
		"accessMethod": "IN",
		"description": ""
		},
		{
		"name": "status",
		"bindVariable": "status",
		"sourceType": "RESPONSE",
		"parameterType": "STRING",
		"accessMethod": "OUT",
		"description": ""
		},
		{

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

```
= :region_id
      AND region_id >
50;

    IF SQL%ROWCOUNT = 0
THEN
      :status_code :=
404;
      :status :=
''error'';
      :message :=
''Region was not found,
or REGION_ID is not
greater than 50.'';
    ELSE
      :status_code :=
200;
      :status :=
''success'';
      :message :=
''Region was updated.'';
    END IF;
  END;');

  ORDS.DEFINE_PARAMETER(

p_module_name        =>
'hr.regions',

p_pattern            =>
'v1/:id',

p_method             =>
'PUT',

p_name               =>
'region_name',

p_bind_variable_name =>
'region_name',

p_source_type        =>
'HEADER',

p_param_type         =>
'STRING',

p_access_method      =>
'IN',

p_comments           =>
```

```
                "name":
"message",

"bindVariable":
"message",

"sourceType":
"RESPONSE",

"parameterType":
"STRING",

"accessMethod": "OUT",

"description": ""
            },
            {
              "name":
"X-ORDS-STATUS-CODE",

"bindVariable":
"status_code",

"sourceType": "HEADER",

"parameterType": "INT",

"accessMethod": "OUT",

"description": ""
            }
          ]
        }
      },
      "delete": {
        "operationId":
"delete_regions_v1_id",
        "parameters": [
          {
            "name":
"id",
            "in":
"path",
            "required":
true,
            "schema": {
              "type":
"string"
            }
          },
          {
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

```
NULL);

  ORDS.DEFINE_PARAMETER(

p_module_name        =>
'hr.regions',

p_pattern            =>
'v1/:id',

p_method             =>
'PUT',

p_name               =>
'region_id',

p_bind_variable_name =>
'region_id',

p_source_type        =>
'HEADER',

p_param_type         =>
'STRING',

p_access_method      =>
'IN',

p_comments           =>
NULL);

  ORDS.DEFINE_PARAMETER(

p_module_name        =>
'hr.regions',

p_pattern            =>
'v1/:id',

p_method             =>
'PUT',

p_name               =>
'status',

p_bind_variable_name =>
'status',

p_source_type        =>
'RESPONSE',
```

```
            "name":
"l_id",
            "in":
"header",
            "required":
false,
            "schema": {
              "type":
"string"
            }
          }
        ],
        "responses": {
          "200": {

"description":
"Response",
            "headers": {
              "X-ORDS-
STATUS-CODE": {

"schema": {

"type": "integer"
                }
              }
            },
            "content": {

"application/json": {

"schema": {

"$ref": "#/components/
schemas/
RegionActionResponse"
                }
              }
            }
          },
          "404": {

"description": "Region
was not found.",
            "content": {

"application/json": {

"schema": {

"$ref": "#/components/
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

p_param_type => 'STRING',
 	
p_access_method => 'OUT',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'PUT',
 	
p_name => 'message',
 	
p_bind_variable_name => 'message',
 	
p_source_type => 'RESPONSE',
 	
p_param_type => 'STRING',
 	
p_access_method => 'OUT',
 	
p_comments => NULL);
 	
ORDS.DEFINE_PARAMETER(
 	
p_module_name => 'hr.regions',
 	
p_pattern => 'v1/:id',
 	
p_method => 'PUT',
 	
p_name => 'X-ORDS-STATUS-CODE',
		schemas/
		RegionActionResponse"
		}
		}
		}
		},
		"422": {
		
		"description": "Region still has countries assigned.",
		"content": {
		
		"application/json": {
		
		"schema": {
		
		"$ref": "#/components/schemas/
		RegionActionResponse"
		}
		}
		}
		}
		},
		"x-dbtools-operation": {
		"moduleName": "hr.regions",
		"pattern": "v1/:id?",
		"method": "DELETE",
		"sourceType": "plsql/block",
		
		"source": "DECLARE child_exists EXCEPTION; PRAGMA EXCEPTION_INIT(child_exists, -2292);
		
		existing_count INTEGER; BEGIN DELETE FROM regions WHERE region_id > :id;
		IF SQL%ROWCOUNT = 0 THEN :status_code := 404; :status := ''error''';

Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

```
p_bind_variable_name =>
'status_code',

p_source_type        =>
'HEADER',

p_param_type         =>
'INT',

p_access_method      =>
'OUT',

p_comments           =>
NULL);

COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;
    RAISE;

END;
```

```
           :message :=
''Region '' || :id ||
'' was not found.'';
ELSE :status_code :=
200; :status
           :=
''success''; :message :=
 ''Region '' || :id ||
'' was deleted.''; END
IF; COMMIT;
           EXCEPTION
WHEN child_exists THEN
SELECT COUNT(*) INTO
existing_count FROM
countries WHERE
region_id >
           :id; :status_
code := 422; :status :=
''error''; :message :=
''This region still has
           '' ||
existing_count || ''
countries assigned,
cannot delete.''; END;",

"parameters": [
            {
              "name":
"status",

"bindVariable":
"status",

"sourceType":
"RESPONSE",

"parameterType":
"STRING",

"accessMethod": "OUT",

"description": ""
            },
            {
              "name":
"message",

"bindVariable":
"message",

"sourceType":
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

```
"RESPONSE",

"parameterType":
"STRING",

"accessMethod": "OUT",

"description": ""
            },
            {
              "name":
"l_id",

"bindVariable": "l_id",

"sourceType": "HEADER",

"parameterType":
"STRING",

"accessMethod": "IN",

"description": ""
            },
            {
              "name":
"X-ORDS-STATUS-CODE",

"bindVariable":
"status_code",

"sourceType": "HEADER",

"parameterType": "INT",

"accessMethod": "OUT",

"description": ""
            }
          ]
        }
      }
    }
  },
  "components": {
    "schemas": {

"RegionActionResponse":
{
        "type":
"object",
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

PL/SQL Module Export	OpenAPI Specification (3.0.x)	OpenAPI Specification (3.1.x)

		<pre>"properties": {
		"status": {
		"type":
		"string"
		},
		"message": {
		"type":
		"string"
		}
		},
		
		"additionalProperties":
		true
		}
		}
		},
		"x-dbtools-
		properties": {
		"itemsPerPage": 25,
		"published":
		"PUBLISHED"
		}
		}</pre>

4.2.9 OCI Security

This section covers the security model for the OCI Database Tools Database API Gateway Configuration Service.

The OCI Database Tools Database API Gateway Configuration service is secured using OCI Identity and Access Management (IAM) and integrates natively with authentication and authorization mechanisms of OCI.

ORDS can authenticate with OCI using any of the following supported methods:

- – OCI Instance Principal authentication
  - – OCI configuration profile authentication

OCI Instance Principal authentication

Instance Principal authentication enables ORDS to authenticate with OCI without the need to store credentials. When ORDS runs on an OCI compute instance and is started using the following command:

```
ords serve --ocid <CONFIG_OCID>
```

Then, it uses the instance principal of the compute instance for authentication by default.

Using this method:

- The compute instance is part of a dynamic group.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- IAM policies grant the dynamic group access to:
  - Database Tools Configuration resources
  - Database Tools Connections
- ORDS automatically retrieves short-lived credentials from OCI
- No OCI configuration files or API keys need to be stored on disk

Advantages:

- Eliminates the need for credential management
- Supports automatic credential rotation
- Provides a strong security posture aligned with OCI best practices
- Recommended for production deployments

Instance Principal authentication is fully managed by the IAM policies assigned to the dynamic group associated with the compute instance.

> **See Also**
>
> Calling Services from an Instance

OCI Configuration Profile Authentication

To enable this method, start ORDS with the following JVM system property:

```
-Doci.profile=<PROFILE_NAME>
```

Using this method:

- ORDS reads credentials from the OCI configuration file (typically `~/.oci/config`)
- Authentication is performed using the specified OCI user, tenancy, and API key
- Access to resources is managed by the IAM policies associated with the OCI user

Advantages:

- Suitable for development and testing environments
- Enables administrators to use existing OCI user credentials
- Does not require ORDS to run on OCI Compute

This approach is generally not recommended for production deployments due to the overhead of managing credentials.

> **See Also**
>
> SDK and CLI Configuration File

Copyright © 2011, 2026, Oracle and/or its affiliates.

4.2.10 Deploying ORDS with the OCI Database API Gateway Configuration service

This section provides the steps to deploy ORDS using the OCI Database API Gateway Configuration service.

Prerequisites

This section describes the prerequisites required for using the OCI Database API Gateway Configuration Service with ORDS.

Before using the OCI Database API Gateway Configuration Service with ORDS, confirm the prerequisites are met:

- An OCI tenancy is available with access to the Database Tools service
- A dedicated compartment exists for Database Tools resources
- A Vault and Master Encryption Key have been created in the appropriate compartment
- A target Oracle Database is available
  - For DATABASE-based configurations, ORDS is already installed in the target database.
    - The target database contains the required ORDS metadata for the REST services that you want to expose
- An ORDS runtime environment is available with a supported Java version installed, such as:
  - An OCI Compute Instance
  - A locally deployed ORDS instance
  - An ORDS Docker container
- An authentication method has been chosen:
  - Instance Principal Authentication (that is, ORDS running on OCI Compute instances)
  - OCI CLI user profile configuration (that is, ORDS running outside OCI or for local development and testing)

See Also

- SDK and CLI Configuration File
- About Database API Gateway Configurations
- IAM Policies Overview
- About Creating Policies
- About Policy Verbs

Steps to deploy ORDS using the OCI Database API Gateway Configuration service

To deploy ORDS using the OCI Database API Gateway Configuration service, perform the following steps:

Copyright © 2011, 2026, Oracle and/or its affiliates.

1. Use an existing or create a new compartment for Database Tools resources.
   a. Alternatively, you can place these resources in the `Root` compartment (that is, at the tenancy level).
2. Use an existing or create a new IAM Domain.
   a. If you are authenticating through Instance Principal (ORDS in an OCI Compute Instance), then choose an IAM Domain that supports Dynamic Groups.

> **See Also**
>
> Feature Availability for Identity Domain Types

   b. Standard Groups are suitable for OCI Profile authentication (IAM users).
3. Configure Matching Rules for your Dynamic Group. For Instance Principal authentication, use the following Dynamic Group Policy Matching Rule for this option:
   `ANY|ALL {instance.id = 'Compute Instance OCID'}`

> **See Also**
>
> Writing Matching Rules to Define Dynamic Groups

4. Create IAM policies to grant your groups access to Oracle Cloud Infrastructure (OCI) services. Dynamic Groups are located in the Identity and Domains service within your OCI domain.
   a. Dynamic Groups (ORDS in an OCI Compute Instance)

Policy	Description
`Allow dynamic-group <dynamic_group_name> to use database-tools-connections in <compartment_name>`	Allows OCI resources in the dynamic group to use Database Tools connections in the named compartment.
`Allow dynamic-group <dynamic_group_name> to read secret-family in <compartment_name>`	Allows the Dynamic Group to read secrets needed for database credentials.
`Allow dynamic-group <dynamic_group_name> to use database-tools-database-api-gateway-configs in <compartment_name>`	Allows the Dynamic Group to use Database Tools Database API Gateway configurations.
`Allow dynamic-group <dynamic_group_name> to read database-tools-family in <compartment_name>`	Allows the Dynamic Group to read Database Tools private endpoints, and endpoint services.

> **See Also**
>
> Writing Policies for Dynamic Groups

   b. Standard Groups (ORDS outside of OCI). Standard Groups are located in User Management within your Domain, in Identity and Domain services:

Copyright © 2011, 2026, Oracle and/or its affiliates.

Policy	Description
`Allow group <group_name> to use database-tools-connections in <compartment_name>`	Allows OCI resources in the dynamic group to use Database Tools connections in the named compartment.
`Allow group <group_name> to read secret-family in <compartment_name>`	Allows the Dynamic Group to read secrets needed for database credentials.
`Allow group <group_name> to use database-tools-database-api-gateway-configs in <compartment_name>`	Allows the Dynamic Group to use Database Tools Database API Gateway configurations.
`Allow group <group_name> to read database-tools-family in <compartment_name>`	Allows the Dynamic Group to read Database Tools private endpoints, and endpoint services.

c. Special considerations:

i. If you are using Bastion Service with Database Connections, ensure the following policy is configured:

Policy	Description
`Allow dynamic-group <group_name> to read bastion-family in <compartment_name>`	Allows a Group to read Bastion resources required when Database Tools connections use private endpoints.
`Allow group <group_name> to read autonomous-databases in compartment <compartment_name>`	To retrieve the wallet from Autonomous Database.

See Also: Bastion IAM Policies

5. Use an existing or create a new database.

> **See Also**
>
> - Overview of Creating a DB System
> - Provision an Autonomous AI Database Instance
> - Creating a Compartment
> - Creating an Identity Domain
> - Identity Domain Types

6. Ensure you have an existing ORDS deployment.

a. If you are creating an OCI Compute instance, ensure that ORDS and a currently supported Java version from the Oracle YUM repository are installed.

i. Optionally, install SQLcl on your Compute instance. SQLcl simplifies connecting to your target database, and its built-in OCI utilities can help troubleshoot OCI connectivity issues.

ii. Additionally, ensure that your Virtual Cloud Network (VCN) and Virtual Network Interface Card (VNIC) are configured with the appropriate routing and security rules.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Launching Your First Linux Instance

> **Note**
>
> For secure, private access from a Virtual Cloud Network (VCN) to your database, create a private endpoint in the Database Tools service. Use this Private Endpoint for your Database API Gateway Configuration.

> **See Also**
>
> - Using Private Endpoints with Database Tools
> - Create a Private Endpoint
> - Database Tools Best Practices

7. Create a Vault and Master Encryption Key for the Compartment.
   - The Vault provides secure storage for the password of the database user. When a new Database Tools Connection is created, a password secret is automatically generated and stored in the Vault.

   > **See Also**
   >
   > – Creating a Vault
   > – Creating a Master Encryption Key

8. Create a Database Tools Connection for the target database.
   a. In the DATABASE:
      i. Optionally, provision a new runtime user (for example: `ORDS_PUBLIC_USER2`):

```
CREATE USER ORDS_PUBLIC_USER<n> IDENTIFIED BY "<password>";
BEGIN
     ORDS_ADMIN.PROVISION_RUNTIME_ROLE(
         p_user => 'ORDS_PUBLIC_USER<n>',
         p_proxy_enabled_schemas => TRUE);
     COMMIT;
END;
/
GRANT CONNECT TO ORDS_PUBLIC_USER<n>;
```

   b. In the CLOUD:
      i. An existing database user should be created prior to creating a new Database Tools Connection.
      ii. Create a Database Tools Connection
      See Also:

      Using the Oracle Cloud Infrastructure Console

Copyright © 2011, 2026, Oracle and/or its affiliates.

c. For either connection type, you can optionally validate the connection from the Database Tools Connection page by selecting Actions > Validate > Validate in the user interface. If the connection is invalid, review the reported error, correct the issue, and then rerun the validation.

   i. If validation fails, resolve the reported issue before continuing. Common causes include:

      - Incorrect database username or password
      - Missing Vault or secret permissions
      - Incorrect database service name
      - Network security list or route table issues
      - Private endpoint configuration issues
      - Wallet or TLS configuration issues

9. Create a Database API Gateway service, choose one of two available deployment methods:

   a. **In the Database**: In this setup, ORDS is already installed in the target database. The ORDS, an additional ORDS Runtime User has been created (i.e. a new ORDS_PUBLIC_USER[n] schema or on an existing schema via the ORDS_ADMIN.PROVISION_RUNTIME_ROLE procedure), and ORDS metadata reside in the database). Only

      i. Under this deployment only the ORDS global and pool configuration settings are retrieved dynamically. API definitions are still defined in the database.

   b. **In the Cloud**: In this setup, ORDS is not installed in the target database. Instead, all ORDS artifacts are dynamically retrieved, including global and pool configuration settings, ORDS metadata, Auto-REST definitions, and user-created ORDS APIs. When choosing which APIs to enable, verify that the target objects are owned by the same schema associated with the Database Tools Connection user.

> **Note**
>
> Refer to sample OpenAPI Specification Document for an example OpenAPI specification that conforms to the OCI Database API Gateway API specifications.

10. Start ORDS using your Database API Configuration.

    a. For Instance Principal authentication (with ORDS deployed on an OCI Compute instance), navigate to the Database API Connection dashboard. From the Actions menu, select View ORDS Command Line to display and copy the required ORDS CLI command. From your OCI compute Instance, start the ORDS service using the following command:
    ```
    ords serve --ocid <CONFIG_OCID>
    ```

    b. If you are using OCI Profile authentication (IAM user), with ORDS deployed locally, external to OCI, or for instance a Docker container, start ORDS using the following command:
    ```
    ords --java-options "-Doci.profile=DEFAULT" serve --OCID <CONFIG_OCID>
    ```


Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> This example assumes an OCI CLI profile exists and the DEFAULT profile has been used.

> **See Also**
>
> Configuring the OCI CLI and profiles

> **Note**
>
> At startup, ORDS dynamically retrieves its configuration from the OCI Database API Gateway service and applies updates at runtime without requiring a restart

4.3 Deploying ORDS with Central Configuration Server

This section describes how to deploy ORDS using a central configuration server.

Overview

By default, Oracle REST Data Services (ORDS) configuration resides locally in files such as the `global/settings.xml` and `databases/default/pool.xml` within an ORDS configuration directory. Alternatively, ORDS can acquire its configuration from a REST API called central configuration server API .

The following are the benefits of using a central configuration server API:

- Centrally managed configuration provides a single place to update, manage, and secure ORDS configuration.
- ORDS pools are dynamically loaded as required, this helps in quick startup time for ORDS.

The Central Configuration Server OpenAPI specification defines the following two REST end points:

- Global Configuration Resource
- Pool Configuration Resource

Typically, a central configuration server is required only when you are managing multiple ORDS nodes or Database Pool configurations.

Prerequisites

- Secure REST API that implements the specified central configuration server API
- ORDS global and database pool information is defined in the OpenAPI specification
- OAuth2.0 authorization server (for issuing access tokens)
- Ability to create an Oracle wallet to store the credentials for accessing the secure central configuration

- OpenAPI Specification
- Database Mapping

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Global Configuration Settings
- Database Pool Configuration Settings
- Security
- Starting ORDS in a Central Configuration Deployment

> **See Also**
>
> ORDS Central Configuration OpenAPI

4.3.1 OpenAPI Specification

Refer to ORDS Central Configuration OpenAPI

4.3.2 Database Mapping

When ORDS is deployed using central configuration, only the global configuration settings are initially retrieved. Appropriate pool configuration is retrieved and cached in the subsequent requests only after a request is made to ORDS for which it requires a Database connection. The global configuration resource provides a `search` HREF template for the pool configuration resource. ORDS uses this template to request the pool configuration. The `search` HREF is templated, meaning the `{host}` string is replaced by a value that identifies the specific pool.

You can use one of the following three methods to replace the `{host}` string to specify the appropriate pool configuration:

- Request Host Method
- Header Method
- Direct URL Specification
- Request Host Method
- Header Method
- Direct URL Specification

4.3.2.1 Request Host Method

Request host method is the default method to replace the `{host}` string in the `search` href. The host of the ORDS request is used.

This method is useful when the sub domain of the ORDS service can be used to identify which pool configuration to use.

Examples

Global Configuration: The href templated search value provided by the Global configuration

```
href="https://central-config.example.com:8585/central/v1/config/pool/{host}"
```

Request to ORDS:

```
GET: https://mydatabase.servername.com/ords/hr/employees/
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Where, `mydatabase.servername.com` is the request host.
ORDS requests the pool configuration from:

```
https://central-config.example.com:8585/central/v1/config/pool/
mydatabase.servername.com
```

4.3.2.2 Header Method

If the sub domain cannot be used to denote the appropriate pool, then a request header can be used. Using the `security.externalHostMappingHeader` setting, the Global configuration specifies the header you want to use.

Examples

Global Configuration:

The href templated search value provided by the Global configuration:

```
href="https://central-config.example.com:8585/central/v1/config/pool/{host}"
```

Request to ORDS:

```
GET: -Header "poolname: mydatabase" https://www.servername.com/ords/hr/
employees/
```

ORDS requests the pool configuration from:

```
https://central-config.example.com:8585/central/v1/config/pool/mydatabase
```

4.3.2.3 Direct URL Specification

Direct URL Specification method is an alternative to the Header Method and aligns with how pool identifiers are retained for file-based configurations. With this method, the pool identifier is specified directly in the request URL. To enable this feature, you must define a new setting in the global configuration `security.externalMappingPathPrefix` and set its value to true.

Global Configuration

The `href` templated search value provided by the global configuration:

```
href="https://central-config.example.com:8585/central/v1/config/pool/{host}"
```

Request to ORDS with a direct URL specification

```
GET: https://www.servername.com/ords/mypool/hr/employees/
```

In this case, `mypool` is the pool identifier included in the request path.

ORDS requests the pool configuration using the following URL:

```
https://central-config.example.com:8585/central/v1/config/pool/mypool
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

4.3.3 Global Configuration Settings

In an ORDS central configuration deployment, global configuration settings must meet the following requirements:

- Must be in a JSON format
- The global settings must be defined in the `settings` object.
- Include a related `Search` link ending with the `{host}` template parameter to describe the Pool configuration resource.

> **See Also**
>
> About the Oracle REST Data Services Configuration Files

Example response for the ORDS global settings

```
{
  "settings": {
    "restEnabledSql.active": true,
    "feature.sdw": true,
    "security.externalHostMappingHeader": "poolname"
  },
  "links": [
    [
      {
        "rel": "collection",
        "href": "https://central-config.example.com:8585/central/v1/config/"
      },
      {
        "rel": "self",
        "href": "https://central-config.example.com:8585/central/v1/config/"
      },
      {
        "rel": "search",
        "href": "https://central-config.example.com:8585/central/v1/config/
pool/{host}",
        "templated": true
      }
    ]
  ]
}
```

4.3.4 Database Pool Configuration Settings

In an ORDS central configuration deployment, pool configuration settings must meet the following requirements:

- Must be in a JSON format
- The pool settings defined in the `database.pool.settings` object.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- When you are configuring a database pool for the Oracle Autonomous Database, pool configuration files must include the contents of its `db.wallet.zip` configuration property encoded in Base64

> **See Also**
>
> About the Oracle REST Data Services Configuration Files

Example 4-1 `db.wallet.zip`

Example pool configuration response with the `db.wallet.zip` file providing a base64 representation of an ADB-S instance wallet.

> **Note**
>
> The `db.username` and `db.password` can be provided as settings. However, it is a best practice to include the credentials in the ADB-S instance wallet before converting it to base64.

> **See Also**
>
> Adding Credentials to an Instance Wallet

```
{
  "database": {
    "pool": {
      "name": "mydatabase",
      "settings": {
        "db.wallet.zip.service":"mydb_low",
        "db.wallet.zip":
"UEsDBBQACAgIAOMid1IAAAAAAAAAAAAAAAALAAAAY3dhbGxldC5zc28BdRuH4TjA...",
        "feature.sdw": true,
        "plsql.gateway.mode": "proxied",
        "restEnabledSql.active": true
      }
    }
  }
}
```

Example 4-2 `db.customURL`

Example pool configuration response with the `db.customURL` providing the pool connection details.

```
{
  "database": {
    "pool": {
      "name": "mydatabase",
      "settings": {
        "db.username" : "ORDS_PUBLIC_USER",
        "db.password" : "ORDS PUBLIC USER PASSWORD",
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
        "db.connectionType" : "customurl",
        "db.customURL" : "jdbc:oracle:thin:@(description=(retry_count=3)
(address_list=(load_balance=on)
         (address=(protocol=tcp)(host=adb.ap-sydney-1.oraclecloud.com)
(port=1521)))

(connect_data=(service_name=database_pool_1_high.adb.oraclecloud.com)))",
        "feature.sdw": true,
        "plsql.gateway.mode": "proxied",
        "restEnabledSql.active": true
      }
    }
  }
}
```

- Using eTag
- Adding Credentials to an Instance Wallet
  This section describes how to add credentials to an instance wallet.

4.3.4.1 Using eTag

Pool configuration responses should include an `eTag`. To identify any changes to a pool configuration, ORDS uses the eTag. If an `eTag` remains unchanged, then ORDS does not refresh the configuration settings of the pool.

4.3.4.2 Adding Credentials to an Instance Wallet

This section describes how to add credentials to an instance wallet.

Adding the database credentials for an ORDS runtime user can be provided using the `db.username` and `db.password` settings. However, a better method is to include the credentials in the the ADB-S instance wallet and then provide that in base64 format using the `db.wallet.zip` setting.

The following command can be used to include the ORDS runtime user credentials. When prompted, provide the instance wallet password.

```
mkstore -wrl <path to the instance wallet> -createCredential <servicename>
<ORDS Runtime user name>
      <ORDS Runtime user password>
```

4.3.5 Security

The Central Configuration API can be secured using basic authentication or OAuth2. The credentials and other information required to acquire an access token or make basic authentication requests are provided using an ORDS credentials wallet.

ORDS supports the following OAuth2.0 grant types:

- Resource Owner Password Credentials Grant
- Client Credentials Grant

Credentials are stored as a secret in the Oracle wallet. The `mkstore` utility can be used to create the ORDS credentials wallet.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> SQLcl provides a `mkstore` command which can be used to create the appropriate ORDS credentials wallet or modify an instance ADB-S instance wallet.

Table 4-5 ORDS Credential Wallet Storage

Secret Alias	Secret Value	Description
`oauth:client_credentials_mode:[Server Name]`	Value must be one of the following: • `basic` • `request_body`	Determines how the client credentials are added to the token request. Default value is `basic`
`oauth:client_id:[Server Name]`	`client_id` of the client application	Used with `grant_type=client_credentials`
`oauth:client_secret:[Server Name]`	Client secret of the client application	Used where `grant_type=client_credentials`
`oauth:grant_type:[Server Name]`	When used, must be one of the following: • `client_credentials` • `password`	The value defaults to `password` when `oauth:username:[hostname]` is included. Otherwise, the default value is `client_credentials`
`oauth:password:[Server Name]`	`password` of the user who needs to be authenticated	Used with `grant_type=password`
`oauth:scope:[Server Name]`	Conforms to Access Scope specification as defined in Access Token Scope	OAuth 2.0 scopes to the request access
`oauth:token_endpoint:[Server Name]`	Absolute URL of an access token endpoint as defined in Token Endpoint	ORDS client uses the `POST` method when accessing the token requests
`oauth:username:[Server Name]`	`username` of the user who needs to be authenticated	Used when `grant_type=password`

- Demonstration to Create an Entry of a Wallet for OAuth2.0 Authentication

4.3.5.1 Demonstration to Create an Entry of a Wallet for OAuth2.0 Authentication

The example in this section demonstrates how to set up a wallet with the `mkstore` utility for an OAuth2.0 client credentials grant type.

Setting up a wallet with the SQLcl mkstore utility

```
sql /nolog
SQL> mkstore -wrl /tmp/wallet -create
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Creating an entry of a wallet for OAuth2.0 Authentication

The Global configuration resource server in this example is `central-config.example.com`.

```
sql /nolog
SQL> mkstore -wrl /tmp/wallet -createEntry oauth:grant_type:central-
config.example.com client_credentials
SQL> mkstore -wrl /tmp/wallet -createEntry oauth:token_endpoint:central-
config.example.com https://myauthserver.com/oauth/token
SQL> mkstore -wrl /tmp/wallet -createEntry oauth:client_id:central-
config.example.com SeqlyQGW4iKgHd@cQ4Xnkg..
SQL> mkstore -wrl /tmp/wallet -createEntry oauth:client_secret:central-
config.example.com SuBNKFsft£t924Dsfgsh..
```

4.3.6 Starting ORDS in a Central Configuration Deployment

ORDS recognizes that a central configuration deployment is required when the following Java options are included in the `ords serve` command.

```
bin/ords --java-options "-Dconfig.url=[ORDS Global Configuration Resource
Location]
-Dconfig.wallet=[ORDS Credentials Wallet Location]" serve
```

Java Options

- `-Dconfig.url`: Provides ORDS with the Global configuration resource and its server is used to reference secrets in the ORDS credentials wallet.
  Example: `-Dconfig.url=https://central-config.example.com:8585/central/v1/config/`
- `-Dconfig.wallet`: Provides ORDS with the ORDS credentials wallet location. Through auto login, it uses the secrets contained in the Oracle wallet to acquire an access token so that the Global and Pool resources can be accessed.
  Example: `-Dconfig.wallet=/tmp/wallet`

You will receive a notification in the command prompt from where the `ords serve` command was issued, indicating that ORDS is running under a central configuration deployment. ORDS then waits for the incoming HTTPS requests for the database pools that have been configured.

- Satisfying the HTTPS Requests

4.3.6.1 Satisfying the HTTPS Requests

ORDS waits for the incoming HTTPS requests for currently configured database pools.

Once the incoming requests arrive, ORDS first determines if the database pool configuration information is cached. If the configuration settings are not available, then ORDS makes a request to the central configuration server to retrieve the pool configuration. The pool is then initialized and the request is handled.

Copyright © 2011, 2026, Oracle and/or its affiliates.

4.4 Deploying ORDS with ETCD

This section describes how to deploy ORDS using ETCD.

Overview

ORDS can retrieve its configuration data from ETCD. The following are the benefits of using ETCD with ORDS:

- You don't need to restart ORDS when you make changes to the configuration settings
- Centralizing data in an ETCD server simplifies the configuration management
- Leverages the robust security features of ETCD
- Distributed architecture of ETCD ensures reliability and scalability

Significance of using ETCD

ETCD is a lightweight, distributed, and highly available key-value store that is widely used in cloud-native applications. ETCD fits seamlessly into ORDS deployments and offers flexibility for the users who want to manage dynamic configurations.

> **Note**
>
> ORDS currently supports ETCD version v3.5 or later.

- Prerequisites
- Database mapping
- Configuration Settings
- Starting ORDS with ETCD

4.4.1 Prerequisites

The following are the prerequisites for using ETCD as a configuration source for ORDS:

- Install ETCD on your target server(s). Confirm installation using the following code:

  ```
  etcdctl version
  ```

  Example output:

  ```
  etcdctl version: 3.5.17
  API version: 3.5
  ```

- Configure authentication and authorization. Refer to the ETCD documentation.
- Ensure that you can create an Oracle wallet to store the credentials for accessing ETCD (optional but recommended).

Copyright © 2011, 2026, Oracle and/or its affiliates.

4.4.2 Database mapping

The database mapping mechanism in this configuration, mirrors the approach used in the remote central configuration API. It provides dynamic and efficient handling of multiple database instances.

> **See Also**
>
> Database Mapping

Adding Credentials to an Instance Wallet

Oracle highly recommends to include the database credentials in the ADB-S instance wallet and then provide that in base64 format using the `db.wallet.zip` setting.

> **See Also**
>
> About the Oracle REST Data Services Configuration Files

4.4.3 Configuration Settings

Similar to the remote central configuration, global and pool configuration settings must meet the following requirements:

- Must be in a JSON format
- The global settings must be defined in the `settings` object.
- The pool settings must be defined in the `database.pool.settings` object.

Global Configuration Settings

The global configuration settings are updated to support ETCD integration.

Example 4-3 Response for the ORDS Global Settings

```
{
  "settings": {
    "restEnabledSql.active": true,
    "feature.sdw": true
  }
}
```

This simplifies the configuration and aligns with the design principles of ETCD.

Database Pool Configuration Settings

`db.wallet.zip`
Example pool configuration response with the `db.wallet.zip` file provides a base64 representation of an ADB-S instance wallet:

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> The `db.username` and `db.password` can be provided as settings. However, it is a best practice to include the credentials in the ADB-S instance wallet before converting it to base64.

```
{
  "database": {
    "pool": {
      "name": "mydatabase",
      "settings": {
        "db.wallet.zip.service":"mydb_low",
        "db.wallet.zip":
"UEsDBBQACAgIAOMid1IAAAAAAAAAAAAAAAALAAAAY3dhbGxldC5zc28BdRuH4TjA...",
        "feature.sdw": true,
        "plsql.gateway.mode": "proxied",
        "restEnabledSql.active": true
      }
    }
  }
}
```

> **See Also**
>
> Adding Credentials to an Instance Wallet

`db.customURL`
Example pool configuration response with the `db.customURL` providing the pool connection details:

```
{
  "database": {
    "pool": {
      "name": "mydatabase",
      "settings": {
        "db.username" : "ORDS_PUBLIC_USER",
        "db.password" : "ORDS PUBLIC USER PASSWORD",
        "db.connectionType" : "customurl",
        "db.customURL" : "jdbc:oracle:thin:@(description=(retry_count=3)
(address_list=(load_balance=on)
         (address=(protocol=tcp)(host=adb.ap-sydney-1.oraclecloud.com)
(port=1521)))

(connect_data=(service_name=database_pool_1_high.adb.oraclecloud.com)))",
        "feature.sdw": true,
        "plsql.gateway.mode": "proxied",
        "restEnabledSql.active": true
      }
    }
  }
}
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

4.4.4 Starting ORDS with ETCD

ORDS recognizes that an ETCD deployment is required when the following Java options are included in the `ords serve` command:

```
export ETCD_ORDS_USERNAME=etcd-username
export ETCD_ORDS_PASSWORD=etcd-password
bin/ords --java-options "-Dconfig.url=[ETCD Location] -Dconfig.source=etcd"
serve
```

Java Options

- `-Dconfig.url`: Provides ORDS with the ETCD IP address and the port number and its server.
  Example: `-Dconfig.url=https://etcd.example.com:2379`
- `-Dconfig.source`: Specifies the source of the configuration data. If the target of the URI is an ETCD server, then it must be set to `etcd`

You will receive a notification in the command prompt from where the `ords serve` command is issued indicating that ORDS is running with ETCD configuration. ORDS then waits for the incoming HTTPS requests for the database pools that have been configured.

Using ORDS Credential Wallet Storage

Alternatively, use a wallet for secure storage of credentials.

Setting up a wallet with the SQLcl mkstore utility

```
sql /nolog
SQL> mkstore -wrl /tmp/wallet -create
```

Creating an entry of a wallet for basic Authentication

The ETCD server in this example is `etcd.example.com`.

> **Note**
>
> `<etcd-username>:<etcd-password>` must be in base64.

```
sql /nolog
SQL> mkstore -wrl /tmp/wallet -createEntry http:basic:etcd.example.com <etcd-
username>:<etcd-password>
```

Start ORDS with the appropriate Java options:

```
bin/ords --java-options "-Dconfig.url=[ETCD Location]
-Dconfig.wallet=[ORDS-ETCD Credentials Wallet Location]" serve
```

Where the Java option `-Dconfig.wallet`, provides ORDS with the ETCD-ORDS credentials wallet location. Through auto login, it uses the secrets contained in the Oracle wallet to get the basic credentials so that the Global and Pool resources can be accessed.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example: `-Dconfig.wallet=/tmp/wallet`

Using Prefixing (Recommended)

Using prefixing method is recommended enhance the configuration organization:

```
export ETCD_ORDS_PREFIX=ords-conf-dev
bin/ords --java-options "-Dconfig.url=[ETCD Location] -Dconfig.source=etcd"
serve
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

5

Overview of Schema Level Configuration Override

This section describes schema level configuration override feature.

The schema level configuration override feature provides a mechanism to override configuration properties for a REST enabled schema. This allows authorized database users to customize Oracle REST Data Services (ORDS) on the fly. For example, a database user can use this mechanism to enable or disable an optional feature. Only a predefined subset of database schema level configurations can be overridden, global configurations cannot be overridden.

- Features of Schema Level Configuration Override
  This section lists the features of schema level configuration override.
- Overridable Configuration Properties
  This section lists the overidable configuration properties.
- Use Cases
  This section shows some use cases for using the schema level configuration override feature.
- Scope and Visibility Matrix
  This section describes how to configure the visibility and the scope for the database and schema users.

5.1 Features of Schema Level Configuration Override

This section lists the features of schema level configuration override.

The features of schema level configuration override are as follows:

- Database users can override the configuration property for their schema with the ords PL/SQL API.
- Database administrator can override the configuration property for any schema in the database with our ords_admin PL/SQL API.
- The configuration value is cached for 10 mins. You must wait for 10 minutes before the new value becomes effective.

> ⓘ **See Also**
>
> - ORDS PL/SQL Package Reference
> - Oracle REST Data Services Administration PL/SQL Package Reference

5.2 Overridable Configuration Properties

This section lists the overidable configuration properties.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 5-1 Overridable Configuration Properties

Key	Description	Scope	Setting Override Value
`restEnabledSql.active`	Specifies whether the REST-Enabled SQL service is active. REST- Enabled SQL is a prerequisite for SDW or Database Actions. SDW is disabled if the value is set to `false`.	Schema	`begin ords.set_property( p_key => 'restEnabledSql.active', p_value => 'false'); commit; end; /`
`feature.sdw`	Specifies whether the Database Actions feature is active.	Schema	`begin ords_admin.set_property(p_key => 'feature.sdw', p_value => 'false'); commit; end; /`
`debug.printDebugToScreen`	Specifies whether the error responses include stack trace.	Schema	`begin ords_admin.set_property(p_key => 'debug.printDebugToScreen', p_value => 'true'); commit; end; /`
`debug.printOWADebug`	Specifies whether the error responses for PL/SQL gateway requests include stack trace. Overrides `debug.printDebugToScreen` for PL/SQL gateway requests.	Database	`begin ords_admin.set_property( p_key => 'debug.printOWADebug', p_value => 'true'); commit; end; /`

5.3 Use Cases

This section shows some use cases for using the schema level configuration override feature.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 5-2 Use Cases of Schema Level Configuration Override feature

Scenario	Solution

An Autonomous Database user wants to debug their ORDS REST Services.	The database user sets the property to `true` for their schema as shown in the following code snippet:
	
	`ords.set_property(`
	`  p_key =>`
	`'debug.printDebugToScreen',`
	`  p_value  => 'true'`
	`);`

Administrator wants a REST-Enabled schema but not have a REST-Enabled SQL.	The administrator sets the property to `true` for the relevant schema as shown in the following code snippet:
	
	`ords_admin.set_property(`
	`  p_schema => 'REST_PRODUCTION',`
	`  p_key => 'restEnabledSql.active',`
	`  p_value  => 'true'`
	`);`

Administrator wants to completely disable SDW for a database.	The administrator sets the property for every schema as shown in the following code snippet:
	
	`ords_admin.set_property(p_schema =>`
	`'s1', p_key =>`
	`'feature.sdw', p_value  => 'false');`
	
	`ords_admin.set_property(p_schema =>`
	`'s2', p_key =>`
	`'feature.sdw', p_value  => 'false');`

REST-Enabled schema wants to clear a property value that was previously set.	The database user runs the following command:
	
	`ords.unset_property(p_key =>`
	`'debug.printDebugToScreen');`

5.4 Scope and Visibility Matrix

This section describes how to configure the visibility and the scope for the database and schema users.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 5-3 Scope and Visibility Matrix

User Type	Database	Schema
Public	• Only administrator can set a value that applies to all the schemas • USER_ORDS_PROPERTIES view shows the value, if it is set • Unprotected REST endpoint shows the value if it is set • Currently used for login message and SDW header and footer	• Administrator and non-administrator user can set a value per schema • There is no default value that automatically applies to a schema • USER_ORDS_PROPERTIES view shows the value if it is set • Unprotected REST endpoint shows the value if it is set
PROTECTED	• Only the administrator can set a value that applies for all the schemas • USER_ORDS_PROPERTIES view shows the value if it is set • Unprotected REST endpoint does not show value if it is set	• Administrator and non-administrator user can set a value per schema • There is no default value that automatically applies to a schema • USER_ORDS_PROPERTIES view shows the value if it is set • Unprotected REST endpoint does not show the value if it is set
PRIVATE	• Only administrator can set the value that applies for all the schemas • USER_ORDS_PROPERTIES view does not show a value if it is set • Unprotected REST endpoint does not show a value if it is set	• Only the administrator can set a value per schema • There is no default value that automatically applies to a schema • USER_ORDS_PROPERTIES view does not show the value if it is set • Unprotected REST endpoint does not show the value if it is set

Copyright © 2011, 2026, Oracle and/or its affiliates.

6

Miscellaneous Configuration Options of Oracle REST Data Services

This section explains how to configure Oracle REST Data Services for connecting to multiple databases for routing requests, and it refers to other documentation sources for other configuration information.

> **Note**
>
> You must restart Oracle REST Data Services after making any configuration changes. For ensuring High Availability, Oracle recommends that you use a load balancer in front of multiple ORDS instances, so that rolling restarts can be achieved.

- Using the REST Enabled SQL Service with MySQL Database
- Configuring Jetty in ORDS Standalone Mode
  This section describes how to configure the Eclipse Jetty Server used in ORDS standalone mode.
- Support for Oracle RAC Fast Connection Failover
  Oracle REST Data Services support the Fast Connection Failover (FCF) feature of Oracle Real Application Clusters (Oracle RAC).
- Configuring ORDS with Kerberos Setup
  This section explains how ORDS can be configured to reference a Kerberos file-based ticket cache and make a connection to an Oracle Database Kerberos authenticated user with the ORDS Runtime Privilege.
- Authorizing Oracle REST Data Services to Access Oracle Data Guard Protected Users
- Configuring REST-Enabled SQL Service Settings
  This section explains how to configure the REST- Enabled SQL service.
- Configuring ICAP Server Integration for Virus Scan
  This section explains how to configure ORDS to integrate with ICAP server for virus scan.
- Configuring the Custom Error Pages
  This section explains how to configure a custom error page instead of the error page generated by Oracle REST Data Services.
- Managing ORDS Administrator Privilege
- Managing ORDS Runtime Privilege
- Using OAuth2 in Non-HTTPS Environments
- ORDS Performance Considerations
  This section lists the key ORDS performance factors that are useful in achieving high performance.
- Configuring ORDS to Connect with OCI Database Tools Connections
  You can configure Oracle REST Data Services (ORDS) to securely connect to your Oracle Database in Oracle Cloud Infrastructure (OCI) by leveraging OCI database tools connections.

Copyright © 2011, 2026, Oracle and/or its affiliates.

6.1 Using the REST Enabled SQL Service with MySQL Database

This section describes the ORDS feature which is only supported with MySQL database running on Oracle Cloud Infrastructure.
You can setup your own ORDS instance to use the REST Enabled SQL Service with a MySQL database over JDBC. The connection details are specified similar to how you specify for other ORDS connection pool. For MySQL JDBC connections, the `db.connectionType` is always `customurl`. The `db.customURL` property must be a valid JDBC connection string for the database. The host machine running the database must be accessible to the host machine with ORDS instance running. The MySQL database accounts to be used must be configured to permit logins from the host machine with ORDS instance running. ORDS supports connecting to Oracle hosted MySQL database servers such as the MySQL Data Service or an Oracle Compute Instance running MySQL server.

- Understanding the Database Credentials Source Setting
- Configuring a Pool for MySQL Database
  Pool configuration is required to use ORDS with MySQL database. You can configure the pool through the ORDS command line interface.

6.1.1 Understanding the Database Credentials Source Setting

For each request received, ORDS creates a JDBC connection for executing the SQL statements in the request. To create the JDBC connection, ORDS can be configured to use the pool connection details, or uses the basic authorization credentials in the request. The credentials are specified using the `db.credentials` source configuration property. Possible values are `POOL` (default value) or `REQUEST`.

> **Note**
>
> To access the REST Enabled SQL Service, a client must have the ORDS SQL developer role.

If you are using the default value as `POOL`, then the credentials in the pool configuration are used for processing any SQL statements in the request. However, the client must provide credentials for an identity management system to authorize them and assign them the SQL developer role. Only then the client can access to the REST Enabled SQL service.

If the value is set to `REQUEST`, then the username and password specified in the pool configuration are still required. However, these credentials are only used to verify the connection details in the pool, when the pool is used for the first time. The username and password in the basic authorization header are used to make a new JDBC connection with the target database. If a connection is established, then the client is considered to have the SQL developer role. This authorizes them to invoke the REST Enabled SQL service. The new JDBC connection is used during the request lifecycle and then closed.

6.1.2 Configuring a Pool for MySQL Database

Pool configuration is required to use ORDS with MySQL database. You can configure the pool through the ORDS command line interface.

You must configure ORDS to be able to use a MySQL database with Oracle REST Data Services (ORDS) running in a customer managed environment. Depending on where you

Copyright © 2011, 2026, Oracle and/or its affiliates.

install Oracle REST Data Services for your customer managed environment, do one of the following:

- If your customer managed environment for Oracle REST Data Services runs in Oracle Cloud Infrastructure, then use an Oracle YUM repository and perform a YUM install of ORDS.
- If your customer managed environment for Oracle REST Data Services runs in some other environment, then download ORDS from the Oracle REST Data Services Download page.

To use ORDS with MySQL database, you do not need any installation to the database, you only need to do a pool configuration. You can configure the pool through the ORDS command line interface.

To configure the pool for MySQL database, perform the following steps:

> **Note**
>
> The credentials in a request are used to execute the SQL statements. The `db.username` specified in the MySQL database is the user who has all the permissions to create a connection and is used to verify the overall pool configuration details.

```
ords config --db-pool mysql set db.connectionType customurl
ords config --db-pool mysql set db.customURL "jdbc:mysql://10.0.1.23/?
sslMode=REQUIRED"
ords config --db-pool mysql set db.username
user_only_has_permission_to_connect_and_nothing_more
ords config --db-pool mysql set db.credentialsSource request
ords config --db-pool mysql set restEnabledSql.active true
ords config --db-pool mysql secret db.password
```

In the preceding example,

- Any relevant properties for the JDBC driver can be specified in the `db.customURL` property. In the preceding example, `db.customURL` value, `sslMode` is set to `REQUIRED`, rather than the default value `PREFERRED`, to ensure a secure connection between ORDS and the MySQL server.
- The database pool is called `mysql`. However, the pool can have any name. The default pool can be configured as a MySQL connection pool. You can have multiple pools defined for as many MySQL databases you wish to use.
- The `db.username` specified is a MySQL database user who has sufficient permissions to create a connection. This database account is used to verify the overall pool configuration details.
- Configuring ORDS for Supported Containers

6.1.2.1 Configuring ORDS for Supported Containers

This section describes how to use ORDS with connection pool configuration in a supported container for a MySQL database.

Specifying the Configuration Location

When you are running ORDS in a standalone mode using the `ords serve` command, there are options for specifying the configuration directory location. When deploying the `ords.war` in a

Copyright © 2011, 2026, Oracle and/or its affiliates.

supported container such as Apache Tomcat or WebLogic Server, the configuration directory location must be specified by setting the `config.url` system property. The mechanism for doing this varies for different container products.

- To set the `config.url` system property before starting Apache Tomcat, run the following command:

  ```
  export JAVA_OPTS="-Dconfig.url=/scratch/my_ords_config"
  ```

- To set `config.url` system property before starting WebLogic Server, run the following command:

  ```
  export JAVA_OPTIONS="-Dconfig.url=/scratch/my_ords_config"
  ```

- Alternatively, use the `ords war` command to create a deployable web application archive file which has the `config.url` context parameter explicitly set and any `jar` files from `lib/ext` folder are included.

MySQL JDBC Jar for ORDS

ORDS does not distribute a MySQL JDBC jar. For ORDS to create a JDBC connection to a MySQL database, the relevant JDBC jar must be in the runtime classpath. Use `OCI YUM mysql-connector-java` or download MySQL Connector/J from `https://www.mysql.com/` and then copy the `jar` file to the relevant location for one of the server modes: Standalone, Apache Tomcat, or WebLogic Server.

> **Note**
>
> The minimum required version of MySQL Connector/J is 8.0.27.

The OCI YUM RPM distribution of ORDS creates a symbolic link to the `OCI YUM mysql-connector-java` JDBC jar.

OCI YUM RPM

```
-- Install MySQL Connector/J community edition
sudo yum install mysql-connector-java

-- Confirm JDBC jar is installed
ls -l /usr/share/java/mysql-connector-java.jar

-- Install ORDS from OCI YUM repository
sudo yum install ords

-- Note that ORDS RPM install will create a symbolic link to ORDS
installation lib/ext/ directory
ls -l /opt/oracle/ords/lib/ext/
```

- ORDS Running in Standalone Mode
- ORDS Deployed in Apache Tomcat
- ORDS Deployed in Weblogic Server

Copyright © 2011, 2026, Oracle and/or its affiliates.

6.1.2.1.1 ORDS Running in Standalone Mode

To be in the runtime classpath when running ORDS in a standalone mode, the MySQL JDBC jar must be first added to the Extension folder. The Extension folder is `lib/ext` directory in the ORDS distribution and is created through the OCI YUM RPM installation process that is outlined in the preceding section.

6.1.2.1.2 ORDS Deployed in Apache Tomcat

> **Note**
>
> When you are using Apache Tomcat, to avoid getting `java.sql.SQLException: No suitable driver` errors, it is necessary to explicitly set the JDBC Driver classname in the pool.
>
> To set the JDBC driver classname in the pool, run the following command:
>
> ```
> ords config --db-pool mysql set jdbc.driverName com.mysql.cj.jdbc.Driver
> ```

To be in the runtime classpath when ORDS is deployed in Apache Tomcat, the MySQL JDBC `jar` must be added to the server classpath or in the deployed web application. Adding the `jar` to the server classpath can be achieved in a number of ways, the most common method is adding the `jar` file to the `$CATALINA_HOME/lib` directory.

See Apache Tomcat documentation for more options and guidelines to determine the most appropriate deployment environment.

To include the JDBC `jar` in the deployed web application, ensure that it is in the the `lib/ext/` folder and uses the `ords war` command to create a deployable web application archive file which has the `config.url` context parameter explicitly set and any `jar` files from `lib/ext` folder included.

> **See Also**
>
> Apache Tomcat 8

6.1.2.1.3 ORDS Deployed in Weblogic Server

To be in the runtime classpath when ORDS is deployed in WebLogic Server, the MySQL JDBC `jar`file must be added to the server classpath or in the deployed web application. One approach to add the `jar` file to the server classpath is to specify the `jar` location in the `WEBLOGIC_CLASSPATH` environment variable in the `commEnv.cmd/sh` script.

See WebLogic Server documentation for more options and guidelines to determine the most appropriate deployment environment.

To include the JDBC `jar` in the deployed web application, ensure that it is in the the `lib/ext/` folder and uses the `ords war` command to create a deployable web application archive file which has the `config.url` context parameter explicitly set and any `jar` files from `lib/ext` included.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Understanding WebLogic Server Deployment

6.2 Configuring Jetty in ORDS Standalone Mode

This section describes how to configure the Eclipse Jetty Server used in ORDS standalone mode.

In a standalone mode, ORDS uses an embedded Eclipse Jetty server. The default configuration of Jetty is optimised for the most common ORDS use cases. This Jetty server configuration can be modified using certain ORDS settings as defined in Java system properties and also in the Jetty XML configuration in the following directory: `${configuration.directory}/global/standalone/etc`.

> **Note**
>
> These settings have no effect when ORDS is deployed in a container such as Apache Tomcat or Oracle WebLogic Server.

- Using Java System Properties
  This section describes how to provide a Java system property to modify the settings in the Jetty configuration used by ORDS.
- Java System Properties
  This section lists the Java system properties.
- Using Jetty XML Configuration Files
  This section describes how to configure Jetty server for additional functionality using the Jetty XML configuration files.

> **See Also**
>
> - Eclipse Jetty server
> - Understanding the Configurable Settings

6.2.1 Using Java System Properties

This section describes how to provide a Java system property to modify the settings in the Jetty configuration used by ORDS.

To specify a system property, you must use the `{{--java-options}}` at the command line and provide the Java System Property definitions as a parameter.

Example:

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example 6-1 Using the {{--java-options}} command line option

```
ords --java-options '-Dthreads.max=300 -Djetty.request.header.size=8192' --
config
      /path/to/config/ serve
```

> **See Also**
>
> JDK Java Options

6.2.2 Java System Properties

This section lists the Java system properties.

Table 6-1 Java System Properties

Java System Property Name	Description	Default Value
`threads.min`	Specifies the minimum number of threads in the Jetty Server (`id="Server"`) thread pool.	10
`threads.max`	Specifies the maximum number of threads in the Jetty Server (`id="Server"`) thread pool.	200
`threads.timeout`	Specifies the maximum thread idle time in miliseconds for the Jetty Server (`id="Server"`) thread pool. Threads that are idle for longer than this period may be stopped.	60000
`jetty.send.xpoweredBy`	Specifies the `SendXPoweredBy` setting in the Jetty `HttpConfiguration (id="httpConfig")`	false
`jetty.output.buffer.size`	Specifies the `OutputBufferSize` setting in the Jetty `HttpConfiguration (id="httpConfig")`	32768
`jetty.request.header.size`	Specifies the `RequestHeaderSize` setting in the Jetty `HttpConfiguration (id="httpConfig")`	65536
`jetty.response.header.size`	Set the `ResponseHeaderSize` setting in the Jetty `HttpConfiguration (id="httpConfig")`	8192
`jetty.send.server.version`	specifies the `SendServerVersion` setting in the Jetty `HttpConfiguration (id="httpConfig")`	false

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 6-1 (Cont.) Java System Properties

Java System Property Name	Description	Default Value
`jetty.send.date.header`	Specifies the `SendDateHeader` setting in the Jetty `HttpConfiguration (id="httpConfig")`	false
`jetty.dump.start`	Specifies the `DumpAfterStart` setting in the Jetty Server `(id="Server")`	false
`jetty.dump.stop`	Specifies the `DumpAfterStop` setting in the Jetty Server `(id="Server")`	false

6.2.3 Using Jetty XML Configuration Files

This section describes how to configure Jetty server for additional functionality using the Jetty XML configuration files.

When ORDS is running in a standalone mode, the Eclipse Jetty Home is `${configuration.directory}/global/standalone/`. The Jetty XML syntax can be used to configure the Jetty Server for additional functionality by placing configuration XML files in the Jetty Home `etc` directory. The capability to do this is provided through the Eclipse Jetty server product.

> **See Also**
>
> Eclipse Jetty documentation

Examples

This section provides examples of augmenting Eclipse Jetty Server behaviour for additional functionalily. The following examples show how you can change Eclipse Jetty Server behaviour and not the ORDS behaviour. The location for Jetty XML files to be used by ORDS at the startup in standalone mode is:

```
${configuration.directory}/global/standalone/etc/
```

Example 6-2 Using a specific access log format

When the configuration setting `standalone.access.log` is provided, ORDS can produce an access log.

```
/global/standalone/etc/jetty-access-log.xml
<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN" "http://www.eclipse.org/
jetty/configure.dtd">
<Configure id="Server" class="org.eclipse.jetty.server.Server">
            <Set name="requestLog">
              <New id="RequestLogImpl"
class="org.eclipse.jetty.server.CustomRequestLog">
                <Arg>/ords/ords-access.log</Arg>
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
                <Arg>%{remote}a - %u %t "%r" %s %O "%{Referer}i" "%{User-
Agent}i"</Arg>
              </New>
            </Set>
</Configure>
```

Example 6-3 Always returning a certain header in the response

Although this can also be achieved through a Load Balancer or Reverse Proxy in front of ORDS. If you want a specific header to be returned in every response from the ORDS server. Then use the following sample code snippet:

```
/global/standalone/etc/jetty-response.xml
<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN" "http://www.eclipse.org/
jetty/configure.dtd">
<Configure id="Server" class="org.eclipse.jetty.server.Server">
  <Call name="insertHandler">
  <Arg>
    <New class="org.eclipse.jetty.rewrite.handler.RewriteHandler">
      <Get id="Rewrite" name="ruleContainer" />
      <Call name="addRule">
        <Arg>
          <New id="header"
class="org.eclipse.jetty.rewrite.handler.HeaderPatternRule">
            <Set name="pattern">*</Set>
            <Set name="headerName">Strict-Transport-Security</Set>
            <Set name="headerValue">max-age=31536000;includeSubDomains</Set>
          </New>
        </Arg>
      </Call>
    </New>
  </Arg>
</Call>
</Configure>
```

> **Note**
>
> In Jetty 12.0.x, the `Name` and `Value` methods of the `HeaderPatternRule` class (located in the `org.eclipse.jetty.rewrite.handler` package) have been renamed to `headerName` and `headerValue`. Ensure that your Jetty configuration files reference the appropriate Document Type Definition (DTD), as well as the correct package names, classes, interfaces, or method names for the version of Jetty in use. To confirm which embedded Jetty version your ORDS deployment is running, refer to the relevant ORDS release notes or review your ORDS Standard Output.

6.3 Support for Oracle RAC Fast Connection Failover

Oracle REST Data Services support the Fast Connection Failover (FCF) feature of Oracle Real Application Clusters (Oracle RAC).

Copyright © 2011, 2026, Oracle and/or its affiliates.

Oracle REST Data Services runs with the Universal Connection Pool (UCP) in all the Application Server environments that it supports, such as WebLogic, Tomcat. UCP in turn supports Fast Connection Failover . To enable FCF, Oracle Notification Service (ONS) must to be enabled. To enable ONS, add entries to the list of properties in the Oracle REST Data Services `settings.xml` configuration file as shown in the following code snippet:

```
<entry key="jdbc.enableONS">true</entry>
<entry key= "jdbc.ONSConfig">nodes=racnode1:4200,racnode2:4200\nwalletfile=/
oracle11/onswalletfile</entry>
```

ONS is the messaging facility used to send the Fast Application Notification (FAN) events. When ONS is enabled, Oracle REST Data Services automatically enables FCF. To Enable specific FCF capabilities such as fail over or other advanced FCF capabilities such as load balancing, you need to add entries in the configuration file for the custom connection as shown in the following code snippet:

```
<entry key="db.connectionType">customurl</entry>
<entry key="db.customURL">jdbc:oracle:thin:@(DESCRIPTION=(FAILOVER=ON)
(ADDRESS_LIST=
        (LOAD_BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=prod_scan.example.com)
(PORT=1521)))
        (CONNECT_DATA=(SERVICE_NAME=ISPRD)))</entry>
```

After updating the `defaults.xml` configuration file, Oracle REST Data Services need to be restarted for the changes to take effect.

UCP supports Fast Connection Failover. FCF listens and responds to FAN events to deal with the following two scenarios:

- **Unplanned outages**: When RAC detects an instance failure, it generates a FAN Down event which FCF picks up. FCF then terminates all connections to the failed instance and directs all future requests to the surviving RAC instances.
- **Planned outages**: For instance, when a Database Administrator (DBA) wants to gracefully shut down a RAC instance for performing some maintenance activity. The instance shutdown generates a FAN Planned Down event which FCF picks up. FCF then directs all new requests to other RAC instances and **drains** or allows currently active transactions to complete.

> **Note**
>
> Long running transactions may need to be terminated forcefully.

6.4 Configuring ORDS with Kerberos Setup

This section explains how ORDS can be configured to reference a Kerberos file-based ticket cache and make a connection to an Oracle Database Kerberos authenticated user with the ORDS Runtime Privilege.

To configure ORDS with Kerberos setup, perform the following steps:

1. Create a new user using external authentication
2. Set up the environment variables

Copyright © 2011, 2026, Oracle and/or its affiliates.

3. Provide a valid ticket
4. Add ORDS pool settings

1. **Create a New User using External Authentication**

   Create a new Oracle Database user using external authentication (Kerberos) and provision the user as an ORDS runtime user.

```
CREATE USER ORDS_PUBLIC_KRBUSER IDENTIFIED EXTERNALLY AS
'<kerberos_principal_name>';
GRANT CONNECT TO "ORDS_PUBLIC_KRBUSER";
BEGIN
     ORDS_ADMIN.PROVISION_RUNTIME_ROLE(
         p_user => 'ORDS_PUBLIC_KRBUSER',
         p_proxy_enabled_schemas => TRUE);
END;
/
```

2. **Set up the Environment Variables**

   > **Note**
   >
   > Ensure that you have a Kerberos configuration file `krb5.conf` and a file-based ticket cache

   Set up the following environment variables:

```
export KRB5_CONFIG=<path to krb5.conf>
export KRB5CCNAME=<path to credential cache>
```

3. **Provide a Valid Ticket**

   Provide a valid ticket in the ticket cache to get authenticated when connecting to the Oracle Database.
   `kinit <principal>`

4. **Add ORDS Pool Settings**

   Add the following pool settings to the `pool.xml` file using the ticket in the ticket cache:

```
<entry key="oracle.net.authentication_services">(KERBEROS5)</entry>
<entry key="oracle.net.kerberos5_mutual_authentication">true</entry>
```

   For example, to run ORDS in standalone mode with Kerberos, execute the following command:

```
ords --java-options “-Djava.security.krb5.conf=$KRB5_CONFIG” serve
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

6.5 Authorizing Oracle REST Data Services to Access Oracle Data Guard Protected Users

To access the database schema objects that are protected by an Oracle Data Vault Realm, it is necessary to grant a proxy user authorization to the Oracle REST Data Services Public User.

The following example authorizes the Oracle REST Data Services Public User, `ORDS_PUBLIC_USER` to proxy the database `HR` user:

```
begin
  DBMS_MACADM.AUTHORIZE_PROXY_USER('ORDS_PUBLIC_USER','HR');
end;
/
```

6.6 Configuring REST-Enabled SQL Service Settings

This section explains how to configure the REST- Enabled SQL service.

> **Note**
>
> Enabling the REST- Enabled SQL service enables authentication against the Oracle REST Data Service enabled database schemas. This makes the database schemas accessible over HTTPS, using the database password. Oracle highly recommends that you provide strong secure database passwords

REST- Enabled SQL service is a feature of Oracle REST Data Service. By default, the REST Enabled SQL service is turned off. To enable the REST- Enabled SQL service and the REST-Enabled SQL Export service, perform the following steps:

1. Run the following command:
   ```
   ords --config <configuration_folder> config [--db-pool <pool_name>] set
   restEnabledSql.active true
   ```
2. Restart Oracle REST Data Services.

6.7 Configuring ICAP Server Integration for Virus Scan

This section explains how to configure ORDS to integrate with ICAP server for virus scan.

ORDS PL/SQL gateway supports the offloading of virus scanning responsibilities to an Internet Content Adaptation Protocol (ICAP) compliant virus scan server when the files are uploaded. The hostname and port of the virus scan server is specified in the `icap.server`, `icap.port`, and `icap.secure.port` global configuration properties.

APEX uses ORDS PL/SQL gateway. Once configured, this ICAP integration is also applied to file uploads in APEX.

To configure ORDS to integrate with ICAP server, perform the following steps:

Copyright © 2011, 2026, Oracle and/or its affiliates.

1. Run the following command:

```
ords --config <configuration_folder> config [--db-pool <pool_name>] set
icap.port <number>
ords --config <configuration_folder> config [--db-pool <pool_name>] set
icap.server <name_or_ip>
```

2. Restart Oracle REST Data Services.

ICAP server must support the following requirements:

- ICAP version 1.0
- Antivirus service named AVSCAN
- Antivirus service that supports action=SCAN
- Previews of at least 4 bytes
- Return header named X-Infection-Found

Once configured, when a file is uploaded through PL/SQL Gateway, ORDS makes a request similar to the following:

```
RESPMOD icap://<icap_server>:<icap_port>/AVSCAN?action=SCAN ICAP/1.0
Host: <icap_server>:<icap_port>
Preview: 4
Allow: 204
Encapsulated: req-hdr=0 res-hdr=153 res-body=200
```

6.8 Configuring the Custom Error Pages

This section explains how to configure a custom error page instead of the error page generated by Oracle REST Data Services.

To configure a custom error page,

1. Run the following command:

   ```
   ords --config /path/to/conf config set error.externalPath /path/to/error/
   pages/folder/
   ```

   Where:

   `/path/to/error/pages/folder` is the path to a folder containing files that define the error pages. The files are stored in `{status}.html` format. Where, `{status}` is the HTTP status code for which you want to create a custom error page.

2. Restart Oracle REST Data Services

Example 6-4 Configuring custom error page for “HTTP 404” status code

To configure a custom error page for the “HTTP 404 – Not Found” status, perform the following steps:

1. Create a file named `404.html`.
2. Save it under `/usr/local/share/ords/error-pages/` folder.
3. Configure the `error.externalPath` parameter to point to `/usr/local/share/ords/errro-pages/` folder.
4. Restart Oracle REST Data Services.

Copyright © 2011, 2026, Oracle and/or its affiliates.

6.9 Managing ORDS Administrator Privilege

Access to the ORDS_ADMIN PL/SQL package is provisioned through the ORDS_ADMINISTRATOR_ROLE. This role can be provisioned through the ORDS_ADMIN package to create additional ORDS administrators.

- Provisioning ORDS_ADMINISTRATOR_ROLE to a User
  This section describes how to provision ORDS_ADMINISTRATOR_ROLE role to a user.
- Unprovisioning ORDS_ADMINISTRATOR_ROLE from a User
  This section describes how to unprovision ORDS_ADMINISTRATOR_ROLE from a user.

6.9.1 Provisioning ORDS_ADMINISTRATOR_ROLE to a User

This section describes how to provision ORDS_ADMINISTRATOR_ROLE role to a user.

You can provision ORDS_ADMINISTRATOR_ROLE role to a user by using either the database GRANT command or through the ORDS_ADMIN.PROVISION_ADMIN_ROLE PL/SQL method (as an ORDS Administrator).

Example 6-5 Using Grant command

```
GRANT ORDS_ADMINISTRATOR_ROLE TO HR_ADMIN;
```

Example 6-6 Using ORDS_ADMIN package method

```
BEGIN
  ORDS_ADMIN.PROVISION_ADMIN_ROLE(
    p_user => 'HR_ADMIN');
END;
/
```

6.9.2 Unprovisioning ORDS_ADMINISTRATOR_ROLE from a User

This section describes how to unprovision ORDS_ADMINISTRATOR_ROLE from a user.

As an ORDS administrator, you can unprovision ORDS_ADMINISTRATOR_ROLE from a user by either using the database REVOKE command or through the ORDS_ADMIN.UNPROVISION_ROLES PL/SQL method.

Example 6-7 Using REVOKE command

```
REVOKE ORDS_ADMINSTRATOR_ROLE FROM HR_ADMIN;
```

Example 6-8 Using ORDS_ADMIN package method

```
BEGIN
  ORDS_ADMIN.UNPROVISION_ROLES(
    p_user => 'HR_ADMIN',
    p_administrator_role => TRUE);
 END;
 /
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

6.10 Managing ORDS Runtime Privilege

The ORDS_RUNTIME_ROLE database role allows a user to act as a runtime user. A runtime user can manage and configure the runtime connection resources required by an ORDS service instance. The ORDS_PUBLIC_USER is one such database user. When additional runtime users are provisioned, it is possible to configure discrete ORDS service instances with different destination addresses and connection pools but hosted on the same Oracle database container.

It is recommended not to re-use a runtime user for any other purpose as it accumulates the grants necessary to proxy to other users. A runtime user only requires the CREATE SESSION privilege in addition to the ORDS_RUNTIME_ROLE role.

- Provisioning ORDS_RUNTIME_ROLE to a User
  This section describes how to provision ORDS_RUNTIME_ROLE role to a user.
- Unprovisioning ORDS_RUNTIME_ROLE from a User
  This section describes how to unprovision the ORDS_RUNTIME_ROLE role from a user

6.10.1 Provisioning ORDS_RUNTIME_ROLE to a User

This section describes how to provision ORDS_RUNTIME_ROLE role to a user.

As an ORDS administrator, you can provision ORDS_RUNTIME_ROLE role to a user by using either the database GRANT command or through the ORDS_ADMIN.PROVISION_ADMIN_ROLE PL/SQL method.

Example 6-9 Using Grant command

```
GRANT ORDS_RUNTIME_ROLE TO ORDS_PUBLIC_USER_2;
```

Example 6-10 Using ORDS_ADMIN package method

```
BEGIN
  ORDS_ADMIN.PROVISION_RUNTIME_ROLE(
    p_user => 'ORDS_PUBLIC_USER_2');
END;
/
```

6.10.2 Unprovisioning ORDS_RUNTIME_ROLE from a User

This section describes how to unprovision the ORDS_RUNTIME_ROLE role from a user

As an administrator, you can unprovision the ORDS_RUNTIME_ROLE from a user, by either using the database REVOKE command or through the ORDS_ADMIN.UNPROVISION_ROLES PL/SQL method.

Example 6-11 Using REVOKE command

```
REVOKE ORDS_RUNTIME_ROLE FROM ORDS_RUNTIME_USER_2;
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Example 6-12 Using ORDS_ADMIN package method

```
BEGIN
  ORDS_ADMIN.UNPROVISION_ROLES(
    p_user => 'ORDS_RUNTIME_USER_2',
    p_runtime_role => TRUE);
 END;
 /
```

6.11 Using OAuth2 in Non-HTTPS Environments

RESTful Services can be protected with the OAuth2 protocol to control access to nonpublic data. To prevent data snooping, OAuth2 requires all requests involved in the OAuth2 authentication process to be transported using HTTPS. The default behavior of Oracle REST Data Services is to verify that all OAuth2 related requests have been received using HTTPS. It will refuse to service any such requests received over HTTP, returning an HTTP status code of 403 Forbidden.

This default behavior can be disabled in environments where HTTPS is not available as follows:

1. Locate the folder where the Oracle REST Data Services configuration is stored, for example: `/path/to/conf`
2. Execute the following command:
   `ords --config /path/to/conf config set security.verifySSL false`
3. Restart Oracle REST Data Services if it is running.

Note that it is only appropriate to use this setting in development or test environments. It is never appropriate to use this setting in production environments because it will result in user credentials being passed in clear text.

> **Note**
>
> Oracle REST Data Services must be restarted after making configuration changes. See your application server documentation for information on how to restart applications.

6.12 ORDS Performance Considerations

This section lists the key ORDS performance factors that are useful in achieving high performance.

- Configuring the Maximum Number of Rows Returned from a Query
- Configuring ORDS Metadata Cache
  This section explains how to configure the ORDS Metadata Cache.

6.12.1 Configuring the Maximum Number of Rows Returned from a Query

To configure maximum number of rows returned from a query, perform the following steps:

1. Run the following command:

Copyright © 2011, 2026, Oracle and/or its affiliates.

```
ords --config <configuration_folder> config set [--db-pool <pool_name>]
misc.pagination.maxRows <number>
```

> **Note**
>
> The default value for `misc.pagination.maxRows` is 10000.

2. Restart Oracle REST Data Services.

6.12.2 Configuring ORDS Metadata Cache

This section explains how to configure the ORDS Metadata Cache.

As the number of REST services grow, the overhead of querying the database for corresponding metadata can have a negative impact on the overall service performance and throughput. Overtime, the queries for `ORDS_METADATA` views take longer time to complete. These queries are executed for every request. The ORDS metadata cache can help improve the overall response time for REST services when the number of services grow to an extent that querying the `ORDS_METADATA` views for every request becomes expensive. The ORDS metadata cache can temporarily hold a copy of privilege and module metadata in memory to reduce the number of database queries performed when a REST service request is received. The cache is enabled by default so that the changes made to the metadata are applied immediately for any subsequent request.

Table 6-2 Configuration Properties for ORDS Metadata Cache

Property	Data Type	Default Value	Description
`cache.metadata.enabled`	Boolean	`true`	Specifies a setting to enable or disable metadata caching.
`cache.metadata.timeout`	Duration	`1s`	Specifies a setting that determines for how long the metadata record remains in the cache. Longer the duration, it takes longer to view the applied changes.

6.13 Configuring ORDS to Connect with OCI Database Tools Connections

You can configure Oracle REST Data Services (ORDS) to securely connect to your Oracle Database in Oracle Cloud Infrastructure (OCI) by leveraging OCI database tools connections.

This configuration approach, enables you to centralize the management of your database connection details, including the critical credentials for the Oracle runtime user (`ORDS_PUBLIC_USER`) and database wallets. Instead of embedding sensitive information directly in your ORDS configuration files, you can store and manage them securely within the OCI service. This not only simplifies the credential rotation and administration but also eliminates the need to store passwords on the compute instance.

To fetch the connection details from the centralized OCI service, ORDS must first authenticate to OCI. This can be achieved using one of the following two authentication methods:

Copyright © 2011, 2026, Oracle and/or its affiliates.

- **Instance Principal:** For ORDS running on an OCI compute instance.
- **OCI Profile:** For ORDS running on any machine with a configured OCI CLI profile.

Prerequisites

Ensure you have the following requirements

- ORDS version 25.2.0 or higher is installed.
- Oracle Database (for example: Autonomous Database) in your OCI tenancy is installed.
- OCI database tools connection that was created for your target database. This connection must be configured with the database credentials for the ORDS runtime user (for example: `ORDS_PUBLIC_USER`)

> **See Also**
>
> Creating a Connection

- Clean up all connection properties in the Oracle REST Data Services (ORDS) pool configuration that are no longer required. For each pool (using `--db-pool <pool-name>`), run the following commands to remove the outdated properties:

```
ords config --db-pool <pool-name> delete db.hostname

ords config --db-pool <pool-name> delete db.port

ords config --db-pool <pool-name> delete db.servicename

ords config --db-pool <pool-name> delete db.username
```

- Using Instance Principal Authentication
  This section describes how to authenticate using instance principal.
- Using OCI Profile Authentication
  This section describes the usage of OCI profile authentication method.

6.13.1 Using Instance Principal Authentication

This section describes how to authenticate using instance principal.

Oracle highly recommends using instance principal method as it is more seamless when ORDS is running on an OCI compute instance such as VM or container instance.

Prerequisites for using Instance Principal method

Your OCI compute instance must be configured as an instance principal, which involves adding it to a dynamic group and creating policies that grant it permission to access the database tools service.

Configuration Steps

Preform the following steps to authenticate using instance principal:

1. Configure the pool to use the Database Tools Connection type:

```
ords config --db-pool <pool-name> set db.connectionType
databaseToolsConnection
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
ords config --db-pool <pool-name> set db.databaseToolsConnection <dbtools-
connection-ocid>
```

2. Set `db.connectionType` to `databaseToolsConnection` and provide the OCID for `databaseToolsConnection`. The `db.authProvider` setting is not required, as it defaults to `oci-instance-principal`.
   Updated sample pool.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
    <comment>Pool configuration for OCI DBTools Connection</comment>
    <entry key="db.connectionType">databaseToolsConnection</entry>
    <entry
key="db.databaseToolsConnection">ocid1.databasetoolsconnection.oc1.eu-
frankfurt-1.amaaaaaa.....</entry>

    <!-- Other pool settings -->
    <entry key="feature.sdw">true</entry>
    <entry key="restEnabledSql.active">true</entry>
</properties>
```

3. Save the `pools.xml` file and restart ORDS.

ORDS automatically uses the identity of the instance to get authentication for OCI, retrieves the connection details from the specified database tools connection and establishes the database pool.

6.13.2 Using OCI Profile Authentication

This section describes the usage of OCI profile authentication method.

OCI profile authentication method is used when you are running ORDS outside of an OCI compute instance (for example: on-premises, on your local machine, or in another cloud) and you need to connect to a database in OCI.

Prerequisites for OCI Profile authentication

You must have the OCI Command Line Interface (CLI) installed and configured with a profile that has the necessary permissions to access the database tools service.

> **See Also**
>
> - Installing the CLI
> - SDK and CLI Configuration File

Configuration Steps

Perform the following steps to configure OCI CLI profile to authenticate and retrieve the connection details:

1. Configure the pool as follows:

```
ords config --db-pool default set db.connectionType
databaseToolsConnection
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
ords config --db-pool default set db.databaseToolsConnection <dbtools-
connection-ocid>
ords config --db-pool default set db.authProvider oci-profile
ords config --db-pool default set db.ociProfile <your-oci-profile>
```

2. Set `db.connectionType` to `databaseToolsConnection`.
3. Set `db.authProvider` to `oci-profile`.
4. Provide the OCID for `db.databaseToolsConnection`.
5. If your OCI CLI profile is not named `DEFAULT`, then specify it using the `db.ociProfile` setting. This step is optional.
   Updated sample pool.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
    <comment>Pool configuration for OCI DBTools Connection using OCI
Profile</comment>
    <entry key="db.connectionType">databaseToolsConnection</entry>
    <entry key="db.authProvider">oci-profile</entry>
    <entry
key="db.databaseToolsConnection">ocid1.databasetoolsconnection.oc1.iad.amaa
aaaa.....</entry>

    <!-- To use a specific profile named 'my-work-profile' -->
    <!-- <entry key="db.ociProfile">my-work-profile</entry> -->

    <!-- Other pool settings -->
    <entry key="feature.sdw">true</entry>
    <entry key="restEnabledSql.active">true</entry>
</properties>
```

6. Save the file and restart ORDS.

ORDS uses the credentials from your configured OCI CLI profile to authenticate and retrieve the connection details.

Copyright © 2011, 2026, Oracle and/or its affiliates.

7

Deploying and Monitoring Oracle REST Data Services

This chapter discusses how to deploy and monitor ORDS.

- Deploying Oracle REST Data Services
- Monitoring Oracle REST Data Services

7.1 Deploying Oracle REST Data Services

To deploy Oracle REST Data Service, you can choose one of the following options:

- Serve Commands for Running in Standalone Mode
- Oracle WebLogic Server
- Apache Tomcat
- Oracle Cloud Infrastructure

- Serve Commands for Running in Standalone Mode
  This section describes the serve command options for running in standalone mode.
- Oracle WebLogic Server
- Apache Tomcat
- Oracle Cloud Infrastructure

7.1.1 Serve Commands for Running in Standalone Mode

This section describes the serve command options for running in standalone mode.

Although Oracle REST Data Services supports the Java EE application servers, you also have the option of running it in standalone mode. This section describes how to run Oracle REST Data Services in a standalone mode.

Standalone mode is suitable for development use and is supported in production deployments. Standalone mode, however, has minimal management capabilities when compared to most Java EE application servers and may not have adequate management capabilities for production use in some environments.

> **Note**
>
> If you are planning to use the Oracle API for MongoDB, then you must install Oracle REST Data Services in a standalone mode. See Enabling and Configuring the Oracle Database API for MongoDB for more information.

- Non-Interactive Serve CLI
- Serve Command Options

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Ending an ORDS Process
  This section explains how to terminate an ORDS process.
- Converting a Private Key to DER (Linux and Unix)
- Configuring a Doc Root for Non-APEX Static Resources

7.1.1.1 Non-Interactive Serve CLI

The non-interactive serve CLI reads the standalone settings from the global `settings.xml` file located in the user-specified configuration folder or the default configuration folder.

```
$ ords --config <CONFIG_PATH> serve
$ ords --config <CONFIG_PATH> serve --port 8777 --apex-images /path/to/apex/
images
```

Example:

Create the Standalone Settings

> **Note**
>
> You can configure and start ORDS using the interactive install command.

- You can create, update, or delete the standalone settings using the following config CLI command:

  ```
  ords --config <your configuration folder> config set standalone.context.path /
  ords
  ```

- If you are using APEX, then you must provide the path to APEX images as follows:

  ```
  ords --config <your configuration folder> config set standalone.static.path /
  path/to/apex/images
  ```

- If you are using HTTPS protocol, then specify the following command:

  ```
  ords --config <your configuration folder> config set standalone.https.port
  8443
  ```

- If you are using a certificate, then specify the following command:

  ```
  ords --config <your configuration folder> config set
  standalone.https.cert /path/to/certificate/mycert.crt
  ords --config <your configuration folder> config set
  standalone.https.cert.key /path/to/certificate/mycert.key
  ```

7.1.1.2 Serve Command Options

Long Option	ShortOption	Description
`--apex-images-context-path <STATIC_CONTEXT_PATH>`		Specifies the context path where APEX static resources are located, defaults to `/i`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Long Option	ShortOption	Description
`--certificate <SSL_CERTIFICATE_PATH>`		Sets the certificate file for HTTPS, implies `--secure/--ssl` SSL certificate path. If you are providing the SSL certificate, you must specify the certificate location.
`--config <CONFIG_PATH>`	`c <CONFIG_PATH>-`	Specifies the path to the folder containing the configuration files.
`--context-path <CONTEXT_PATH>`		Sets the first part of the path of the URL from which ORDS is served. Defaults to `/ords`
`--document-root <DOCUMENT_ROOT_PATH>`		Specifies the path from which static files are served at the root of the server. The context path takes priority over static files that have the same name.
`--ip-addresses <INTERFACE_IP>`		Specifies ORDS which interface to bind to, use 0.0.0.0 (The default) to bind to all the network interfaces.
`--key <SSL_KEY_PATH>`		Sets the certificate key to use for HTTPS, implies `--secure/-ssl, requires --certificate`. Specifies the SSL certificate key path. If you are providing the SSL certificate, you must specify the certificate key location.
`--port <PORT>`		Specifies the HTTP listen port, default value is 8080. Set the port on which the server should run.
`--secure`		Specifies HTTPS listen port, default value is 8443. Must specify options `--secure --port <PORT>` to use HTTPS. Starts the server with HTTPS capabilities

7.1.1.3 Ending an ORDS Process

This section explains how to terminate an ORDS process.

To terminate the ORDS process, use the Control + C keys on Mac or Linux to stop the running ORDS process. This will end both the ORDS process and its associated JVM process. Alternatively, you can manually locate the ORDS and JVM process IDs and kill them.

- To find the process IDs (PIDs), you can use:
  - `ps` on macOS/Linux
  - `tasklist` in Command Prompt
  - `Get-Process` in PowerShell
  - Activity Monitor on macOS
  - Task Manager on Windows

To terminate the processes, you can use the following methods:

- On macOS/Linux: `kill [ORDS PID] [JVM PID]`
- In Command Prompt: `taskkill /PID [ORDS PID] /PID [JVM PID] /F`

Copyright © 2011, 2026, Oracle and/or its affiliates.

- In PowerShell: `Stop-Process -ID [ORDS PID], [JVM PID] -Force`
- Manually end the process using Activity Monitor on macOS
- Manually end the process using Task Manager on Windows

> **Note**
>
> ORDS also provides a standalone server manager that integrates with Linux system managers such as Systemd and System V. For more information, refer to the `README.md` file located at `[ords product folder]/linux-support/` directory.

> **See Also**
>
> - **macOS/Linux**:
>   - ps command
>   - kill command
> - **PowerShell**
>   - Get-Process
>   - End-Process
> - **Command prompt**
>   - tasklist
>   - taskkill

7.1.1.4 Converting a Private Key to DER (Linux and Unix)

Usually, you would have created a private key and a Certificate Signing Request before obtaining your signed certificate. The private key needs to be converted into DER in order for Oracle REST Data Services to read it properly.

For example, assume that the original private key was created using the OpenSSL tool with a command similar to either of the following:

```
openssl req -new -newkey rsa:2048 -nodes -keyout yourdomain.key -out
yourdomain.csr
```

or

```
openssl genrsa -out private.em 2048
```

In this case, you must run a command similar to the following to convert it and remove the encryption: openssl pkcs8 -topk8 -inform PEM -outform DER -in yourdomain.key -out yourdomain.der -nocrypt

```
openssl pkcs8 -topk8 -inform PEM -outform DER -in yourdomain.key -out
yourdomain.der -nocrypt
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

After doing this, you can include the path to your certificate and and the key of the certificate. Example: `yourdomain.crt, yourdomain.der`, when you are prompted while using the `ords` install interactive command.

You can also add or modify the entries using the following ORDS configuration set commands:

```
ords --config <your configuration folder> config set standalone.https.cert
      /path/to/yourdomain.crt


ords --config <your configuration folder> config set standalone.https.cert.key
      /path/to/yourdomain.der
```

Verify your configuration settings using the following command:

```
ords --config <your configuration folder> config list
```

Also, ensure that your https port configuration setting, `standalone.https.port` contains a value:

```
ords --config <your configuration folder> config set standalone.https.port
8443
```

7.1.1.5 Configuring a Doc Root for Non-APEX Static Resources

You can configure a doc root for standalone mode to deploy static resources that are outside the `/i` folder that is reserved for APEX static resources.

To do so, execute the following command:

```
ords --config /path/to/conf config set standalone.doc.root /var/www/html .
```

The preceding example makes any resource located within `/var/www/html` available under `http://server:port/`. For example, if the file `/var/www/html/hello.txt` exists, it will be accessible at `http://server:port/hello.txt`.

The value specified for `standalone.doc.root` is stored in `${config.dir}/global/settings.xml` in the `standalone.doc.root property`. If a custom doc root is not specified, then the default doc root value of `${config.dir}/global/doc_root` is used. Any file placed within this folder is available at the root context.

This approach has the following features and considerations:

- HTML resources can be addressed without their file extension. For example, if a file named `hello.html` exists in the doc root, it can be accessed at the URI `http://server:port/hello`.
- Attempts to address a HTML resource with its file extension are redirected to the location without an extension. For example, if the URI `http://server:port/hello.html` is accessed, then the client is redirected to `http://server:port/hello`.

  The usual practice is to serve HTML resources without their file extensions, so this feature facilitates that practice, while the redirect handles the case where the resource is addressed with its file extension.
- Index pages for folders are supported. If a folder contains a file named `index.html` or `index.htm`, then that file is used as the index page for the folder. For example, if `/var/www/`

Copyright © 2011, 2026, Oracle and/or its affiliates.

`html` contains `/abc/xyz/index.html`, then accessing `http://server:port/abc/xyz/` displays the contents of `index.html`.

- Addressing a folder without a trailing slash causes a redirect to the URI with a trailing slash. For example, if a client accesses `http://server:port/abc/xyz`, then the server issues a redirect to `http://server:port/abc/xyz/`.
- Resources are generated with weak etags based on the modification stamp of the file and with a Cache Control header that causes the resources to be cached for 1 hour.

7.1.2 Oracle WebLogic Server

This section describes how to deploy Oracle REST Data Services on Oracle WebLogic Server. It assumes that you have completed the installation process and are familiar with Oracle WebLogic Server. If you are unfamiliar with domains, managed servers, deployment, security, users and roles, refer to your Oracle WebLogic Server documentation.

Topics:

- About Oracle WebLogic Server
- Configuring Oracle APEX Images
- Launching the Administration Server Console
- Deploying ORDS on Oracle WebLogic Server
- Configuring WebLogic to Handle HTTP Basic Challenges Correctly
- Verifying the State and Health of ords and i

- About Oracle WebLogic Server
- Configuring Oracle APEX Images
- Launching the Administration Server Console
- Deploying ORDS on Oracle WebLogic Server
- Configuring WebLogic to Handle HTTP Basic Challenges Correctly
- Verifying the State and Health of ords and i

7.1.2.1 About Oracle WebLogic Server

You can download Oracle WebLogic Server from Oracle Technology Network.

To learn more about installing Oracle WebLogic Server, see *Oracle Fusion Middleware Getting Started With Installation for Oracle WebLogic Server* and *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

> **See Also**
>
> Oracle Fusion Middleware Software Downloads

7.1.2.2 Configuring Oracle APEX Images

This section is only applicable if you are using Oracle APEX. If you are not using Oracle APEX, then you can skip this section that provides details about configuring Oracle APEX Images.

Copyright © 2011, 2026, Oracle and/or its affiliates.

If you are using Oracle APEX, then the Oracle APEX image files for that Oracle APEX version must also be accessible to the end user. It is recommended that you configure the Oracle APEX instance wherever applicable to use the resources from the Oracle Content Network. The use of the Oracle CDN may not be feasible when you are using Oracle APEX in a network that restricts internet access. In such a scenario, you must deploy a web application to provide the Oracle APEX image files.

Before you begin, you must identify the directory where you extracted the Oracle APEX distribution. In the following steps, this directory is referred to as `<apex directory>`. You then define a WebLogic web application which refers to this location and deploy that web application with a context-root value which is the default Oracle APEX image prefix: `i`.Create a directory which is the source for your web archive. In that directory, create a `WEB-INF` directory with the following files:

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE web-app PUBLIC
"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
 <!-- This Web-App leverages the alternate doc-root functionality in WebLogic
to serve static content
      For WebLogic refer to the weblogic.xml file in this folder
  -->
</web-app>
```

weblogic.xml

```
<weblogic-web-app xmlns="http://www.bea.com/ns/weblogic/weblogic-web-app">
 <!-- This element specifies the context path the static resources are served
from -->
 <context-root>/i</context-root>
 <virtual-directory-mapping>
  <!-- This element specifies the location on disk where the static resources
are located -->
  <local-path>/scratch/apex/apex_21.2_en/images</local-path>
  <url-pattern>/*</url-pattern>
 </virtual-directory-mapping>
</weblogic-web-app>
```

Ensure that the local-path value in the weblogic.xml corresponds to your <apex directory>/images directory.

Where:

- *<apex directory>* is the directory location of Oracle APEX.

Create a Web Application with the `web.xml` and `weblogic.xml` files

```
jar cMf i.war WEB-INF/web.xml WEB-INF/weblogic.xml
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> Do not perform this action in the ORDS product directory. No other `war` file other than `ords.war` should exist in the ORDS product directory.

The created images WAR does not contain the static resources; instead, it references the location where the static resources are stored. Therefore the static resources must be available at the specified path on the server where the WAR is deployed.

Deploy the `i.war` file to your WebLogic Server following the deployment steps that are applicable to your WebLogic Server environment. Once deployed, access the following URL to verify that the content is available:

```
http(s)://hostname:port/i/apex_version.txt
```

In response, the Oracle APEX version you have configured is displayed.

> **Tip**
>
> The `i.war` has a context-root with value `/i`. This is the default Oracle APEX image prefix. If you have configured Oracle APEX with the different image prefix value, then the context-root must be changed to that value.

7.1.2.3 Launching the Administration Server Console

To launch the Administration Server console:

1. Start an Administration Server.
2. Launch the WebLogic Administration Console by typing the following URL in your web browser:

   ```
   http://<host>:<port>/console
   ```

   Where:

   - *<host>* is the DNS name or IP address of the Administration Server.
   - *<port>* is the port on which the Administration Server is listening for requests (port `7001` by default).
3. Enter your WebLogic Administrator username and password.
4. If your domain is in *Production* mode, click the **Lock & Edit** button on the left-pane below the submenu Change Center. If your domain is in *Development* mode, this button does not appear.

7.1.2.4 Deploying ORDS on Oracle WebLogic Server

> **Tip**
>
> The Oracle REST Data Services files, `ords.war` and `i.war`, must be available before you start this task.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> When you run ORDS in standalone mode using the ORDS `serve` command, you can specify the location of the configuration directory.
>
> For deploying the `ords.war` in a supported container such as WebLogic Server, you must specify the configuration directory location. To do this, use one of the following approaches:
>
> - The recommended approach is to generate a Web Application archive file with the configuration directory location specified for that `war` file.
> - Set the `config.url` system property in the managed server so that it applies to all the ORDS deployments.
>   To generate the `war` file, use the `ords war` command.
>
>   To set the `config.url` system property, before starting WebLogic Server, run this command: `export JAVA_OPTIONS="-Dconfig.url=/scratch/my_ords_config"`

To deploy:

1. Go to the WebLogic Server Home Page. Below Domain Configuration, select **Deployments**.

   The Summary of Deployments is displayed.
2. Click **Install**.
3. Specify the location of the `ords.war` file and click **Next**.

   The `ords.war` file is located in the folder where you unzipped the Oracle REST Data Services ZIP file.

   > **Tip**
   >
   > WebLogic Server determines the context root from the file name of a WAR archive. If you need to keep backward compatibility, so that URLs are of the form *http://server/apex/...* rather than *http://server/ords/...*, then you must rename `ords.war` to `apex.war` before the deployment.

   The Install Application assistant is displayed.
4. Select **Install this deployment as an application** and click **Next**.
5. Select the servers and/or clusters to which you want to deploy the application or module and click **Next**.

   > **Tip**
   >
   > If you have not created additional Managed Servers or clusters, you do not see this assistant page.

6. In the Optional Settings, specify the following:
   a. Name - Enter:

      ```
      ords
      ```


Copyright © 2011, 2026, Oracle and/or its affiliates.

b. Security - Select the following:

Custom Roles: Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor

c. Source accessibility - Select:

Use the defaults defined by the deployment's targets

7. Click **Next**.

A summary page is displayed.

8. Under Additional configuration, select one of the following:

- **Yes, take me to the deployment's configuration** - Displays the Configuration page.
- **No I will review the configuration later** - Returns you to the Summary of Deployments page.

9. Review the summary of configuration settings that you have specified.

10. Click **Finish**.

11. Repeat the previous steps to deploy the `i.war` file.

In the optional settings, specify the following:

a. Name - Enter:

`i`

b. Security - Select:

Custom Roles: Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor

c. Source Accessibility - Select:

Use the defaults defined by the deployment's targets

12. If your domain is in Production Mode, then on the Change Center click **Activate Changes**.

Related Topics

- Installing and Configuring Oracle REST Data Services
  This chapter describes how to install, configure, and upgrade Oracle REST Data Services (ORDS).
- Configuring Oracle APEX Images

7.1.2.5 Configuring WebLogic to Handle HTTP Basic Challenges Correctly

By default WebLogic Server attempts to intercept all HTTP Basic Authentication challenges. This default behavior needs to be disabled for Oracle REST Data Services to function correctly. This is achieved by updating the `enforce-valid-basic-auth-credentials` flag. The WebLogic Server Administration Console does not display the `enforce-valid-basic-auth-credentials` setting. You can use WebLogic Scripting Tool (WLST) commands to check, and edit the value in a running server.

The following WLST commands display the domain settings:

```
connect('weblogic','weblogic','t3://localhost:7001')
cd('SecurityConfiguration')
cd('mydomain')
ls()
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

If the domain settings displayed, contains the following entry:

```
-r--   EnforceValidBasicAuthCredentials             true
```

Then you must set this entry to `false`.

To set the entry to `false`, use the WLST commands as follows:

```
connect('weblogic', 'weblogic', 't3://localhost:7001')
edit()
startEdit()
cd('SecurityConfiguration')
cd('mydomain')
set('EnforceValidBasicAuthCredentials','false')
save()
activate()
disconnect()
exit()
```

> **Note**
>
> WebLogic Server must be restarted for the new settings to take effect.

In the preceding example:

- `weblogic` is the WebLogic user having administrative privileges
- `weblogic` is the password
- `mydomain` is the domain
- The AdminServer is running on the `localhost` and on port `7001`

Related Topics

- WebLogic Server Command Reference

7.1.2.6 Verifying the State and Health of ords and i

In the Summary of Deployments, select the **Control** tab and verify that both the `ords` and `i` State are Active and the Health status is OK.

If `ords` and/or `i` are not Active, then enable them. In the Deployments table, select the check box next to `ords` and/or `i`. Click **Start** and select **Servicing all requests** to make them active.

7.1.3 Apache Tomcat

This section describes how to deploy Oracle REST Data Services on Apache Tomcat.

Topics:

- About Apache Tomcat
- Configuring Oracle APEX Images
- Deploying ORDS on Apache Tomcat

Copyright © 2011, 2026, Oracle and/or its affiliates.

- About Apache Tomcat
- Configuring Oracle APEX Images
- Deploying ORDS on Apache Tomcat

7.1.3.1 About Apache Tomcat

> **Tip**
>
> This section assumes that you have completed the installation process and are familiar with Apache Tomcat. If you are unfamiliar with domains, servers, applications, security, users and roles, see your Apache Tomcat documentation.

You can download Apache Tomcat from:

> **See Also**
>
> Tomcat 8 Software Downloads

7.1.3.2 Configuring Oracle APEX Images

This section is only applicable if you are using Oracle APEX. If you are not using Oracle APEX, then you can skip this section that provides details about configuring Oracle APEX Images.

If you are using Oracle APEX, then the Oracle APEX image files for that Oracle APEX version must also be accessible to the end user. It is recommended that you configure the Oracle APEX instance wherever applicable to use the resources from the Oracle Content Network.

The use of the Oracle CDN may not be feasible when using Oracle APEX in a network that restricts internet access. In such a scenario, you must deploy a web application to provide the Oracle APEX image files.

To configure Oracle APEX Images on Apache Tomcat:

- Copy the contents of the <*apex directory*>/images folder to <*Tomcat directory*>/webapps/i/.

  Where:

  – <*apex directory*> is the directory location of the Oracle APEX distribution.

  – <*Tomcat directory*> is the folder where Apache Tomcat is installed.

7.1.3.3 Deploying ORDS on Apache Tomcat

> **Tip**
>
> The Oracle REST Data Services file ords.war must be available before you start this task.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> When you run ORDS in standalone mode using the ORDS `serve` command, you can specify the location of the configuration directory.
>
> For deploying the `ords.war` in a supported container such as Apache Tomcat, you must specify the configuration directory location. To do this, use one of the following approaches:
>
> - The recommended approach is to generate a Web Application archive file with the configuration directory location specified for that `war` file.
> - Set the `config.url` system property in the managed server so that it applies to all the ORDS deployments.
>
> To generate the `war` file, use the `ords war` command.
>
> To set the `config.url` system property, before starting Apache Tomcat, run this command: `export JAVA_OPTS="-Dconfig.url=/scratch/my_ords_config"`

To deploy ORDS on Apache Tomcat:

1. Move the `ords.war` file into the `webapps` folder where Apache Tomcat is installed.

   > **Tip**
   >
   > Apache Tomcat determines the context root from the file name of a WAR archive. If you need to keep backward compatibility, so that URLs are of the form *http://server/apex/...* rather than *http://server/ords/...*, then you must rename `ords.war` to `apex.war` before moving it into to the `webapps` folder.

2. Access Oracle APEX by typing the following URL in your web browser:

   `http://<hostname>:<port>/ords/`

   Where:

   - <*hostname*> is the name of the server where Apache Tomcat is running.
   - <*port*> is the port number configured for Apache Tomcat application server.

Related Topics

- Installing and Configuring Oracle REST Data Services
  This chapter describes how to install, configure, and upgrade Oracle REST Data Services (ORDS).
- Configuring Oracle APEX Images

7.1.4 Oracle Cloud Infrastructure

As a cloud user, you can deploy Oracle REST Data Services with high availability on Oracle Cloud Infrastructure.

Topics:

- About Oracle Cloud Infrastructure (OCI)
- Configuring ORDS for High Availability on Oracle Cloud Infrastructure

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Advantages of Deploying ORDS with High Availability on Oracle Cloud Infrastructure
- About Oracle Cloud Infrastructure
- Configuring ORDS for High Availability on Oracle Cloud Infrastructure
- Advantages of Deploying ORDS with High Availability on Oracle Cloud Infrastructure

> **See Also**
>
> Reference Architecture: Deploying ORDS with High Availability

7.1.4.1 About Oracle Cloud Infrastructure

Oracle Cloud Infrastructure (OCI) is a set of complementary cloud services that enable you to build and run a wide range of applications and services in a highly available hosted environment. Oracle Cloud Infrastructure offers high-performance compute capabilities (as physical hardware instances) and storage capacity in a flexible overlay virtual network that is securely accessible from your on-premises network.

7.1.4.2 Configuring ORDS for High Availability on Oracle Cloud Infrastructure

You can choose to use the following configurations for high availability of ORDS on Oracle Cloud Infrastructure. Your requirements might differ from the configurations described in this section.

- When you create a VCN, determine the number of CIDR blocks required and the size of each block based on the number of resources that you plan to attach to subnets in the VCN. When you design the subnets, consider your traffic flow and security requirements.
- Use Oracle Cloud Guard to proactively monitor and maintain the security of your resources in Oracle Cloud Infrastructure.
- For production applications, the Oracle database instance should be adhering to Oracle Maximum Availability Architecture (MAA) deployment model in OCI.
- When using RAC with the Oracle Database, ensure that the database connection information used by ORDS is pointing to the SCAN listener and not an individual node.
- You can create load balancers with upper and lower bounds to help scale based on the number of requests coming in. It can be as small as 10mbps up to 8000mbps.

7.1.4.3 Advantages of Deploying ORDS with High Availability on Oracle Cloud Infrastructure

Following are the advantages of deploying ORDS on Oracle Cloud Infrastructure:

- Compute, load balancers, and Database Cloud Instances can all scale to handle increased load. Additional instances can be quickly created and added to the Load Balancer configuration with the compute or ORDS tier. For Exadata Cloud Service, the X8M platform can not only scale CPU, but also the nodes can be added to the RAC cluster to add additional computing power.
- Ensure that your subnet and NSG ingress/egress uses very granular rules. To get access to a compute or database tier, use Bastion as a Service, this ensures that only authorized users can access these instances and is also a much more secure method than exposing SSH ports to the public internet.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- For database deployments, adhere to the Oracle Maximum Availability Architecture (MAA) guide.
- Auto-scaling and scaling in general for each compute and database tier helps control costs and you need to pay only for what is being used with no excess or wasted CPU, memory, or instances. Using a flexible load balancer can also control costs.

7.2 Monitoring Oracle REST Data Services

Standard Java runtime environment diagnostic and monitoring tools are used to gain an insight on the health of an Monitoring Oracle REST Data Services (ORDS) instance running in Apache Tomcat, WebLogic Server, or a standalone mode. These tools track the memory and CPU usage, stuck threads, and other resources. ORDS provides additional insight through the ORDS instance API. The metrics available through the instance API makes it possible to check the status (valid or invalid) of the database pools and to gauge how the pools are being used. This helps in determining the actual load on the system and inform configuration changes in the future.

Topics:

- Using the Instance API
- Using OpenTelemetry with Oracle REST Data Services
- ORDS CLI Commands for Managing Pools Using JMX

- Using the Instance API
- Using OpenTelemetry with Oracle REST Data Services
  This section describes how to configure ORDS to use OpenTelemetry.
- ORDS CLI Commands for Managing Pools Using JMX
  This section describes the ORDS CLI commands to list or destroy the connection pools through JMX based pool management.
- Using OCI Monitoring Service with Oracle REST Data Services
  This section describes how to configure ORDS to send metrics to the Oracle Cloud Infrastructure (OCI) monitoring service and configure alarms.
- Using OCI Logging Service with Oracle REST Data Services
  This sections describes how to configure Java logging in ORDS for JSON-formatted output and on how to set up an agent to forward these logs to the OCI logging service.

7.2.1 Using the Instance API

The ORDS instance API service neither provides access to the database nor does it require the client to specify a database user for authentication. However, the ORDS instance returns information on the database pools. The instance API can be used as a basic health check service. To get a summary of the number of valid and invalid database pools, send a GET request to `/ords/_/instance-api/stable/status`. For example: `curl --user sysadmin:oracle http://<server>/ords/_/instance-api/stable/status`. This service returns a count of valid and invalid pools. It also returns links to additional information with more details on the database pools cache.

ORDS can be deployed as a single instance or in a cluster. In a cluster, you must address each instance directly to get the specific information about that specific instance as the database pool statistics for one instance may differ from the other instance. However, if the load balancer routes to each instance in a round robin basis (as recommended), then every instance will have similar pool statistics.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Enabling the ORDS Instance API
  This section explains how to enable the ORDS instance API.
- Authorization for Using the ORDS Instance API
- API Document

7.2.1.1 Enabling the ORDS Instance API

This section explains how to enable the ORDS instance API.

To enable the ORDS instance API, use the following:

- `ords -config /path/to/config config set -global instance.api.enabled true`
- Restart ORDS

7.2.1.2 Authorization for Using the ORDS Instance API

The System Administrator role is required to use the ORDS instance API. For production environments, it is recommended that a user with this role is configured through the mid-tier.

7.2.1.3 API Document

An OpenAPI description of the ORDS instance API services is available at `http://<server>/ords/_/instance-api/stable/metadata-catalog/openapi.json`.

7.2.2 Using OpenTelemetry with Oracle REST Data Services

This section describes how to configure ORDS to use OpenTelemetry.

ORDS can be configured to provide telemetry data (metrics, logs, and traces) through OpenTelemetry. OpenTelemetry, also known as OTEL in short, is a vendor-neutral open-source observability framework for instrumenting, generating, collecting, and exporting telemetry data to assist in the analysis and tuning of software systems. As an industry standard it is natively supported by a number of vendors.

Configure ORDS to export telemetry data to an OpenTelemetry compliant collector

Perform the following steps to run ORDS to exports telemetry data to an OpenTelemetry collector:

1. Download the latest OpenTelemetry Java Agent.
2. Configure OpenTelemetry Java Agent

> **See Also**
> - Download OpenTelemetry Java agent
> - Configure OpenTelemetry Java Agent

3. Add `-javaagent` option using `--java-options` parameter when using ORDS standalone or `CATALINA_OPTS` environment variable for Tomcat.

Copyright © 2011, 2026, Oracle and/or its affiliates.

The following example configures OpenTelemetry agent to export telemetry data to an OpenTelemetry Collector, running on host `otel-collector-host` and running ORDS in a standalone mode:

```
export OTEL_SERVICE_NAME=ords
export OTEL_METRICS_EXPORTER=none
export OTEL_TRACES_EXPORTER=otlp
export OTEL_EXPORTER_OTLP_ENDPOINT=http://otel-collector-host:4317
export JDK_JAVA_OPTIONS=
ords --java-options "-javaagent:/path/to/opentelemetry-javaagent.jar" --
config /path/to/config serve
```

When you deploy ORDS on Apache Tomcat or Oracle WebLogic Server, the same Java Agent and environment variables can be specified. See Apache Tomcat or Oracle WebLogic server documentation for the most appropriate approach to use for that platform.

Configuring OpenTelemetry Collector to export telemetry data to Oracle APM

Oracle APM provides a comprehensive set of features to monitor the applications. It supports OpenTelemetry tracing.

To export telemetry traces to APM, perform the following steps:

1. Create an APM Domain in OCI.
2. Record the `<Data upload endpoint>` of the APM domain.
3. Create and record the `<Private Data Key>` of the APM domain.
4. Install `otelcol`.
   Refer to OpenTelemetry documentation for the installation instructions
5. Configure the `Zipkin` exporter in the `otelcol` configuration file. For example: `/etc/otelcol/config.yaml` on `Linux`. Replace `<Data Upload Endpoint>` and `<Private Data Key>` with the actual endpoints and keys.
   Example code snippet:

   Language: yml

   ```
   extensions:
     health_check:
     pprof:
       endpoint: 0.0.0.0:1777
     zpages:
       endpoint: 0.0.0.0:55679

   receivers:
     otlp:
       protocols:
         grpc:
         http:

   processors:
     batch:

   exporters:
     logging:
       verbosity: detailed
   ```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
  zipkin:
    endpoint: "<Data Upload Endpoint>/20200101/observations/private-span?
dataFormat=zipkin&dataFormatVersion=2&dataKey=<Private Data Key>"

service:
  pipelines:
    traces:
      receivers: [otlp]
      processors: [batch]
      exporters: [logging, zipkin]

    metrics:
      receivers: [otlp]
      processors: [batch]
      exporters: [logging]

  extensions: [health_check, pprof, zpages]
```

6. Run the `otelcol` service with the updated configuration.
   With this configuration, OpenTelemetry Collector (`otelcol`) exports telemetry traces to Oracle APM using the `Zipkin` exporter with the specified `<Data Upload Endpoint>` and `<Private Data Key>`

> **Note**
>
> The `OTEL_xxxx` environment variables are described in detail in OpenTelemetry documentation.

> **See Also**
>
> - OpenTelemetry Documentation
> - Apache Tomcat
> - Oracle WebLogic Server

7.2.3 ORDS CLI Commands for Managing Pools Using JMX

This section describes the ORDS CLI commands to list or destroy the connection pools through JMX based pool management.

ORDS pool command uses the Java Attach API to connect to the local Java process running ORDS and then lists or destroys the Universal Connection Pools (UCP) through JMX based pool management.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> - JMX based pool management
> - Attach API
> - Local Monitoring and Management Using JConsole

In order to use the `ords pool` command:

- ORDS must have JMX UCP enabled. It is enabled by default.
- You must specify the Oracle REST Data Services instance process ID.

The `ords pool` command can be used to list the pools or destroy the pools without the need to restart the service.

- **List the pools in a running ORDS instance**:

  > **Note**
  >
  > Before running the `list` command, identify the process ID of the ORDS instance. To identify the process ID of the ORDS instance, use the command `jps -ml` to list the running Java processes on the local host.

  Run the following command to list the available pools, specify the identified process ID:

  ```
  ords pool --pid 3459 list
  ```

  In response, the list of available pools is displayed in the console.

- **Destroy a pool in a running ORDS instance**:
  To destroy the specified pool name:

  Run the following destroy command specifying the process ID and the pool name enclosed in double quotes:

  ```
  ords pool --pid 3459 destroy "|pdb2|lo|-2023-05-24T21-28-03.076379Z"
  ```

  A message is displayed indicating that the pool has been destroyed

  The pool `|pdb2|lo|-2023-05-24T21-28-03.076379Z` was destroyed.

7.2.4 Using OCI Monitoring Service with Oracle REST Data Services

This section describes how to configure ORDS to send metrics to the Oracle Cloud Infrastructure (OCI) monitoring service and configure alarms.

OCI monitoring service provides comprehensive capabilities for monitoring cloud resources and applications, including metrics and alarms. It uses metrics to monitor resources and alarms for notifying the administrators when the metrics meet the specified triggers. ORDS provides an utility to process the access logs generated by the server and send metrics to the OCI monitoring service. Administrators can utilize OCI dashboard and metrics explorer to monitor and troubleshoot ORDS related issues.

- ords-metrics Utility
  `ords-metrics` is an utility that is used to process the ORDS access logs and send metrics to OCI monitoring service.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Configuring ORDS to Send Metrics to OCI Monitoring Service
- Enable JMX Metrics
  ORDS provides a range of JMX metrics, which can be sent to the OCI monitoring service when the appropriate JMX metrics groups are enabled in ords-metrics using the `-m` option.
- Configuring OCI Alarms
  This section outlines the steps for configuring alarms in Oracle Cloud Infrastructure (OCI).

7.2.4.1 ords-metrics Utility

`ords-metrics` is an utility that is used to process the ORDS access logs and send metrics to OCI monitoring service.

ords-metrics Utility Options

ords-metrics utility supports the following options:

Option	Description	Required
`-d logdir`	Specifies the access log directory. Default value is `/var/log/ords/access`.	no
`-n namespace`	Specifies the OCI monitoring namespace. Must match the following pattern: `^[a-z][a-z0-9_]*[a-z0-9]$` Default value is: `dbtools_ords`.	no
`-r resource`	Specifies the OCI monitoring resource group. Default value is `ords`.	no
`-c compartment`	Specifies the OCI metric compartment OCID.	yes
`-p profile`	Specifies the OCI profile name. Instance principal is used when this option is not specified.	no
`-a prefix`	Specifies add prefix to the metric name.	no
`-b heartbeat`	Specifies the heartbeat interval in second. Default value is 60. To disable, set the value to 0 .	no
`-H host`	Specifies the hostname. Default value is `FQDN` of the local host.	no
`-t`	Specifies the sent response time metrics (in milliseconds).	no
`-i include`	Specifies the comma separated list of status code or range of code to include. For example: 200, 500-599. Default range is 100-599.	no
`-v`	Specifies verbose mode.	no
`-m jmx`	Specifies a comma separated list of JMX metrics to collect. For example: `jvm`, `os`, `thread`, `ucp`	no
`-o paths`	Specifies a comma separated list of request path to use in a metrics dimension.	no
`-f pidfile`	Specifies the path for ORDS PID file.	no
`-t`	Specifies the sent response time metrics (in milliseconds).	no
`-q`	Exits after processing the existing records	no

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 7-1 ORDS Metric Name

Name	Description
`HttpStatus<code>`	Specifies the HTTP status code. For example: `HttpStatus200`
`Heartbeat`	Specifies the heartbeat of the `ords-metrics`.

Table 7-2 ORDS Metrics Dimensions

Name	Description
`host`	Specifies the ORDS Hostname.
`availabilityDomain`	Specofies the OCI Instance Availability Domain.
`faultDomain`	Specifies the OCI Instance Fault Domain.
`path`	Specifies the request base path (configurable with `-o` paths).

7.2.4.2 Configuring ORDS to Send Metrics to OCI Monitoring Service

Perform the following steps to configure ORDS to send the metrics to the OCI monitoring service:

1. Enable access log in global `settings.xml` file. The following example configures ORDS to create access logs under the directory `/var/log/ords/access` and retain the files for 10 days:

   ```
   <entry key="standalone.access.log">/var/log/ords/access</entry>
   <entry key="standalone.access.log.retainDays">10</entry>
   ```

2. Configure `ords-metrics` to send metrics to the OCI public monitoring service.
   - **With API key and OCI profile**:
     a. Generate an API key
     b. Create an OCI profile with `key_file` set to the API key.
     c. Run `bin/ords-metrics` with `-d <logdir> -p <profile> -c <compartment> [ -n <namespace> ]`
   - **With instance principal:**
     a. Create a dynamic group with matching rules that include the ORDS compute instance
     b. Create a policy granting permissions to the dynamic group to access OCI monitoring service in your compartment
     c. Run `bin/ords-metrics` with `-d <logdir> -c <compartment> [ -n <namespace> ]` on the ORDS instance

7.2.4.3 Enable JMX Metrics

ORDS provides a range of JMX metrics, which can be sent to the OCI monitoring service when the appropriate JMX metrics groups are enabled in ords-metrics using the `-m` option.

Copyright © 2011, 2026, Oracle and/or its affiliates.

JVM Statistics	Description
`jvm`	Specifies the JVM statistics.
`thread`	Specifies the Thread statistics
`os`	Specifies the operating system metrics.
`ucp`	Specifies the UCP Pool metrics.

The following example enables sending `jvm` and `ucp` metrics to OCI monitoring service:

```
ords-metrics -d /var/log/ords/access -c ocid1.compartment... -m jvm,ucp
```

The ords-metrics utility automatically discovers the process ID (PID) of the ORDS process. However, if multiple ORDS processes are running on a system, the `-f` option can be used to specify the PID file for the target ORDS instance.

7.2.4.4 Configuring OCI Alarms

This section outlines the steps for configuring alarms in Oracle Cloud Infrastructure (OCI).

Perform the following steps to configure alarms in OCI:

1. Create an OCI notification topic

   Copy the create alarm script `examples/ords-metrics/create_alarms.sh` to another location and replace the following variables with your desired settings:

   ```
   export alarm_compartment_id=ocid1.compartment...
   export metric_compartment_id=ocid1.compartment...
   export topic_id=ocid1.onstopic...
   export namespace=dbtools_ords
   ```

2. Adjust the alarm thresholds based on your requirement
3. Run the customized `create_alarm.sh` script

7.2.5 Using OCI Logging Service with Oracle REST Data Services

This sections describes how to configure Java logging in ORDS for JSON-formatted output and on how to set up an agent to forward these logs to the OCI logging service.

To fully leverage OCI logging with ORDS, configure ORDS to generate logs in a JSON format and deploy a logging agent to efficiently route these messages. This integration enables centralized log management, real-time monitoring, and seamless adherence to organizational and regulatory compliance requirements.

- Configure Java Logging in ORDS
  This section outlines the steps for configuring Java logging in ORDS.
- Create an agent configuration
  This section explains how to create an agent configuration.

7.2.5.1 Configure Java Logging in ORDS

This section outlines the steps for configuring Java logging in ORDS.

Perform the following steps to configure Java logging in ORDS:

Copyright © 2011, 2026, Oracle and/or its affiliates.

1. Create a `logging.properties` file that configures a `FileHandler` to use `oracle.dbtools.logging.JSONLogFormatter`, enabling the logging messages in JSON format.
   The following sample configuration demonstrates how to set up file-based logging with automatic log rotation and the ORDS JSON log formatter:

   ```
   handlers= java.util.logging.FileHandler, java.util.logging.ConsoleHandler

   # Configure the FileHandler
   java.util.logging.FileHandler.level = INFO
   java.util.logging.FileHandler.pattern = /path/to/ords.log
   java.util.logging.FileHandler.limit = 10485760    # 10MB per log file
   java.util.logging.FileHandler.count = 10          # Rotate through 10 files
   java.util.logging.FileHandler.append = true
   java.util.logging.FileHandler.formatter =
   oracle.dbtools.logging.JSONLogFormatter
   ```

2. Before starting ORDS, use the `JDK_JAVA_OPTIONS` environment variable to specify the path to your `logging.properties` file.
   For example:

   ```
   export JDK_JAVA_OPTIONS="-Djava.util.logging.config.file=/path/to/
   logging.properties"
   ords serve
   ```

7.2.5.2 Create an agent configuration

This section explains how to create an agent configuration.

Set up a logging agent configuration to route log messages generated by ORDS to the OCI Logging service.

> **See Also**
>
> Creating an Agent Configuration

Copyright © 2011, 2026, Oracle and/or its affiliates.

8

Using the Multitenant Architecture with Oracle REST Data Services

This section outlines installing, configuring, upgrading and uninstalling Oracle REST Data Services in a multitenant container database.

- Setting Up ORDS in a CDB Environment
- Setting Up ORDS in an Application Container

> **Note**
>
> Oracle API for MongoDB is only supported for installations in a non-CDB Database or PDB. See Installing Oracle REST Data Services for more information on installing Oracle REST Data Services in a non-CDB Database or PDB.

Once ORDS metadata is saved in the database, it is necessary to provide a runtime configuration setting for ORDS to efficiently map requests to the corresponding pluggable databases.

- Setting Up ORDS in a CDB Environment
  This section describes how to setup Oracle REST Data Services (ORDS) into a multitenant container database (CDB) environment.
- Setting Up ORDS in an Application Container
  This section describes how to setup Oracle REST Data Services in an application container.
- Making All PDBs Addressable by Oracle REST Data Services (Pluggable Mapping)

> **See Also**
>
> Making All PDBs Addressable by Oracle REST Data Services (Pluggable Mapping)

8.1 Setting Up ORDS in a CDB Environment

This section describes how to setup Oracle REST Data Services (ORDS) into a multitenant container database (CDB) environment.

Oracle Database 12*c* Release 1 (12.1) introduced the multitenant architecture. This database architecture has a multitenant container database (CDB) that includes a root container, `CDB$ROOT`, a seed database, `PDB$SEED`, and multiple pluggable databases (PDBs). A PDB appears to users and applications as if it were a non-CDB. Each PDB is equivalent to a separate database instance in Oracle Database Release 11*g*.

The root container, `CDB$ROOT`, holds common objects that are accessible to every PDB utilizing metadata links or object links. The seed database, `PDB$SEED`, is used when you create a new

Copyright © 2011, 2026, Oracle and/or its affiliates.

PDB to seed the new pluggable database. The key benefit of the Oracle Database 12*c* multitenant architecture is that the database resources, such as CPU and memory, can be shared across all of the PDBs. This architecture also enables many databases to be treated as one for tasks such as upgrades or patches, and backups.

The installation process when you have multiple releases is described in the following section:

- Installation Enabling Multiple Releases

Preinstallation Tasks for Oracle REST Data Services CDB Installation

- Ensure that the PDBs are open (not mounted/closed) in read/write mode (except for `PDB$SEED`, which remains in read-only mode). For more information, see Oracle Multitenant Administrator's Guide
- Ensure that the default and temporary tablespaces to be used by the `ORDS_METADATA` schema and the `ORDS_PUBLIC_USER` user exist and that you know the tablespace names. The installation procedure creates those users, but it does not create the tablespaces.
- Ensure that when a common user is used for the PL/SQL Gateway, the user has also been explicitly created in the PDB$SEED container otherwise, the installer will not be able to configure the PL/SQL Gateway correctly.

> **Note**
>
> `ORDS_METADATA` and `ORDS_PUBLIC_USER` are installed in the seed container, and the default and temporary tablespaces exist in `PDB$SEED`. If these tablespaces do not already exist, then you must create the tablespaces in `PDB$SEED`. For more information, see Oracle Multitenant Administrator's Guide

- Installation Enabling Multiple Releases
  This section describes the installation process when you have multiple releases of Oracle REST Data Services and patch sets in the PDBs in a multitenant environment.
- Upgrading Oracle REST Data Services in a CDB Environment
- Uninstalling Oracle REST Data Services in a CDB Environment

8.1.1 Installation Enabling Multiple Releases

This section describes the installation process when you have multiple releases of Oracle REST Data Services and patch sets in the PDBs in a multitenant environment.

When Oracle REST Data Services is installed into a CDB, the proxy user, Oracle REST Data Services public user (`ORDS_PUBLIC_USER`) is installed in the root container and is a common user. The `ORDS_METADATA` schema is a local user that contains the metadata for Oracle REST Data Services. Both the `ORDS_METADATA` schema and the `ORDS_PUBLIC_USER` are installed in the seed container (`PDB$SEED`) and all of the pluggable databases.

Since the `ORDS_METADATA` is installed as a local user, this provides you the flexibility of installing multiple Oracle REST Data Services releases in the pluggable databases.

- Command Line Installation
- Interactive Installation
  This section describes the installation prompts for installing Oracle REST Data Services into a CDB to enable multiple Oracle REST Data Services releases.
- Silent Installation

Copyright © 2011, 2026, Oracle and/or its affiliates.

8.1.1.1 Command Line Installation

You must provide the `SYS AS SYSDBA` credentials in the Root (`CDB$ROOT`) container to perform the installation.

8.1.1.2 Interactive Installation

This section describes the installation prompts for installing Oracle REST Data Services into a CDB to enable multiple Oracle REST Data Services releases.

To install Oracle REST Data Services into a CDB to enable multiple Oracle REST Data Services releases, perform the following steps:

Enter the following command:

```
ords --config <CONFIG_FOLDER> install
```

You are prompted for necessary information to create or update the ORDS configuration files, install ORDS schema, ORDS database user, and related database objects in the database. When you are prompted for the connection information, ensure that you specify the connection information to your CDB.

```
Enter a number to select the database connection type to use
    [1] Basic (host name, port, service name)
    [2] TNS (TNS alias, TNS directory)
    [3] Custom database URL
  Choose [1]:
  Enter the database host name [localhost]: myhost
  Enter the database listen port [1521]:    1523
  Enter the database service name [orcl]:   mycdb
```

> **Note**
>
> To use the pluggable mapping feature, see Making All PDBs Addressable by Oracle REST Data Services (Pluggable Mapping) for more information.

> **See Also**
>
> Creating an Additional Database Pool

8.1.1.3 Silent Installation

To perform a silent installation, provide the options on the command-line. Ensure that you specify the connection information to your CDB.

Enter the following command:

```
ords --config <CONFIG_FOLDER> install <OPTIONS>
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Non-Interactive Command-Line Interface Installation (Silent)

8.1.2 Upgrading Oracle REST Data Services in a CDB Environment

When you use the latest release of Oracle REST Data Services (ORDS), and ORDS already exists in your database from an earlier ORDS release, then you can upgrade the ORDS schema in the CDB and its pluggable databases (PDBs) using the install command.

For example:

```
ords install --interactive --log-folder <LOG_FOLDER>
```

> **Note**
>
> If you are using the ORDS interactive install command to upgrade ORDS in the CDB, and if ORDS is already installed or upgraded to the current release, then you are not prompted for the information.

- Migrating Oracle REST Data Services in the CDB to Enable Multiple Releases
  This section describes how to migrate Oracle REST Data Services in the CDB to enable multiple releases.

8.1.2.1 Migrating Oracle REST Data Services in the CDB to Enable Multiple Releases

This section describes how to migrate Oracle REST Data Services in the CDB to enable multiple releases.

Starting with release 18.2.0 and later, if you have an Oracle REST Data Services schema and `ORDS_METADATA` that is installed in the `CDB$ROOT` container, then during upgrade it will migrate the common `ORDS_METADATA` schema to your PDBs as a local schema. Oracle database 12.1.0.2 and later releases support this change.

8.1.3 Uninstalling Oracle REST Data Services in a CDB Environment

To uninstall Oracle REST Data Services from a CDB, use the `uninstall` command.

For example:

```
ords uninstall --interactive --log-folder <LOG_FOLDER>
```

Oracle REST Data Services will be removed from the CDB and its pluggable databases (PDBs).

8.2 Setting Up ORDS in an Application Container

This section describes how to setup Oracle REST Data Services in an application container.

Starting with ORDS release 20.2.1, Oracle REST Data Services can be installed or upgraded into an application container using the ORDS SQL scripts provided in the `ords.version.number.zip` file.

Copyright © 2011, 2026, Oracle and/or its affiliates.

An application container consists of an application root where the application is defined and one or more PDBs that share the metadata about the application from the application root. You can have multiple application containers within a CDB and each container can have different versions of Oracle REST Data Services. Installing or upgrading Oracle REST Data Services in an application container is done against the application root container. When an application PDB wants to use the upgraded version, it must synchronize with the application root. Oracle REST Data Services continues to run in the application PDB with the existing version until the application PDB synchronizes with the application root.

Topics:

- Prerequisites for Creating ORDS in an Application Container
- Installing ORDS in the Application Root Container
- Creating an Application Seed
- Creating an Application PDB from the Application Seed
- ORDS Configuration Files Setup
- Running ORDS
- Upgrading ORDS in the Application Container
- Uninstalling ORDS from the Application Container
- Verifying ORDS in the Application Container
- Prerequisites for Creating ORDS in an Application Container
  This section describes the prerequisites for installing ORDS in an application container.
- Installing ORDS in the Application Root Container
  This section describes how to install ORDS in the application root container.
- Creating an Application Seed
  This section describes how to create an application seed.
- Creating an Application PDB from the Application Seed
  This section describes how to create an application PDB that is seeded from the application seed
- Configuring the PLSQL Gateway for APEX
  This section describes how to configure the PL/SQL Gateway to work with APEX, if required. The APEX_PUBLIC_USER must be configured to proxy to the ORDS_PUBLIC_USER to support ORDS RESTful Services from within APEX.
- ORDS Configuration Files Setup
  This section describes how to setup the ORDS configuration files:
- Running ORDS
  This section lists the different methods available to run ORDS after installing ORDS in the application container.
- Upgrading ORDS in the Application Container
  This section describes how to upgrade ORDS in the application container.
- Uninstalling ORDS from the Application Container
  This section describes how to uninstall ORDS from the application container.
- Verifying ORDS in the Application Container
  This section describes how to verify ORDS in the application container.

Copyright © 2011, 2026, Oracle and/or its affiliates.

8.2.1 Prerequisites for Creating ORDS in an Application Container

This section describes the prerequisites for installing ORDS in an application container.

Following prerequisites must be met before you install ORDS in an application container:

- Download ORDS version 20.2.1 or later from Oracle REST Data Services Downloads.
- Extract the ORDS SQL scripts.
- To obtain the ORDS SQL scripts, execute the following commands:

```
unzip ords.version.number.zip ords.war
unzip ords.war 'WEB-INF/lib/ords-installer-*.jar'
unzip  'WEB-INF/lib/ords-installer-*.jar' 'db/*'
mv db scripts
```

The ORDS SQL scripts are located in the scripts folder. The scripts folder contains the subdirectories for the install, upgrade, validate, and uninstall SQL scripts. You can run these SQL scripts using SQLcl, SQL*Plus, or SQL Developer.

- Creating an Application Root Container
  This section describes how to create an application root container.

8.2.1.1 Creating an Application Root Container

This section describes how to create an application root container.

To create an application root container:

1. Ensure that the current container is in `CDB$ROOT`.
2. Use the AS APPLICATION CONTAINER clause of the `CREATE PLUGGABLE DATABASE` statement to create an application container.
3. Open the application container.

Example:

```
CREATE PLUGGABLE DATABASE ords_app_root1 AS APPLICATION CONTAINER ADMIN USER
admin IDENTIFIED BY <admin_password>

FILE_NAME_CONVERT=('pdbseed', 'ords_app_root1');

ALTER PLUGGABLE DATABASE ords_app_root1 OPEN;
```

> **Note**
>
> `ords_app_root1` and the `admin` user in the preceding example can be any valid Oracle identifier.
> If Oracle managed files is enabled in the CDB or the `PDB_FILE_NAME_CONVERT` initialization parameter is set, then omit the `FILE_NAME_CONVERT` clause.
>
> The ORDS users, namely `ORDS_PUBLIC_USER` and `ORDS_METADATA`, must not exist in the seed (for example, `pdbseed`) or cloned pdb.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Creating an Application Container

8.2.2 Installing ORDS in the Application Root Container

This section describes how to install ORDS in the application root container.

To install ORDS in the application root container, perform the following steps:

1. Connect to the application root container.
2. Run `/path/to/scripts/install/core/ords_app_con_install.sql` command using the following parameters:
   - Log folder (must include the forward slash at the end)
   - Default tablespace for ORDS schema
   - Temporary tablespace for ORDS schema
   - Default tablespace for ORDS proxy user
   - Temporary tablespace for ORDS proxy user
   - ORDS proxy user password
   - Scripts path (requires the fully qualified path to the ORDS scripts)

> **Note**
>
> The tablespaces must already exist in the database.

```
ALTER SESSION SET CONTAINER = ords_app_root1;

@/path/to/scripts/install/core/ords_app_con_install.sql /path/to/logs/ SYSAUX
TEMP SYSAUX TEMP <PASSWORD> /path/to/scripts
```

Where:

The `ords_app_con_install.sql` creates an application named ORDS and assigns the application version to the ORDS product version. The product version format is Year.Quarter.Patch.rJulianDay24HRMM (for example, 20.2.1.r2121800).

The preceding script installs ORDS and creates the following:

- The ORDS schema, `ORDS_METADATA`
- The ORDS proxy user, `ORDS_PUBLIC_USER` and
- The related database objects in the application container

> **See Also**
>
> Verifying ORDS in the Application Container

Copyright © 2011, 2026, Oracle and/or its affiliates.

8.2.3 Creating an Application Seed

This section describes how to create an application seed.

An application seed is used to provision application PDBs with the application root's applications pre-installed.

To create an application seed:

1. Ensure that the current container is in the `CDB$ROOT`.
2. Alter session and set container to the application root.
3. Use the AS SEED clause of the `CREATE PLUGGABLE DATABASE` statement to create an application seed.
4. Sync the ORDS application with the application seed.
5. Compile invalid objects.
6. Open the application seed in a read only mode.

> **Note**
>
> `ords_app_root1` and the `admin` user in the following example can be any valid Oracle identifier.
> If Oracle managed files is enabled in the CDB or the `PDB_FILE_NAME_CONVERT` initialization parameter is set, then omit the `FILE_NAME_CONVERT` clause.

```
ALTER SESSION SET CONTAINER = ords_app_root1;
CREATE PLUGGABLE DATABASE AS SEED ADMIN USER admin IDENTIFIED BY
<admin_password>
FILE_NAME_CONVERT=('pdbseed', 'ords_app_root1_seed');
ALTER PLUGGABLE DATABASE ords_app_root1$seed open;
ALTER SESSION SET CONTAINER = ords_app_root1$seed;
ALTER PLUGGABLE DATABASE application ORDS sync;
begin
  sys.dbms_utility.compile_schema('ORDS_METADATA', FALSE);
end;
/
ALTER PLUGGABLE DATABASE ords_app_root1$seed close immediate;
ALTER PLUGGABLE DATABASE ords_app_root1$seed open read only;
```

> **See Also**
>
> Creating an Application Container

Copyright © 2011, 2026, Oracle and/or its affiliates.

8.2.4 Creating an Application PDB from the Application Seed

This section describes how to create an application PDB that is seeded from the application seed

An application PDB is created by issuing the `CREATE PLUGGABLE DATABASE` statement from the application root.

To create an application PDB from the application seed:

1. Ensure that the current container is in `CDB$ROOT`.
2. Alter session and set the container to the application root.
3. Use the `CREATE PLUGGABLE DATABASE` command to create a PDB from the application seed.

> **Note**
>
> `ords_app_pdb1` and the `admin` user in the following example can be any valid Oracle identifier.
> If Oracle managed files is enabled in the CDB or the `PDB_FILE_NAME_CONVERT` initialization parameter is set, then omit the `FILE_NAME_CONVERT` clause.

```
ALTER SESSION SET CONTAINER=ords_app_root1;
CREATE PLUGGABLE DATABASE ords_app_pdb1 ADMIN USER admin IDENTIFIED BY <admin
password>
FILE_NAME_CONVERT=('ords_app_root1_seed','ords_app_pdb1');
ALTER PLUGGABLE DATABASE ords_app_pdb1 OPEN;
ALTER SESSION SET CONTAINER = ords_app_pdb1;
select app_name, app_version, app_status from dba_applications where app_name
= 'ORDS';
APP_NAME     APP_VERSION         APP_STATUS
--------     ---------------     ------------
ORDS         20.2.1.r2121800     NORMAL
```

> **See Also**
>
> Creating an Application Container

8.2.5 Configuring the PLSQL Gateway for APEX

This section describes how to configure the PL/SQL Gateway to work with APEX, if required. The APEX_PUBLIC_USER must be configured to proxy to the ORDS_PUBLIC_USER to support ORDS RESTful Services from within APEX.

To configure the PL/SQL Gateway:

- Ensure that the current container is in `CDB$ROOT`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Alter session and set the container to the application pdb, example: `ords_app_pdb1`.

```
ALTER SESSION SET CONTAINER = ords_app_pdb1;
ALTER USER APEX_PUBLIC_USER GRANT CONNECT THROUGH ORDS_PUBLIC_USER;
begin
  ords_admin.config_plsql_gateway(
      p_runtime_user       => 'ORDS_PUBLIC_USER',
      p_plsql_gateway_user => 'APEX_PUBLIC_USER',
      p_comments           => 'APEX proxied to ORDS');
  commit;
end;
/
```

Using ORDS CLI command, add the additional setting:

```
ords --config <configuration folder> config set plsql.gateway.mode proxied
```

8.2.6 ORDS Configuration Files Setup

This section describes how to setup the ORDS configuration files:

Topics:

- Specifying the ORDS Configuration Folder
- Creating the ORDS Configuration Files for the Application Root Container

- Specifying the ORDS Configuration Folder
  This section describes how to specify the ORDS configuration folder.
- Creating the ORDS Configuration Files for the Application Root Container
  This section describes how to create the ORDS configuration files for the application root container.

8.2.6.1 Specifying the ORDS Configuration Folder

This section describes how to specify the ORDS configuration folder.

You must provide your configuration folder prior to creating your configuration. Refer to Setting Up the Configuration Folder Structure section for setting-up the folder structure.

8.2.6.2 Creating the ORDS Configuration Files for the Application Root Container

This section describes how to create the ORDS configuration files for the application root container.

To create the ORDS configuration files for the application root container, execute the following install command to create the configuration files:

```
ords --config <configuration folder> install --interactive --config-only --log-folder <Log Folder>
```

Where the `--config-only` option must be specified to create the configuration files. There are a list of options displaying the current values, you can change the values. When this list of options are prompted for the service name, specify the application root servicename.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example 8-1 Creating ORDS Configuration

You are prompted for the necessary information to create a database pool in your ORDS configuration files. When the multiple options prompts are displayed, you must specify the same password that you used for the proxy user, `ORDS_PUBLIC_USER`.

> **See Also**
>
> - Installing ORDS in the Application Root Container
> - Multiple Options for Database Pool and Standalone Configuration

> **Note**
>
> If Oracle APEX is installed in your database, then specify the APEX images location. Example:
>
> APEX static resources location: `/path/to/apex/images`.
>
> Oracle recommends to use HTTPS protocol instead of HTTP.

Example 8-2 Using a Basic Connection

```
Oracle REST Data Services - Interactive Install
  Enter a number to select the database connection type to use
    [1] Basic (host name, port, service name)
    [2] TNS (TNS alias, TNS directory)
    [3] Custom database URL
  Choose [1]:
  Enter the database host name [localhost]:
  Enter the database listen port [1521]:
  Enter the database service name [orcl]: ords_app_root1

Specify the Application Root servicename.

  Enter a number to update the value or select option A to Accept and Continue
    [1] Connection Type: Basic
    [2] Basic Connection: HOST=localhost PORT=1521 SERVICE_NAME=ords_app_root1
    [3] Database password for ORDS runtime user (ORDS_PUBLIC_USER): <generate>
    [4] Additional Feature: Database Actions
    [5] Configure for Standalone Mode: Yes
    [6]    Protocol: HTTP
    [7]       HTTP Port: 8080
    [8]   APEX static resources location:
    [A] Accept and Continue - Create database pool
    [Q] Quit - Do not proceed. No changes
  Choose [A]: 3

Select option 3 to update the ORDS_PUBLIC_USER password.

  Enter your choice for the runtime user password (ORDS_PUBLIC_USER)
    [S] Specify your own password
    [G] Generate password
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
    [C] Cancel - No Changes
  Choose [G]: S

Select option S to specify the password.

  Enter the database password for ORDS_PUBLIC_USER:
Confirm password:

Enter the same password you specified for the ORDS_PUBLIC_USER in section
Installing ORDS in the Application Root
    Container.

Enter a number to update the value or select option A to Accept and Continue
    [1] Connection Type: Basic
    [2] Basic Connection: HOST=localhost PORT=1521 SERVICE_NAME=ords_app_root1
    [3] Database password for ORDS runtime user (ORDS_PUBLIC_USER):
<specified password>
    [4] Additional Feature: Database Actions
    [5] Configure for Standalone Mode: Yes
    [6]    Protocol: HTTP
    [7]       HTTP Port: 8080
    [8]   APEX static resources location:
    [A] Accept and Continue - Create database pool
    [Q] Quit - Do not proceed. No changes
  Choose [A]:

Make any additional changes.

  Enter a number to update the value or select option A to Accept and Continue
    [1] Connection Type: Basic
    [2] Basic Connection: HOST=localhost PORT=1521 SERVICE_NAME=ords_app_root1
    [3] Database password for ORDS runtime user (ORDS_PUBLIC_USER):
<specified password>
    [4] Additional Feature: Database Actions
    [5] Configure for Standalone Mode: Yes
    [6]    Protocol: HTTPS
    [7]       HTTPS Port: 8443
    [8]    Certificate Type: Use my SSL Certificate
    [9]      Path for the SSL Certificate: /path/to/mycert.crt
    [10]      Path for the SSL Certificate private key: /path/to/mycert.der
    [11]   APEX static resources location: /path/to/apex/images
    [A] Accept and Continue - Create database pool
    [Q] Quit - Do not proceed. No changes
  Choose [A]:

Select option A to create your database pool.
```

8.2.7 Running ORDS

This section lists the different methods available to run ORDS after installing ORDS in the application container.

Once you install ORDS in the application container and create the ORDS configuration files, run ORDS using one of the following methods:

- Standalone Mode

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Deploy on Oracle WebLogic Server
- Deploy Oracle REST Data Services on Apache Tomcat

> **See Also**
>
> Deploying Oracle REST Data Services

8.2.8 Upgrading ORDS in the Application Container

This section describes how to upgrade ORDS in the application container.

You can upgrade an application in an application container. These operations are performed in the application root. The application container propagates the application changes to the application PDBs when the application PDBs synchronize with the application in the application root.

Prerequisites:

- ORDS must already be installed in the application container.
- Upgrading ORDS from an earlier release to a new release.

To upgrade ORDS in the application root:

1. In SQLcl or SQL*Plus, connect to the application root.
2. Run `/path/to/scripts/upgrade/ords_app_con_upgrade.sql` with the following parameters:
   - Log folder (must include the forward slash at the end)
   - Scripts path (requires the fully qualified path to the ORDS scripts)

> **Note**
>
> The `ords_app_con_upgrade.sql` script upgrades ORDS in the application root container to the release that you are using. For example, if the ORDS application version is 20.2.1.r2121800, and the ORDS upgrade script is 20.3.0.r2601900, then the script upgrades ORDS to release 20.3.0.r2601900 in the application root container.

To synchronize the ORDS application in an application PDB with the upgrade changes in the application root:

1. In SQLcl or SQL*Plus, ensure that the current container is the application PDB.
2. Run the `ALTER PLUGGABLE DATABASE APPLICATION` statement specifying the ORDS application with the SYNC clause.

```
ALTER SESSION SET CONTAINER = ords_app_pdb1;

ALTER PLUGGABLE DATABASE APPLICATION ORDS SYNC;
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Verifying ORDS in the Application Container

8.2.9 Uninstalling ORDS from the Application Container

This section describes how to uninstall ORDS from the application container.

You can uninstall an application from an application container. These operations are performed in the application root. The application container propagates the application changes to the application PDBs when the application PDBs synchronize with the application in the application root.

To uninstall ORDS from the application root:

1. In SQLcl or SQL*Plus, connect to the application root.
2. Run `/path/to/scripts/uninstall/core/ords_app_con_uninstall.sql` with the following parameters:
   - Log folder (must include the forward slash at the end)
   - Scripts path (requires the fully qualified path to the ORDS scripts)

```
ALTER SESSION SET CONTAINER = ords_app_root1;

@/path/to/scripts/uninstall/core/ords_app_con_uninstall.sql /path/to/logs/ /
path/to/scripts
```

To synchronize the application PDB to uninstall the ORDS application:

1. In SQLcl or SQL*Plus, ensure that the current container is the application PDB.
2. Run the `ALTER PLUGGABLE DATABASE APPLICATION` statement specifying the ORDS application with the SYNC clause.

For example:

```
ALTER SESSION SET CONTAINER = ords_app_pdb1;

ALTER PLUGGABLE DATABASE APPLICATION ORDS SYNC;
```

> **See Also**
>
> Verifying ORDS in the Application Container

8.2.10 Verifying ORDS in the Application Container

This section describes how to verify ORDS in the application container.

To verify the ORDS for install, upgrade, validate, and uninstall in the application container:

- Manually inspect the following log files for any errors:
  - `Install - ordsinstall_<timestamp>.log`
  - `Upgrade - ordsupgrade_<timestamp>.log`
  - `Uninstall - ordsuninstall_<timestamp>.log`

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Query `dba_applications` to verify if the ORDS application exists and its application version is the same as the ORDS product version.

  ```
  SQL> select app_name, app_version, app_status from dba_applications where
  app_name = 'ORDS';

      APP_NAME     APP_VERSION         APP_STATUS
      --------     ---------------     ------------
      ORDS         20.2.1.r2121800     NORMAL
  ```

- Query `dba_app_errors` to check for any errors:
  ```
  SQL> select app_name, app_statement, errornum, errormsg from dba_app_errors
  where app_name = 'ORDS';

  no rows selected
  ```

  If you are uninstalling ORDS from the application container, the `APP_STATUS` contains the value `UNINSTALLED`.

8.3 Making All PDBs Addressable by Oracle REST Data Services (Pluggable Mapping)

Pluggable mapping refers to the ability to make all PDBs in a CDB root or in an application root container addressable by Oracle REST Data Services. To use this feature, follow the instructions described in this topic.

If the Oracle REST Data Services configuration file includes the `db.serviceNameSuffix` parameter, this indicates that the Oracle REST Data Services pool points to a CDB root or an application root, and that the PDBs connected to that CDB root or an application root should be made addressable by Oracle REST Data Services. For Pluggable mapping to work, you must specify this parameter setting. You can specify the `db.serviceNameSuffix` parameter at a pool-specific level or at a global level. If you set the parameter to a global level, then you can apply it to all the database pool configurations that define a CDB connection. Also, the Database Actions landing page (`<server-name>:<port-number>/ords/sql-developer`) displays a prompt for the PDB name, if you set the parameter to a global level.

The value of the `db.serviceNameSuffix` parameter must correspond to the value of the DB_DOMAIN database initialization parameter, based on the following rules:

- If the DB_DOMAIN value is blank, then the `db.serviceNameSuffix` parameter must also be an empty string.
- If the DB_DOMAIN value is not blank, then the `db.serviceNameSuffix` parameter must have a value starting with a period (.).

To set the value of the `db.serviceNameSuffix` parameter:

1. In SQLcl or SQL*Plus, connect to the root as a user with SYSDBA privileges.
2. Check the value of the `DB_DOMAIN` database initialization parameter.

   ```
   SQL> show parameter DB_DOMAIN
   ```

3. Exit SQLcl or SQL*Plus.

   ```
   SQL> exit
   ```

4. If the `db.serviceNameSuffix` parameter value is not defined, enter a command in the following format to add an entry to the configuration file:

Copyright © 2011, 2026, Oracle and/or its affiliates.

```
ords --config /path/to/config config --db-pool pool-name set "db.serviceNameSuffix"
".value-of-DB_DOMAIN"
```

Where `pool-name` is one of the following:

- `default` for the default and migrated pool
- <*custom_pool*> as specified during ORDS install

Example 1: You want to make PDBs in a CDB root or an application root addressable globally. Specify global settings by entering the following command:

```
ords --config /path/to/config config set --global "db.serviceNameSuffix" ".value-of-
DB_DOMAIN"
```

> **Note**
>
> The approach shown in Example 1, that is, setting the property for all the pools through the `global/settings.xml` file, works the best for most use cases.

Example 2: You want to make PDBs in a CDB root or an application root addressable for your PL/SQL Gateway, and your pool name is `my_pool`. Enter the following command:

```
ords --config /path/to/conf config --db-pool my_pool set "db.serviceNameSuffix"
".example.com"
```

For example, if the database pointed to by apex has a `DB_DOMAIN` value of `example.com` and contains the two PDBs `pdb1.example.com` and `pdb2.example.com`, the first PDB will be mapped to URLs whose path starts with `/ords/pdb1/`, and the second PDB will be mapped to URLs whose path starts with `/ords/pdb2/`.

Example 3: You want to make PDBs in a CDB root or an application root addressable globally and the DB_DOMAIN is an empty value. Enter the following command to specify `db.serviceNameSuffix` parameter globally:

```
ords --config /path/to/config config set --global "db.serviceNameSuffix" ""
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

9

Installing and Configuring Customer Managed ORDS on Autonomous Database

This section explains how to install and configure Customer Managed Oracle REST Data Services (ORDS) on Autonomous Database.

- About Customer Managed Oracle REST Data Services on Autonomous Database
  When you provision an Autonomous Database instance, by default Oracle REST Data Services (ORDS) is preconfigured and available for the instance. With the default ORDS, Oracle performs any required configuration, patching, and maintenance. Additionally, you can also configure Autonomous Database to use ORDS running in a customer managed environment.
- Preinstallation Tasks
  This section describes the preinstallation tasks.
- ORDS Command-Line Interface for Customer Managed ORDS

9.1 About Customer Managed Oracle REST Data Services on Autonomous Database

When you provision an Autonomous Database instance, by default Oracle REST Data Services (ORDS) is preconfigured and available for the instance. With the default ORDS, Oracle performs any required configuration, patching, and maintenance. Additionally, you can also configure Autonomous Database to use ORDS running in a customer managed environment.

When you use the default ORDS on Autonomous Database, you cannot modify any of the ORDS configuration options. Use a customer managed environment if you want manual control of the configuration and management of Oracle REST Data Services. For example, use this option when your applications require larger connection pools or if you need more control over the ORDS configuration options.

When ORDS runs in a customer managed environment, you are responsible for configuration, patching, and maintenance of ORDS in the customer managed environment. After you configure Autonomous Database to use your customer managed ORDS in addition to the existing autonomously managed ORDS, you can route ORDS HTTPS traffic through your environment. The default Autonomous Database web server and ORDS are still running and ORDS traffic goes to the ORDS running in the customer managed environment. This provides an additional and alternative HTTPS solution for Autonomous Database.

Installing and configuring a customer managed environment for ORDS allows you to run ORDS with configuration options that are not possible using the default Oracle managed ORDS available with Autonomous Database.

Installing and configuring a customer managed environment for ORDS is only supported with Autonomous Database.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> - A version of Oracle REST Data Services within one major release of what is running in Autonomous Database (Serverless or Dedicated) is required to use a customer managed environment for ORDS with a version closest to that release. **For example**:
>
> If your Autonomous Database is currently hosting ORDS version 22.4, then you must be running the latest possible ORDS version with customer managed environment. Maintain the latest version of ORDS as close as possible, for example 23.1 or 22.3 for the best results.
>
> > **Note**
> >
> > The latest version of ORDS can be found in the OCI YUM repository and also at ORDS Downloadable Zip File location.

> **Note**
>
> Web servers should be installed on dedicated systems that are separate from the database servers.

> **See Also**
>
> - Oracle REST Data Services (ORDS) best practices
> - Interactive Command-Line Interface Installation and refer to section *HTTPS Protocol and Use my SSL Certificate* for more information and
> - Adding a Signed SSL Certificate to Oracle REST Data Services

9.2 Preinstallation Tasks

This section describes the preinstallation tasks.

Before performing a customer-managed ORDS installation, you must:

- Download the wallet from your Oracle Autonomous AI Database instance
- If you are using Oracle APEX, then setup the APEX static resources

- Downloading the Autonomous AI Database Wallet
  This section describes how to download client credentials (Wallets) from the Oracle Cloud Infrastructure console.
- Oracle APEX Static Resources
  This section describes how to set up the APEX static resources.

Copyright © 2011, 2026, Oracle and/or its affiliates.

9.2.1 Downloading the Autonomous AI Database Wallet

This section describes how to download client credentials (Wallets) from the Oracle Cloud Infrastructure console.

Customer managed ORDS instances for the Autonomous AI Database requires the Autonomous AI Database wallet for the target database. For detailed instructions, see Download Client Credentials (Wallets).

9.2.2 Oracle APEX Static Resources

This section describes how to set up the APEX static resources.

If you are using ORDS with APEX, then setting up the APEX static resources is mandatory.

> **See Also**
>
> Managing Static Resources (Images)

9.3 ORDS Command-Line Interface for Customer Managed ORDS

The ORDS Command-Line Interface (CLI) provides the interactive and silent command `install adb` to automate configuring a Customer Managed ORDS. This includes creating the ORDS configuration in your environment. If you want to use Autonomous Database with Oracle REST Data Services (ORDS) running in a customer managed environment, execute this command. This creates an ORDS runtime database user. ORDS can connect, and provide the privileges to that runtime user. In addition, it creates and provides privileges to the PL/SQL gateway database user used for APEX, PL/SQL Gateway and OWA, and allow connections through the runtime user. The runtime database user and gateway database user are created in the Autonomous Database.

- Interactive Install for ADB Command Line Interface
- Silent Installation of ADB on Command-Line Interface

9.3.1 Interactive Install for ADB Command Line Interface

Use the ORDS Command-Line Interface (CLI) to interactively prompt you for the following information to setup your Customer Managed ORDS.

- Wallet Path
- Net Service Name from tnsnames.ora contained in the wallet zip file
- Administrator user
- Runtime database user
- PL/SQL gateway user
- Additional Database Features
- Standalone options

Copyright © 2011, 2026, Oracle and/or its affiliates.

Examples:

- `ords install adb`
- `ords install adb --interactive [OPTIONS]`
- `ords install adb -i [0PTIONS]`
- Customer Managed ORDS Command Options
- Interactive Installation Prompts
  This section describes the interactive installation prompts to setup your Customer Managed ORDS.

9.3.1.1 Customer Managed ORDS Command Options

Option	Description
`admin-user <DATABASE USER>`	The administrator database user with privileges to create users and grant privileges to database users in the Autonomous Database.
`db-pool <POOL NAME>`	The name of the database connection pool.
`db-user <DATABASE USER>`	The ORDS runtime database user.
`gateway-user <DATABASE USER>`	The PLSQL gateway database user that has privileges to access the stored procedures.
`feature-db-api <BOOLEAN>`	Specifies if you want to enable DB API feature. Possible values are `true` or `false`. • If the value is set to `true`, then DB API feature is enabled. • If the value is set to `false`, then DB API feature is disabled. Returns an error if the specified options are `--feature-sdw true` and `--feature db-api false`.
`feature-rest-enabled-sql <BOOLEAN>`	Specifies if you want to enable REST-Enabled SQL feature. Possible values are `true` or `false`. • If the value is set to `true`, then the REST-Enabled SQL feature is enabled. • If the value is set to `false`, then the REST-Enabled SQL feature is disabled. Returns an error if the specified options `are --feature-sdw true` and `--feature-rest-enabled-sql false`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Option	Description
`feature-sdw <BOOLEAN>`	Specifies if you want to enable Database Actions feature. Possible values are true or false. • If the value is set to true, then the Database Actions feature is enabled. • If the value is set to false, then the Database Actions feature is disabled. If the option is set to true, then the following settings are set to true in the configuration file: • `database.api.enabled` • `restEnabledSql.active` Returns an error if `--feature-sdw true` and any of following options are specified, and are set to false: • `--feature-db-api` • `--feature-rest-enabled-sql`
`-h, --help`	Shows usage information for a command.
`-i, --interactive`	Prompts for the required information.
`log-folder <FOLDER>`	Writes the logs in the folder when creating the users and granting privileges to the user. If this option is omitted, then the output is written to standard output.
`password-stdin`	Reads the password value from standard input when redirecting input to a file or here document.
`wallet <PATH>`	The location of the wallet zip file downloaded from Autonomous Database. Returns an error if the wallet is omitted and the `db.wallet.zip.path` setting does not exist in the ORDS configuration.
`wallet-service-name <NET SERVICE NAME>`	Specifies the net service name in the `tnsnames.ora` file located in the wallet zip file. If the `--wallet-service-name` option is omitted, or if the `db.wallet.zip.service` property is not detected in the ORDS global configuration settings (that is, in the `global.xml` file), then ORDS defaults to using the `<db>_LOW` Consumer Group. This value is retrieved from the `tnsnames.ora file`.

9.3.1.2 Interactive Installation Prompts

This section describes the interactive installation prompts to setup your Customer Managed ORDS.

To setup your Customer Managed ORDS, use the ORDS Command-Line Interface (CLI) to interactively prompt you for the information.

Example:

```
ords install adb --interactive --prompt-password --log-folder <LOG
    FOLDER>
```

Where:

Copyright © 2011, 2026, Oracle and/or its affiliates.

- `--prompt-password`: prompt you for the runtime database user's password and the gateway database user's passwords
- `--prompt-password`: If this option is omitted, then the passwords are generated. Special care should be considered for database user's password. If you plan to use ORDS on multiple servers and use the same runtime database user and gateway database user, then specify the `--prompt-password` option to ensure that the same passwords are being used.

Table 9-1 Interactive Installation Prompts

Prompt Number	Prompt	Description

1.	`Enter a number to select the database pool to update, or create an additional database pool.`
	`  The selected (or created) database pool will be used to configure a Customer Managed ORDS.`
	`  [1] default  MYADB_MEDIUM  /path/to/myadb/wallet.zip`
	`  [2] Create an additional database pool`
	`  Choose [1]:`
		Refer to Entering a Number to Select the Database Pool

2.	`Enter the database pool name.`	Refer to Entering the Database Pool Name
3.	`Enter the Autonomous Database Wallet path: /path/to/wallet.zip`	Refer to Entering the Autonomous Database Wallet Path

4.	`Enter a number to select the TNS Network alias to use`
	`  [1] DEMO_LOW  ...service_name=g123_demo_low.adb.oraclecloud....`
	`  [2] DEMO_MEDIUM  ...service_name=g123_demo_medium.adb.oracleclo...`
	`  [3] DEMO_HIGH  ...service_name=g123_demo_high.adb.oraclecloud...`
	`  Choose [1]:`
		Refer to Enter a Number to Select the TNS Network Alias

5.	`Provide database user name with administrator privileges.` `  Enter the administrator username [ADMIN]:`	Enter the Administrator Username
6.	`Enter the database password for ADMIN:`	Enter the Database Password for ADMIN

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 9-1 (Cont.) Interactive Installation Prompts

Prompt Number	Prompt	Description
7.	`Enter the ORDS runtime database username [ORDS_PUBLIC_USER2]:`	Entering the ORDS Runtime Database Username
8.	`Enter the database password for ORDS_PUBLIC_USER2:` `Confirm password:`	Entering the Database Password for ORDS_PUBLIC_USER2
9.	`Enter the PL/SQL Gateway database username:`	Entering the PL/SQL Gateway Database Username
10.	`Enter the database password for ORDS_PLSQL_GATEWAY2:` `Confirm password:`	Enter the Database Password for ORDS_PLSQL_GATEWAY2

11.	`Connecting to Autonomous database user: ADMIN`
	`TNS Service: DEMO_MEDIUM`
	`Retrieving information`	

12.	`Enter a number to select additional feature(s) to enable:`
	`[1] Database Actions (Enables all features)`
	`[2] REST Enabled SQL and Database API`
	`[3] REST Enabled SQL`
	`[4] Database API`
	`[5] None`
	`Choose [1]:`
		Entering a Number to Select and Enable Additional Feature

13.	`Enter a number to configure and start ORDS in standalone mode`
	`[1] Configure and start ORDS in standalone mode`
	`[2] Skip`
	`Choose [1]:`
		Enter a Number to Configure and Start ORDS

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 9-1 (Cont.) Interactive Installation Prompts

Prompt Number	Prompt	Description

14.	`Enter a number to select the protocol`
	`[1] HTTP`
	`[2] HTTPS`
	`Choose [1]: 2`
		Entering a Number to Use HTTP or HTTPS Protocol

15.	`Enter the APEX static resources location:` `/path/to/apex/images`	Entering the APEX Static Resources Location

- Entering a Number to Select the Database Pool
- Entering the Database Pool Name
- Entering the Autonomous Database Wallet Path
- Enter a Number to Select the TNS Network Alias
- Enter the Administrator Username
- Enter the Database Password for ADMIN
- Entering the ORDS Runtime Database Username
- Entering the Database Password for ORDS_PUBLIC_USER2
- Entering the PL/SQL Gateway Database Username
- Enter the Database Password for ORDS_PLSQL_GATEWAY2
- Entering a Number to Select and Enable Additional Feature
- Enter a Number to Configure and Start ORDS
- Entering a Number to Use HTTP or HTTPS Protocol
- Entering the HTTPS Port
- Entering a Number to Select the Certificate Type
- HTTPS Protocol and Use Self-Signed Certificate
- HTTPS Protocol and Use my SSL Certificate
- Entering the APEX Static Resources Location

9.3.1.2.1 Entering a Number to Select the Database Pool

This prompt is displayed, if an ORDS configuration exists and contains database pool(s).

You can select a database pool to update, or create an additional database pool for your Customer Managed ORDS.

If option 2 is selected, then Prompt number 2 is displayed. Otherwise, Prompt number 3 is displayed.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> If this is the first time you are setting up the Customer Managed ORDS, and the ORDS configuration does not exist, then you are prompted for the wallet location. See Prompt number 3.

9.3.1.2.2 Entering the Database Pool Name

Specify the database pool name.

9.3.1.2.3 Entering the Autonomous Database Wallet Path

Specify the location and filename of the downloaded Autonomous Database wallet.

> **Note**
>
> If this is the first time you are setting up the Customer Managed ORDS, then you are prompted for the wallet location. Otherwise, if an ORDS configuration already exists, then Prompt number 1 is displayed.

9.3.1.2.4 Enter a Number to Select the TNS Network Alias

Select the TNS alias name.

> **Note**
>
> The list of TNS alias names are in the `tnsnames.ora` file located in the `wallet.zip` file.

> **See Also**
>
> Local Naming Parameters in the tnsnames.ora File

9.3.1.2.5 Enter the Administrator Username

Specify a database user with administrator privileges. Defaults the database user to ADMIN.

9.3.1.2.6 Enter the Database Password for ADMIN

Specify the password for administrator database user.

9.3.1.2.7 Entering the ORDS Runtime Database Username

Specify the ORDS runtime database user. Defaults the database user to `ORDS_PUBLIC_USER2`.

9.3.1.2.8 Entering the Database Password for ORDS_PUBLIC_USER2

Prompts for the password if `--prompt-password` option is specified on the command line. Otherwise, the password prompt is not displayed, and the password is generated.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> If the runtime database user does not exist in the Autonomous Database, then the runtime database user is created and granted privileges. If the runtime database user already exists in the Autonomous Database, and the runtime user's password does not match the password in the Autonomous Database, then the runtime database user password is changed.

9.3.1.2.9 Entering the PL/SQL Gateway Database Username

Specify the ORDS PL/SQL gateway database user.

9.3.1.2.10 Enter the Database Password for ORDS_PLSQL_GATEWAY2

Prompts for the password if `--prompt-password` option is specified on the command line. Otherwise, the password prompt is not displayed, and the password is generated.

> **Note**
>
> If the PL/SQL gateway database user does not exist in the Autonomous Database, then the gateway database user is created and granted privileges. If the gateway user already exists in the Autonomous Database, and the gateway user's password does not match the password in the Autonomous Database, then the gateway database user password is changed..

9.3.1.2.11 Entering a Number to Select and Enable Additional Feature

Select the additional feature that you want to enable.

> **See Also**
>
> - About Oracle Database Actions
> - REST- Enabled SQL Service
> - ORDS Database API

9.3.1.2.12 Enter a Number to Configure and Start ORDS

You can configure ORDS to run in standalone mode. In addition, you can start ORDS in standalone mode after setup is completed for Customer Managed ORDS.

9.3.1.2.13 Entering a Number to Use HTTP or HTTPS Protocol

Displays the HTTP/HTTPS Protocol prompts. Select the protocol (HTTP or HTTPS) that you want to use.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> Oracle highly recommends to run ORDS in a standalone mode with HTTPS versus HTTP.

- Entering the HTTP Port

9.3.1.2.13.1 Entering the HTTP Port

Specify the HTTP port. The default HTTP port is 8080.

9.3.1.2.14 Entering the HTTPS Port

Specify the HTTPS port. The default HTTPS port is 8443.

Table 9-2 HTTPS Protocol Prompts

Prompt	Description

`Enter a number to select the protocol`
`[1] HTTP`
`[2] HTTPS`
`Choose [1]:`
	Select the protocol (HTTP or HTTPS) that you want to use.

`Enter the HTTPS port [8443]:`	Specify the HTTPS port. The default HTTPS port is 8443.

`Enter a number to select the certificate type`
`[1] Use self-signed certificate (generates automatically)`
`[2] Use my SSL certificate (requires SSL certificate and SSL certificate private key)`
`Choose [1]: 1`
	Refer to:
	• Entering a Number to Select the Certificate Type
	• HTTPS Protocol and Use Self-Signed Certificate
	• HTTPS Protocol and Use my SSL Certificate

9.3.1.2.15 Entering a Number to Select the Certificate Type

You have the option to select the self-signed certificate that generates automatically when running ORDS in standalone mode, or you provide the SSL certificate which requires both the SSL Certificate and its private key.

9.3.1.2.16 HTTPS Protocol and Use Self-Signed Certificate

Selecting the prompt `Use self-signed certificate` prompts you for the SSL hostname.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 9-3 Example of Use Self-Signed Certificate

Prompt
Enter a number to select the certificate type [1] Use self-signed certificate (generates automatically) [2] Use my SSL certificate (requires SSL certificate and SSL certificate private key) Choose [1]: 1 Enter the SSL hostname: myhost

ORDS generates the Self-Signed Certificate when running ORDS in a standalone mode.

The value entered here is stored in the `standalone.https.host` configuration setting. The Self-Signed Certificate is generated only once and the Common Name value for that certificate is taken from the `standalone.https.host` configuration setting. Changing the `standalone.https.host` after the Self-Signed Certificate is generated does not change the Common Name in that certificate.

Table 9-4 HTTPS Protocol and Use Self-Signed Certificate Prompts

Prompt	Description
SSL Hostname: myhost	Specify the SSL Hostname.

9.3.1.2.17 HTTPS Protocol and Use my SSL Certificate

Selecting Use my SSL Certificate requires you to provide an existing SSL certificate and its private key.

Example of Use my SSL Certificate

```
Enter a number to select the certificate type
   [1] Use self-signed certificate (generates automatically)
   [2] Use my SSL certificate (requires SSL certificate and SSL certificate
private key)
Choose [1]: 2

 Enter the path for the SSL Certificate: /path/to/mycertificate.crt
 Enter the path for the SSL Certificates private key: /path/to/mycert.der
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Table 9-5 HTTPS Protocol and Use my SSL Certificate Prompts

Prompt	Description
`Path for the SSL Certificate: /path/to/mycert.crt`	Refer to Entering the Path for the SSL Certificate
`Path for the SSL Certificate private key: /path/to/mycert.der`	Refer to Entering the Path for the SSL Certificate's Private Key

- Entering the Path for the SSL Certificate
- Entering the Path for the SSL Certificate's Private Key

9.3.1.2.17.1 Entering the Path for the SSL Certificate

Specifying the SSL certificate file location and filename.

Example:

```
/path/to/mycert.crt
```

> **See Also**
>
> Converting a Private Key to DER (Linux and Unix)

9.3.1.2.17.2 Entering the Path for the SSL Certificate's Private Key

Specifying the SSL certificate's private key file location and filename.

Example

```
/path/to/mycert.der
```

9.3.1.2.18 Entering the APEX Static Resources Location

This prompt displays only if you are maintaining the APEX static resources.

> **Note**
>
> If you are using the Oracle Content Delivery Network for the APEX static resources, then this prompt is not displayed.

> **See Also**
>
> Oracle APEX Static Resources

Copyright © 2011, 2026, Oracle and/or its affiliates.

9.3.2 Silent Installation of ADB on Command-Line Interface

For silent installation, provide the following command options to setup your Customer Managed ORDS:

- Database Pool: If this option is omitted, then the `default` database pool is used.
- Wallet Path: This is required if this option does not exist in the ORDS configuration database pool.
- Wallet Service Name: The TNS alias name from `tnsnames.ora` file contained in the wallet zip file. If this option is omitted, and the setting `db.wallet.zip.service` does not exist in the ORDS configuration database pool, then the wallet service name defaults to `<DB>_LOW`.
- Administrator username and password (Required)
- Runtime database username and password (Required)
- PL/SQL gateway username and password (Required)
- Additional Database Features

Install ADB Command

```
ords install adb [OPTIONS]
```

- Using Input Redirection

> **See Also**
>
> Customer Managed ORDS Command Options

9.3.2.1 Using Input Redirection

This section describes how to redirect the standard input using the Here document or to a file.

Redirect STDIN to a file

Redirect STDIN to a file that contains the password. In the following example, the file must contain three passwords. Each password must be on a separate line.

Example:

```
ords install adb --admin-user <DATABASE USER> --db-user <DATABASE USER> --
gateway-user <DATABASE
      USER> --wallet <PATH> --wallet-service-name <NET SERVICE NAME> --
feature-sdw <BOOLEAN> --log-folder
      <FOLDER> --password-stdin < filename.txt
```

Where the `filename.txt` contains passwords:

```
<PASSWORD FOR admin-user>
<PASSWORD FOR db-user>
<PASSWORD FOR gateway-user>
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Starting from left to right, the first password belongs to the first user option (`--admin-user`) on the command line. The second password belongs to the second user option on the command line (`--db-user`) and the third password belongs to the third user (`--gateway-user`) option on the command-line.

Redirect Standard Input Using Here Document

Redirect STDIN using the Here document (also known as heredoc) for the password(s). The heredoc consists of the '<<' redirection operator followed by a delimiter token.

Each password must be on a separate line and it is ended by the delimiter token.

Example:

```
ords install adb --admin-user <DATABASE USER> --db-user <DATABASE USER> --
gateway-user <DATABASE USER>
--wallet <PATH> --wallet-service-name <NET SERVICE NAME> --feature-sdw
<BOOLEAN> --log-folder <FOLDER>
--password-stdin << EOF
<PASSWORD FOR admin-user>
<PASSWORD FOR db-user>
<PASSWORD FOR gateway-user>
EOF
```

Starting from left to right, the first password belongs to the first user option (`--admin-user`) on the command line. The second password belongs to the second user option on the command line (`--db-user`) and the third password belongs to the third user (`--gateway-user`) option on the command-line. The Here document is ended by the token EOF.

> **Note**
>
> Once the operation is complete, delete the file or script that contains the passwords.

Copyright © 2011, 2026, Oracle and/or its affiliates.

10

Enabling and Configuring the Oracle Database API for MongoDB

This section describes how to enable and configure the Oracle Database API for MongoDB.

Starting with ORDS release 22.3, Oracle REST Data Services supports the Oracle Database API for MongoDB when running in a standalone mode. This enables the use of MongoDB drivers, frameworks, and tools to develop your JSON document-store applications against the Oracle Database. The Oracle Database API for MongoDB, translates the MongoDB wire protocol into SQL statements that are executed using the ORDS connection pools.

Figure 10-1 Architecture Diagram for Oracle Database API for MongoDB

Following points must be considered:

- The MongoDB concept of a database is mapped to the concept of a schema in Oracle Database. Specifically, an ORDS-enabled schema. For example, when you insert a JSON document into a collection in the database `foo`, the API for MongoDB inserts the document into a collection in the ORDS-enabled schema `foo`.
- Authentication and authorization when using the API for MongoDB depends on the Oracle Database users and access controls and not the MongoDB users. When you are connecting a MongoDB client, you must specify the Oracle Database credentials using the LDAP authentication mechanism of MongoDB using the connection options `authMechanism=PLAIN` and `authSource=$external`. The protocol passes the username and password as a plain text and so, TLS/SSL must be enabled using the connection option `tls=true`.
- The Oracle API for MongoDB maps the MongoDB commands to the corresponding SQL statements on the table backing the collection.
  For example, a MongoDB command such as `emp.find({"name":"John"})` is executed against the database using a SQL select statement similar to `select data from emp e where e.data."name" = 'John'`.
- When required, SQL can be executed directly over JSON collections. With this model, you get the speed, flexibility, and ease-of-use of a NoSQL document store while still having the ability to use SQL for analytics and reporting directly over your natively stored JSON data.

- Getting Started
- Requirements
  This section lists the client and database requirements.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Configurable Settings for MongoDB
  This section lists the editable configuration settings to support the MongoDB API stored in the global configuration, located at `global/settings.xml`.
- Examples
  This section lists some examples that use the `ords config set` command to store the MongoDB settings in the current working directory (CWD) `global/settings.xml` and also provides an example for MongoDB listener settings in `global/settings.xml` file.
- Accessing the Connection Pools
- Logging MongoDB API Access
  This section describes how to enable logging of requests to the MongoDB API.

See Also

- Oracle Database API for MongoDB
- Interactive Command-Line Interface Installation
- Serve Commands for Running in Standalone Mode

10.1 Getting Started

Perform the following steps to get started to use the Oracle Database API for MongoDB:

1. **Install and configure ORDS**:
   From a command prompt, install and configure ORDS using the following commands:

   Note

   - If you are installing ORDS against the Autonomous Database, then use the command `ords install adb`.
   - Oracle API for MongoDB is only supported for installations in a non-CDB database or PDB.

   See Also

   - Installing Oracle REST Data Services
   - Installing and Configuring Customer Managed ORDS on Autonomous Database

   a. Install ORDS in a standalone mode in a non-CDB Database or in a PDB.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> Oracle API for MongoDB is only supported for installations in a non-CDB database or PDB.

```
ords install
```

b. By default, the Oracle API for MongoDB is not enabled. To enable the API, in your configuration, include the following:

```
ords config set mongo.enabled true
```

c. Start ORDS

```
ords serve
```

When the MongoDB API is enabled, you will get a notification and you get the connect string when ORDS is started.

Log showing that the Oracle API for MongoDB is enabled:
After starting ORDS using the `ords serve` command, the log shows a message similar to the following to verify if the Oracle API for MongoDB is enabled:

```
Disabling document root because the specified folder does not exist: ./
config/global/doc_root
2022-08-17T15:23:04.043Z INFO        Oracle API for MongoDB listening on
port: 27017
2022-08-17T15:23:04.050Z INFO        The Oracle API for MongoDB connection
string is:
mongodb://[{user}:{password}@]localhost:27017/{user}?
authMechanism=PLAIN&authSource=$external&ssl=true&retryWrites=false&loadBal
anced=true
```

2. **Create an ORDS-enabled user**:
Leave the server running and from SQLcl, create an ORDS-enabled user.

> **See Also**
>
> Creation of Users, Authentication, and Authorization

```
create user foo identified by "<Password>";
grant soda_app, create session, create table, create view, create
sequence, create procedure, create job, unlimited tablespace to foo;
connect foo/<Password>
exec ords.enable_schema;
```

3. **Connect to ORDS using the MongoDB shell**:
By default, ORDS uses a self-signed certificate for secure communication. The Mongo API does not allow self-signed certificates, so you cannot directly connect with any Mongo tool or driver to the Mongo API.

The alternatives are as follows:

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Disabling TLS option is specifically for local environments that are not dealing with sensitive data. To disable TLS, you must invoke the following command before starting ORDS:

```
ords config set mongo.tls false
ords serve
```

> **Note**
>
> Mongo API connect string changes to reflect that `mongo.tls` is not enabled. For example: `...ssl=false...`.

- Use the mongo tools and drivers to explicitly disable the requirement for TLS to require a valid signed certificate.

  The settings change depending on the tools and the APIs. For example, the `mongosh` has a ar`--tlsAllowInvalidCertificates` command line argument, whereas `mongoimport` has `--tlsInsecure` command line argument.

> **See Also**
>
> Support for MongoDB APIs, Operations, and Data Types — Reference

- Configure the properties `standalone.https.cert` and `standalone.https.cert.key` to aquire a valid signed cerfificate.

Example using `mongosh` without a valid signed certificate:

```
 mongosh  --tlsAllowInvalidCertificates 'mongodb://
foo:<Password>@localhost:27017/foo?
authMechanism=PLAIN&authSource=$external&tls=
true&retryWrites=false&loadBalanced=true'foo> db.createCollection('emp');
    { ok: 1 }
    foo> db.emp.insertOne({"name":"Blake","job": "Intern","salary":30000});
    ...
    foo> db.emp.insertOne({"name":"Smith","job": "Programmer","salary":
60000,"email" : "smith@oracle.com"});
    ...
    foo> db.emp.insertOne({"name":"Miller","job": "Programmer","salary":
70000});
    ...
    foo> db.emp.find({"name":"Miller"});
    [
      {
        _id: ObjectId("6320bfc40dd73b60ef5641b9"),
        name: 'Miller',
        job: 'Programmer',
        salary: 70000
      }
    ]


    foo> db.emp.updateOne({"name":"Miller"}, {$set:
{"email":"miller@oracle.com"}})
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
```

4. The data inserted from the MongoDB client can be accessed from SQL:

```
SQL> select json_serialize(e.data)
     from emp e;

JSON_SERIALIZE(E.DATA)
--------------------------------------------------------------------------------
-----
{"_id":"6320bfa30dd73b60ef5641b7","name":"Blake","job":"Intern","salary":30
000}
{"_id":"6320bfb30dd73b60ef5641b8","name":"Smith","job":"Programmer","salary
":60000,"email":"smith@oracle.com"}
{"_id":"6320bfc40dd73b60ef5641b9","name":"Miller","job":"Programmer","salar
y":70000,"email":"miller@oracle.com"}

SQL> select e.data."name".string() n,
           e.data."job".string() j
     from emp e
     where e.data."email".string() = 'miller@oracle.com';

N                    J
-------------------- ----------------------
Miller               Programmer
```

10.2 Requirements

This section lists the client and database requirements.

The MongoDB API supports Oracle Database version 21c or later and the Autonomous Oracle Database 19c or later (serverless, dedicated, and cloud@customer). In general, Oracle API for MongoDB supports MongoDB tools and drivers that support the loadBalanced connection option.

> **See Also**
>
> Tools and Drivers for Oracle Database API for MongoDB

10.3 Configurable Settings for MongoDB

This section lists the editable configuration settings to support the MongoDB API stored in the global configuration, located at `global/settings.xml`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> Oracle recommends users to use the Oracle REST Data Services command-line interface to edit the configuration files.

Table 10-1 Configuration Settings to Support the Mongo API

Key	Type	Description
`mongo.enabled` (Mandatory property)	boolean	Specifies to enable the API for MongoDB. Default value is `false`. Specifies to enable the API for MongoDB, set the value to `true`.
`mongo.tls`	boolean	Enables the use of the MongoDB API with non-TLS connections (Oracle Database 19c and later).
`mongo.access.log`	path	Specifies the path to the folder where you want to store the API for MongoDB access logs. If the path is not specified, then no access is generated.
`mongo.host`	string	Specifies a comma separated list of host names or IP addresses to identify a specific network interface on which to listen. Default value is `0.0.0.0`.
`mongo.port`	integer	Specifies the API for MongoDB listen port. Default value is `27017`.
`mongo.idle.timeout`	duration	Specifies the maximum idle time for a connection in milliseconds. Default value is `30m`
`mongo.op.timeout`	duration	Specifies the maximum time for a database operation in milliseconds. Default value is `10m`.

> **See Also**
>
> - About the Oracle REST Data Services Configuration Files

10.4 Examples

This section lists some examples that use the `ords config set` command to store the MongoDB settings in the current working directory (CWD) `global/settings.xml` and also provides an example for MongoDB listener settings in `global/settings.xml` file.

Examples of using the ords config set command

- `ords config set mongo.tls false`
- `ords config set mongo.enabled true`
- `ords config set mongo.host example.com`
- `ords config set mongo.port 27017`
- `ords config set mongo.idle.timeout 40m`
- `ords config set mongo.op.timeout 15m`

Copyright © 2011, 2026, Oracle and/or its affiliates.

Example of Mongo Listener settings in global/settings.xml

```
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>Saved on Wed Jun 15 01:33:58 UTC 2022</comment>
<entry key="debug.printDebugToScreen">true</entry>
. . .
<entry key="mongo.tls">false</entry>
<entry key="mongo.enabled">true</entry>
<entry key="mongo.host">example.com</entry>
<entry key="mongo.port">27017</entry>
<entry key="mongo.idle.timeout">40m</entry>
<entry key="mongo.op.timeout">15m</entry>
. . .
</properties>
```

Examples of using the ords config get command

- `ords config get mongo.enabled`
  `true`
- `ords config get mongo.port`
  `27016`

Using the ords serve Command

Use the ords `serve` command to run in standalone mode.

> **Note**
>
> To enable the Oracle API for MongoDB, prior to running the `serve` command, you must set `mongo.enabled` property to `true`

> **See Also**
>
> - Setting Up the Configuration Folder Structure
> - Serve Commands for Running in Standalone Mode

10.5 Accessing the Connection Pools

Oracle REST Data Services supports the ability to connect to more than one database. Multiple named pools can be defined using the install command. Adding a pool creates a corresponding directory under `./databases` within the ORDS configuration directory. An initial install of Oracle REST Data Services typically adds a default database connection pool named `default`.

By default, when you are connecting to the MongoDB API, all database requests are directed to the default connection pool. Optionally, you can route MongoDB API requests to other connection pools using the hostnames file. For example, assume that the database pool named `mydb1` has a configuration directory at `ords_config/databases/mydb1`.

Copyright © 2011, 2026, Oracle and/or its affiliates.

> **Note**
>
> You cannot use multiple pools when `mongo.tls` key is disabled, only default pool is supported.

If you create the hostname file `ords_config/databases/mydb1/hostnames` containing two hostnames as follows:

```
www.example.com
example.com
```

Then, the following connection strings routes to `mydb1` conection pool instead of the `default` connection pool:

```
"mongodb://www.example.com:27017/scott?
authMechanism=PLAIN&authSource=$external&ssl=true&retryWrites=false&loadBalanc
ed=true"

"mongodb://example.com:27017/scott?
authMechanism=PLAIN&authSource=$external&ssl=true&retryWrites=false&loadBalanc
ed=true"
```

> **See Also**
>
> - Routing Based on the Request Hostname

10.6 Logging MongoDB API Access

This section describes how to enable logging of requests to the MongoDB API.

By default, requests to the MongoDB API are not logged. To enable logging access to MongoDB API, you must set the configuration property `mongo.access.log` to a directory path. If the directory path is not absolute, it is resolved relative to the the ORDS configuration directory (`<ords config>`). If the directory does not exist, then ORDS creates the directory on startup. ORDS then adds an access log file entry within this directory each time the MongoDB API is accessed.

Example:

```
ords config set mongo.access.log mongologs
```

This command writes access log files under `<ords config>/mongologs/` folder.

Copyright © 2011, 2026, Oracle and/or its affiliates.

11

Migration of Legacy Configuration

This section describes how to migrate the legacy configuration and the URL mapping.

Legacy configuration is any configuration of ORDS for versions starting with 2.x to version 21.4.x. These configurations need to be migrated to the new directory structure for 22.1 and higher versions of ORDS.

- Migrating From Legacy Configurations Using the CLI Installation
  To migrate your applications from legacy configurations, you can use the `install` command or the `migrate config` command.
- Migrating URL Mapping

11.1 Migrating From Legacy Configurations Using the CLI Installation

To migrate your applications from legacy configurations, you can use the `install` command or the `migrate config` command.

You can migrate your applications from legacy configurations in the following two ways:

Migrating the Configuration and Upgrading ORDS

You use this option when you migrate your legacy configurations (ORDS 21.4.x or earlier versions) to a new configuration structure, and also upgrade ORDS in the database. You can perform this using one of the following two options:

Using the Interactive Install Command

If you use the interactive `install` command, then it migrates the legacy configurations to the new configuration structure. Then, depending on the options that you selected in the earlier steps of installation, it prompts you to either upgrade ORDS in the database, or create or update additional database pools.

```
ords --config <configuration_folder> install --interactive
--legacy-config <legacy_config_folder> --log-folder <log_folder>
```

Using the Non-Interactive Install Command

If you use the non-interactive `install` command, then you must specify the required information to migrate the configuration in the command line.

```
ords --config <configuration_folder> install [--db-pool <pool_name>] --admin-
user <user>
 --legacy-config <folder> --log-folder <log_folder> --password-stdin <  <text-
file>
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Migrating the Configuration Only

You use this option when you only migrate your legacy configurations (ORDS 21.4.x or earlier version) to a new configuration structure, but do not upgrade ORDS to a newer version.

```
ords --config <configuration_folder> migrate config <legacy_config_folder>
```

When migrating the configuration completes, the following message is displayed:

```
Legacy configuration files located at <legacy_config> are no longer being
used to configure ORDS
 or its connection pools. Your migrated configuration files are now located
at <configuration_folder>.
```

11.2 Migrating URL Mapping

The legacy URL mapping file is `/path/to/conf/url-mappings.xml` that contains different types of mappings.

The installer migrates the url-mappings. For each url-mapping that contains the following type, it performs the following:

- `base-path type`: creates a path file for that database-pool containing the base path (single or multiple path segments)
- `base-url type`: creates a hostname file for that database-pool containing the hostname.

The base-path mapping pattern may contain a path that consists of a single segment or multiple segments.

Example:

- `/sales`
- `/pls/hr`

The base-url mapping pattern may contain only the hostname, or it can include the protocol, hostname, and path.

Example:

https://example.com/ords/support

www.sales.com

The ORDS install CLI migrates the `url-mapping.xml` as follows:

For each base-path type mapping (for example, `/sales`), the path is added to the paths file of the pool.

> **Note**
>
> If the paths file does not exist, it is created.

For each base-url type mapping (for example https://example.com/ords/support), the hostname (for example example.com) is added to the hostnames file of the pool.

Copyright © 2011, 2026, Oracle and/or its affiliates.

If the base-url mapping contains a path after the context (for example, `/ords`), then the path (for example, `/support`) is added to the paths file.

> **Note**
>
> If the hostnames or paths files do not exist, then it is created.

> **Note**
>
> A SEVERE warning message is displayed if:
>
> - Duplicate paths or hostnames across different database pools exist
> - The database pool that the mapping points to does not exist
> - The hostname or path is invalid

> **Note**
>
> The name of the directory that contains the `pool.xml` configuration file is used in the request URL path mapping. When you are migrating a `url-mapping.xml` file, the pool name from that file is used as the directory name. There is a possibility that a pool name overlaps with an entry in the paths file from another pool definition. The pool name takes precedence in the request URL mapping.

The following is an example `url-mapping.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<pool-config xmlns="http://xmlns.oracle.com/apex/pool-config">
 <pool name="db1" base-path="/sales" updated="2021-09-23T19:22:19.944Z"/>
 <pool name="db1" base-path="finance/" updated="2021-09-23T19:22:50.189Z"/>
 <pool name="db2" base-url="https://example.com/ords/support"
updated="2021-09-23T19:23:04.751Z"/>
 <pool name="db3" base-path="/pls/hr" updated="2021-09-23T19:23:14.832Z"/>
 <pool name="db4" base-path="employee" updated="2021-09-23T19:23:24.252Z"/>
 <pool name="db5" base-url="www.sales.com"
updated="2021-09-23T19:23:35.322Z"/>
</pool-config>
```

Example after migrating the url-mappings to the database pools folder:

```
+- databases/
  +- db1/
    +- pool.xml
    +- paths     // contains: [sales, finance]
  +- db2/
    +- pool.xml
    +- hostnames // contains: [example.com]
    +- paths     // contains: [support]
  +- db3/
    +- pool.xml
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
  +- paths     // contains: [/pls/hr]
+- db4/
  +- pool.xml
  +- paths     // contains: [employee]
+- db5/
  +- pool.xml
  +- hostnames // contains: [www.sales.com]
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

A

Setting-up a PL/SQL Gateway User

This section describes how to configure:

- PL/SQL gateway user for APEX
- PL/SQL gateway user to serve PL/SQL gateway requests directly

- Configuring a PL/SQL Gateway User for APEX
  This section explains how to configure the PL/SQL gateway user for APEX.
- Configuring a PL/SQL Gateway User to Serve PL/SQL Gateway Requests Directly

A.1 Configuring a PL/SQL Gateway User for APEX

This section explains how to configure the PL/SQL gateway user for APEX.

If you have installed ORDS and APEX, and the APEX PL/SQL gateway user needs to be configured, then perform the following:

1. Repair ORDS.

   > **See Also**
   >
   > Repairing the Oracle REST Data Services Installation

2. Verify if the configuration setting in the database pool is set correctly.

   ```
   ords --config <config path> config --db-pool <pool name> list
   ```

   > **Note**
   >
   > If you have only a single database pool (For example: Pool named `default`), then omit the `--db-pool` option.

   If the setting `plsql.gateway.mode` does not exist or is not set to `proxied`, then run the following command:

   ```
   ords --config <config path> config --db-pool <pool name> set
   plsql.gateway.mode
       proxied
   ```


Copyright © 2011, 2026, Oracle and/or its affiliates.

A.2 Configuring a PL/SQL Gateway User to Serve PL/SQL Gateway Requests Directly

If you want the database pool to serve the PL/SQL gateway requests directly, then specify the options `--gateway-user <database user>` and `--gateway-mode direct` to configure the PL/SQL gateway user.

- Installing/Upgrading the PL/SQL Gateway User
  This section explains how to configure the PL/SQL gateway user while installing or upgrading ORDS.
- Configuring the PL/SQL Gateway User

A.2.1 Installing/Upgrading the PL/SQL Gateway User

This section explains how to configure the PL/SQL gateway user while installing or upgrading ORDS.

To configure the PL/SQL gateway to install or upgrade ORDS, you can use one of the following ways:

- Install/upgrade ORDS and configure PL/SQL gateway in interactive mode
- Install/upgrade ORDS and configure PL/SQL gateway in non-interactive (silent) mode

Install/upgrade ORDS and configure PL/SQL gateway in interactive mode

To install using the interactive mode, use the interactive `ords install -i` command and specify the `--gateway-user` and `--gateway-mode` options.

```
ords --config <config path> install -i --gateway-user <database user> --
gateway-mode direct

ords --config /path/to/config install -i --gateway-user EXAMPLE_USER1 --
gateway-mode direct
```

This command prompts for the connection information and the password of the gateway user. The password of the gateway user must match with the existing password of the database user. The following options are displayed and enables you to make changes:

```
...

Oracle REST Data Services - Interactive Install

  Enter a number to select the database connection type to use
    [1] Basic (host name, port, service name)
    [2] TNS (TNS alias, TNS directory)
    [3] Custom database URL
  Choose [1]:
  Enter the database host name [localhost]:
  Enter the database listen port [1521]:
  Enter the database service name [orcl]: pdb1
  Enter the PL/SQL Gateway database password for EXAMPLE_USER1:
Confirm password:
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
Connecting to database user: EXAMPLE_USER1 url: jdbc:oracle:thin:@//
localhost:1521/pdb1
  Provide database user name with administrator privileges.
    Enter the administrator username: sys
  Enter the database password for SYS AS SYSDBA:

Retrieving information.
ORDS is not installed in the database. ORDS installation is required.

  Enter a number to update the value or select option A to Accept and Continue
    [1] Connection Type: Basic
    [2] Basic Connection: HOST=localhost PORT=1521 SERVICE_NAME=pdb1
           Administrator User: SYS AS SYSDBA
    [3] Database password for ORDS runtime user (ORDS_PUBLIC_USER): <generate>
    [4] Database password for ORDS gateway user (EXAMPLE_USER1): *****
    [5] ORDS runtime user and schema tablespaces:  Default: SYSAUX Temporary
TEMP
    [6] Configure and start ORDS in Standalone Mode: Yes
    [7]    Protocol: HTTP
    [8]       HTTP Port: 8080
    [9]   APEX static resources location:
    [A] Accept and Continue - Create configuration and Install ORDS in the
database
    [Q] Quit - Do not proceed. No changes
  Choose [A]:
```

See Also

- Scenario 3: Multiple Database Pools
- Multiple Options with Default Values

Install/upgrade ORDS and configure PL/SQL gateway in non-interactive (silent) mode

To install using the non-interactive mode, use `ords install` command and specify `--gateway-user` and `--gateway-mode` options to configure the PL/SQL gateway user. Additional connection information options and the administrator user is required for the installation and configuration.

```
ords --config <path> install --db-pool <pool name> --admin-user
<administrator user>
--db-hostname <hostname> --db-port <port> --db-servicename <database
servicename>
--gateway-user <username> --gateway-mode direct --password-stdin < secret.txt

ords --config /path/to/config install --db-pool db1 --admin-user SYS
--db-hostname localhost --db-port 1521 --db-servicename pdb1 --gateway-user
EXAMPLE_USER1
--gateway-mode direct --password-stdin < secret.txt
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

> **See Also**
>
> Non-Interactive Command-Line Interface Installation (Silent)

A.2.2 Configuring the PL/SQL Gateway User

You can configure the PL/SQL gateway using either of the following modes :

- Running in interactive mode
- Running in non-interactive (silent) mode

Running in Interactive Mode

To configure the PL/SQL gateway user, use the `ords install -i --config-only` interactive command, specifying `--gateway-user` and `--gateway-mode` options.

```
ords install -i --config-only --gateway-user <username> --gateway-mode direct
```

Running in Non-Interactive (Silent) Mode

To configure the PL/SQL gateway user, use the `ords install --config-only` silent command, specifying `--gateway-user` and `--gateway-mode` options.

```
ords --config <path> install --config-only --db-pool <pool name> --admin-user
<administrator user>
--db-hostname <hostname> --db-port <port> --db-servicename <database
servicename>
--gateway-user <username> --gateway-mode direct --password-stdin < secret.txt
```

For example:

```
ords --config /path/to/config install --config-only --db-pool db1 --admin-
user SYS  --db-hostname localhost
--db-port 1521 --db-servicename pdb1 --gateway-user EXAMPLE_USER1 --gateway-
mode direct --password-stdin < secret.txt
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

B

Non-Interactive ORDS CLI Commands Examples

This section provides some examples for non-interactive install/upgrade, and uninstall.

The following are examples for non-interactive install/upgrade, and uninstall.

The password file must contain the password(s) in the correct order to match the order of the user(s), option(s) that are specified on the command-line.

The following examples can be used for a PDB, CDB. Ensure that the `--db-servicename` option contains the correct value for the servicename for the CDB, PDB.

If the `--log-folder <FOLDER>` option is specified and the log folder does not exist, then the folder is created.

- Non-Interactive Install/Upgrade
  This section explains how to create a proxied APEX_PUBLIC_USER.
- Non-Interactive Install/Upgrade (create a proxied PLSQL Gateway User)
  This section explains how to create a proxied PLSQL Gateway User.
- Non-Interactive Install/Upgrade (specify PLSQL Gateway User and not using RESTful Services)
  This section explains how to specify PLSQL Gateway User not using RESTful Services.
- Non-Interactive Install/Upgrade (disable a configured PLSQL Gateway)
  This section explains how to disable a configured PLSQL Gateway.
- Non-Interactive Uninstall ORDS
  This section explains how to uninstall ORDS in the database.
- config CLI Examples
  This section provides config CLI examples.
- Resolving 570 Server Error Response Code
  This section provides the steps to resolve a 570 Server Error Response code.

B.1 Non-Interactive Install/Upgrade

This section explains how to create a proxied APEX_PUBLIC_USER.

If APEX is installed and the `APEX_PUBLIC_USER` exists in the database, then the APEX_PUBLIC_USER is automatically configured to be a proxied user.

```
cat password.txt

<password for admin-user>

<password for proxy-user>

ords --config /path/to/test/config install --db-pool pdb1 --admin-user SYS --
proxy-user --db-hostname localhost --db-port 1521 --db-servicename orcl --
feature-sdw true --log-folder /path/to/logs --password-stdin < password.txt
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

B.2 Non-Interactive Install/Upgrade (create a proxied PLSQL Gateway User)

This section explains how to create a proxied PLSQL Gateway User.

Specify a PL/SQL gateway user and `--gateway-mode` proxied.

```
ords --config /path/to/test/config install --db-pool pdb2 --admin-user SYS --
proxy-user --db-hostname localhost --db-port 1521 --db-servicename pdb2 --
feature-sdw true --gateway-user EXAMPLE_USER1 --gateway-mode proxied --log-
folder /path/to/logs --password-stdin < password.txt
```

B.3 Non-Interactive Install/Upgrade (specify PLSQL Gateway User and not using RESTful Services)

This section explains how to specify PLSQL Gateway User not using RESTful Services.

Specify a PL/SQL gateway user and --gateway-mode direct. The direct mode indicates that you are only using the PL/SQL GATEWAY and not RESTful Services. The gateway user password is required for direct mode.

```
cat gatewaypwd.txt

<password for admin-user>

<password for proxy-user>

<password for gateway-user>

ords --config /path/to/test/config install --db-pool pdb3 --admin-user SYS --
proxy-user --db-hostname localhost --db-port 1521 --db-servicename pdb3 --
gateway-user EXAMPLE_USER2 --gateway-mode direct --log-folder /path/to/logs --
password-stdin < gatewaypwd.txt
```

B.4 Non-Interactive Install/Upgrade (disable a configured PLSQL Gateway)

This section explains how to disable a configured PLSQL Gateway.

Disables the proxied PL/SQL gateway user.

```
ords --config /path/to/test/config install --db-pool pdb2 --admin-user SYS --
proxy-user --db-hostname localhost --db-port 1521 --db-servicename pdb2 --
feature-sdw true --gateway-mode disabled --log-folder /path/to/logs --password-
stdin < password.txt
```

B.5 Non-Interactive Uninstall ORDS

This section explains how to uninstall ORDS in the database.

Copyright © 2011, 2026, Oracle and/or its affiliates.

To uninstall ORDS, execute the following commands:

```
cat adminpwd.txt
<password for admin-user>

ords --config /path/to/test/config uninstall --admin-user SYS --db-hostname
localhost --db-port 1521
--db-servicename orcl --log-folder /path/to/logs < adminpwd.txt
```

B.6 config CLI Examples

This section provides config CLI examples.

Non-Interactive Update Wallet

If you want to change the password in the wallet, then specify the following:

```
ords --config /path/to/test/config config secret --password-stdin db.password <
userpwd.txt
```

If you want to view the password in the wallet, then specify the following:

```
ords --config /path/to/test/config config get --secret db.password
```

Where:

`userpwd.txt` contains the password.

> **Note**
>
> If you are changing the wallet for a specific pool, then you must provide the `--db-pool` option. The preceding example updates the default pool.

B.7 Resolving 570 Server Error Response Code

This section provides the steps to resolve a 570 Server Error Response code.

This issue is commonly encountered during an upgrade or repair when the ORDS_PUBLIC_USER password has been changed. When ORDS starts, it attempts to connect to the target database using the credentials stored in the associated Wallet (.SSO file).

If the wallet password does not match the database password, ORDS attempts to login three times. Repeated failures locks the account. Perform the following steps to resolve this issue:

Perform the following steps to resolve this issue:

1. Stop the affected ORDS instance/process.
2. Unlock the `ORDS_PUBLIC_USER` account using the following command:

   ```
   alter user ORDS_PUBLIC_USER unlock;
   ```


Copyright © 2011, 2026, Oracle and/or its affiliates.

a. Verify the account status using the following command:

```
select username, account_status from dba_users where
username ='ORDS_PUBLIC_USER';
```

3. You can either change or update the `ORDS_PUBLIC_USER` password (to match the current database password) or verify the password that is stored in the ORDS Wallet (.SSO file).
   - To change the `ORDS_PUBLIC_USER` password, use the following command:

     ```
     ords --config <PATH TO ORDS CONFIG> config secret db.password
     ```

   - To verify the password, use the following command:

     ```
     ords --config <PATH TO ORDS CONFIG> config get --secret db.password
     ```

4. If you change the password in the `.SSO` wallet, restart your shell or terminal session and then retry the upgrade or repair:

   ```
   ords --config <Path to your ORDS configuration folder> install
   ```

   When prompted, choose the database pool associated with the mismatched wallet password.

> **Note**
>
> During installation, the ORDS installer updates the database password if it detects a difference in the password.

Copyright © 2011, 2026, Oracle and/or its affiliates.

C

ORDS Legacy Commands and New ORDS CLI Commands

Table C-1 ORDS Legacy and New CLI Commands

Legacy Command	New CLI Command	Description

`java -jar ords.war configdir`	`ords --config /path/to/config <command>` Example of using config option:
	
	`ords --config /path/to/config install <OPTIONS>`
	`ords --config /path/to/config uninstall <OPTIONS>`
		Specifies the ORDS configuration folder that contains configuration files.
		
		New CLI Command: If the configuration folder is not specified, your current working directory is your configuration folder. The command does not store the configuration location. You must provide the configuration location or use the default configuration location.
		
		Legacy configuration directory:
		
		The configuration location is stored in `ords.war` file. When the user executes the ords command, (example: install, setup) and the configuration location exists in `ords.war` file, then it does not prompt for the configuration location.
		
		Refer to Setting Up the Configuration Folder Structure for more information.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table C-1 (Cont.) ORDS Legacy and New CLI Commands

Legacy Command	New CLI Command	Description

`java -jar ords.war install advanced`	`/ords install`
	`ords --config /path/to/config install`
	`ords --config /path/to/config install --interactive`
	`ords --config /path/to/config install -i`
		Install ORDS in interactive mode.
		Legacy install advanced:
		Prompts for the configuration information only if the ORDS configuration does not exist.
		Prompts if you want to run in a standalone mode.

`java -jar ords.war install --silent`
`java -jar ords.war install simple`
`java -jar ords.war`
	`ords --config /path/to/config install <OPTIONS>`	Install ORDS in a non-interactive mode. No prompting for new install CLI command if the user provides install options and not specifying the `-i` or `--interactive` option.
		Legacy install simple or no command : (`java -jar ords.war`)
		Prompts for the configuration information only if the ORDS configuration does not exist.
		If the configuration file does not exist and if the default parameter file does not contain any configuration values, then it prompts.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table C-1 (Cont.) ORDS Legacy and New CLI Commands

Legacy Command	New CLI Command	Description
`java -jar ords.war schema`	`ords --config /path/to/config install`	Install or upgrade ORDS only. The configuration file is not updated. **New CLI Command**: `Enter a number to select the type of installation [1] Install or upgrade ORDS in the database only [2] Create or update a database pool and install/upgrade ORDS in the database [3] Create or update a database pool only Choose [1]:` Option 1 is equivalent.

`java -jar ords.war uninstall advanced`
`java -jar ords.war uninstall`
	`ords uninstall`
	`ords --config /path/to/config uninstall`
	`ords --config /path/to/config uninstall --interactive`
	`ords --config /path/to/config uninstall -i`
		Uninstalls ORDS in interactive mode.

`java -jar ords.war uninstall --silent` `java -jar ords.war uninstall simple`	`ords --config /path/to/config uninstall <OPTIONS>`	Uninstall ORDS in non-interactive mode. If you provide uninstall options and not specify the `-i` or `--interactive` option, then you are not prompted for new uninstall CLI command.
`java -jar ords.war set-property <KEY> <VALUE>`	`ords --config /path/to/config config set <KEY> <VALUE>`	Updates the setting in the configuration file. See Updating the Configuration Settings for a list of options.
`java -jar ords.war standalone`	`ords --config /path/to/config serve`	Launch ORDS in standalone mode.
`java -jar ords.war user <USERNAME> ROLES`	`ords --config /path/to/config config user add <NAME> <ROLE>`	Create the user and its roles to provide permission to the ORDS REST Services.

`java -jar ords.war validate`	`ords install repair`
	`ords --config /path/to/conf install repair`
	`ords --config /path/to/conf install repair --interactive`
	`ords --config /path/to/conf install repair -i`
		Repair ORDS in interactive mode.

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table C-1 (Cont.) ORDS Legacy and New CLI Commands

Legacy Command	New CLI Command	Description
`java -jar ords.war validate --silent`	`ords --config /path/to/conf install repair <OPTIONS>`	Repair ORDS in non-interactive mode. There is no prompting for install repair CLI command if user provides install options afer omitting the `-i` or `--interactive` option.

Copyright © 2011, 2026, Oracle and/or its affiliates.

D

About the Oracle REST Data Services Configuration Files

The section describes the Oracle REST Data Services configuration files.

- Understanding the Configuration Folder Structure
- Understanding the Configuration File Format
- Understanding the Configurable Settings
  This section lists the configuration settings for the `settings.xml` and `pool.xml` configuration files, and secure settings for the wallet.

D.1 Understanding the Configuration Folder Structure

The configuration folder has the following structure:

```
+- global/
    +- settings.xml
    +- credentials
    +- wallet/
    +- standalone/
+- databases/
    +- default/
        +- pool.xml
        +- wallet/
    +- myapp/
         +- hostnames
         +- pool.xml
         +- wallet/
    +- myapp2/
         +- paths
         +- pool.xml
         +- wallet/
```

The global/ folder contains settings that apply across the entire ORDS instance:

- `settings.xml`: Contains settings that are configured across the entire ORDS instance. For example: `debug.printToScreen=true`
- `credentials`: The ORDS user password file
- `wallet/`: Contains an Oracle auto login wallet that contains the instance wide encryption and mac keys previously stored in `security.crypto.enc.password` and `security.crypto.mac.password` configuration settings in `defaults.xml`.
- `standalone/`: Contains standalone mode related resources such as the HTTPS certificate and key.

The `databases/` folder contains database pool configurations.

Copyright © 2011, 2026, Oracle and/or its affiliates.

- Each pool configuration is located in its own folder. The base path url mapping for a pool is inferred from the folder name. If the folder is named `foo/`, then requests can be mapped to the pool by accessing https://server/ords/foo/...
- The database pool folder name must only contain lowercase alphabet a-z, digits 0-9, '-', '.' or '_' character.
- The folder named `default/` is reserved and is used to map requests that are not mapped to any other pool. It is equivalent to the `apex.xml` pool in the old structure.
- The folder named `databases/<pool-name>/wallet/` contains an Oracle auto login wallet that contains the credentials for the database pool. The database username and password must be stored in the wallet. The `db.password` settings must not be used. The wallet must conform to the requirements for ORDS wallets.
- Alternatively, the folder may contain a file named hostname or paths, but not both.

D.2 Understanding the Configuration File Format

Configuration files use the standard Java XML properties file format, where each configuration setting contains a key and a corresponding value. The following is an example of `settings.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<entry key="db.connectionType">basic</entry>
<entry key="db.hostname">localhost</entry>
<entry key="db.port">1521</entry>
<entry key="db.servicename">orcl</entry>
<entry key="jdbc.DriverType">thin</entry>
<entry key="jdbc.InitialLimit">3</entry>
<entry key="jdbc.MinLimit">1</entry>
<entry key="jdbc.MaxLimit">10</entry>
<entry key="jdbc.MaxStatementsLimit">10</entry>
<entry key="jdbc.InactivityTimeout">1800</entry>
<entry key="jdbc.statementTimeout">900</entry>
<entry key="jdbc.MaxConnectionReuseCount">1000</entry></properties>
```

D.3 Understanding the Configurable Settings

This section lists the configuration settings for the `settings.xml` and `pool.xml` configuration files, and secure settings for the wallet.

> **Note**
>
> - Oracle recommends users to use the Oracle REST Data Services command-line interface to edit the configuration files.
> - To add or change a *Secure setting*, you must use the ords config secret command.
> - To add, change, or delete a setting refer to Updating the Configuration Settings section.
>   Example: `ords config secret db.password`

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`apex.security.administrator.roles`	string	Specifies the comma delimited list of additional roles to assign authenticated APEX administrator type users.	Not applicable	Pool specific	APEX Security Administrator Roles
`apex.security.user.roles`	string	Specifies the comma delimited list of additional roles to assign authenticated regular APEX users.	Not applicable	Pool specific	APEX Security User Roles
`autoupgrade.api.aulocation`	string	specifies a configuration setting for `AutoUpgrade.jar` location.	Not applicable	Pool specific	Oracle Database Auto Upgrade API Location
`autoupgrade.api.enabled`	boolean	Specifies a configuration setting to enable AutoUpgrade REST API features.	Not applicable	Pool specific	Oracle Database Auto Upgrade API Enabled
`autoupgrade.api.jvmlocation`	string	Specifies a configuration setting for AutoUpgrade REST API JVM location.	Not applicable	Pool specific	Oracle Database Auto Upgrade API JVM Location
`autoupgrade.api.loglocation`	string	Specifies a configuration setting for AutoUpgrade REST API log location.	Not applicable	Pool specific	Oracle Database Auto Upgrade API Log Location
`database.api.management.services.disabled`	boolean	Specifies to disable the Database API administration related services. Only applicable when Database API is enabled.	Not applicable	Global	Database API Management Services Disabled
`db.adminUser`	string	Specifies the username for the database account that ORDS uses for administration operations in the database.	Not applicable	Pool specific	Not applicable
`db.cdb.adminUser`	string	Specifies the username for the database account that ORDS uses for the Pluggable Database Lifecycle Management.	Not applicable	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`db.credentialsSource`	string	Specifies the source for database credentials when creating a direct connection for running SQL statements. Value can be one of pool or request. If the value is `pool`, then the credentials defined in this pool is used to create a JDBC connection. If the value `request` is used, then the credentials in the request is used to create a JDBC connection and if successful, grants the requestor SQL Developer role. Default value: `pool`	Not applicable	Pool specific	Database Credentials Source
`cache.metadata.graphql.expireAfterAccess`	duration	Specifies the duration after a GraphQL schema is not accessed from the cache that it expires. Default value: 1 minute	`2m`	Global	GraphQL Schema Expiration After Access
`cache.metadata.graphql.expireAfterWrite`	duration	Specifies the duration after a GraphQL schema is cached that it expires and has to be loaded again. Default value: 2 minutes	`5m`	Not applicable	GraphQL Schema Expiration After Write
`feature.graphql`	string	Specifies which GraphQL features are enabled. Value can be one of: `enabled`, `disable_graphiql` or `disable_all`. • If the value is set to `enabled`, then all the GraphQL features are available. • If the value is set to `disable_graphiql`, then the GraphQL feature is disabled. • If the value is set to `disable_all`, then all the GraphQL features are unavailable. The default value is `enabled`.	`enabled`	Pool specific	GraphQL Enabled

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`cache.metadata.jwks.enabled`	boolean	Specifies the setting to enable or disable JWKS caching. Supported values: • `true` • `false` Default value: `true`	`true`	Global	JWKS Metadata Caching Enabled
`cache.metadata.jwks.initialCapacity`	numeric	Specifies the initial capacity of the JWKS cache.	`10`	Global	JWKS Cache Initial Capacity
`cache.metadata.jwks.maximumSize`	numeric	Specifies the maximum capacity of the JWKS cache.	`10000`	Global	JWKS Cache Maximum Size
`cache.metadata.jwks.expireAfterAccess`	duration	Specifies the duration after a JWK is not accessed from the cache that it expires. By default this is disabled.	`2m`	Global	JWKS Expiration After Access
`cache.metadata.jwks.expireAfterWrite`	duration	Specifies the duration after a JWK is cached, that is, it expires and has to be loaded again. Default value: 5 minutes	`5m`	Global	JWKS Expiration After Write
`db.invalidPoolTimeout`	duration	Specifies how long to wait before retrying an invalid pool. Default value: `15m`	Not applicable	Global	Invalid Pool Timeout
`db.poolDestroyTimeout`	duration	Indicates how long to wait to gracefully destroy a pool before moving to forcefully destroy all connections including borrowed ones. Default value: `5m`	Not applicable	Pool specific	Pool Destroy Timeout
`db.wallet.zip`	string	Specifies the wallet archive (provided in BASE64 encoding) containing connection details for the pool.	Not applicable	Pool specific	Not applicable
`db.wallet.zip.path`	string	Specifies the path to a wallet archive containing connection details for the pool.	Not applicable	Pool specific	Not applicable
`db.wallet.zip.service`	string	Specifies the service name in the wallet archive for the pool.	Not applicable	Pool specific	Not applicable
`debug.trackResources`	boolean	Specifies to enable tracking of JDBC resources. If not released causes in resource leaks or exhaustion in the database. Tracking imposes a performance overhead.	Not applicable	Pool specific	Track Resources
`feature.graphql.max.nesting.depth`	numeric	Specifies the maximum join nesting depth limit for GraphQL queries. Defaults to 5.	`10`	Global	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`feature.openservicebroker.exclude`	boolean	Specifies to disable the Open Service Broker services available for the pool.	Not applicable	Pool specific	Exclude Open Service Broker
`feature.sdw`	boolean	Specifies to enable the Database Actions feature.	Not applicable	Pool specific	Not applicable
`http.cookie.filter`	string	Specifies a comma separated list of HTTP Cookies to exclude when initializing an Oracle Web Agent environment.	Not applicable	Pool specific	HTTP Cookie Filter
`instance.api.enabled`	boolean	Specifies the setting to enable or disable the instance API service. The default value is false.	`true`	Global / Pool specific	Instance API Enabled
`jdbc.auth.admin.role`	string	Identifies the database role that indicates that the database user must get the SQL Administrator role.	Not applicable	Pool specific	Auth Admin Role
`jdbc.cleanup.mode`	Not applicable	Specifies how a pooled JDBC connection and corresponding database session, is released when a request has been processed. Default value: `RECYCLE`	Not applicable	Pool specific	Cleanup Mode
`jdbc.sessionlesstxn.timeout`	duration	Specifies the default timeout for sessionless transactions. Default value: 60 seconds	`60s`	Pool specific	Sessionless Transaction Timeout
`owa.trace.sql`	boolean	Specifies a boolean property. If it is true, then it causes a trace of the SQL statements performed by Oracle Web Agent to be echoed to the log.	Not applicable	Pool specific	SQL Tracing
`plsql.gateway.mode`	string	Indicates if the PL/SQL Gateway functionality should be available for a pool or not. Value can be one of `disabled`, `direct`, or `proxied`. • If the value is `direct`, then the pool serves the PL/SQL Gateway requests directly. • If the value is proxied, the `PLSQL_GATEWAY_CONFIG` view decides which user to proxy.	`proxied`	Pool specific	PL/SQL Gateway Mode

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`request.traceHeaderName`	string	Specifies the name of the HTTP request header that uniquely identifies the request end to end as it passes through the various layers of the application stack. In Oracle this header is commonly referred to as the ECID (Entity Context ID).	Not applicable	Global	Request Trace Header Name
`security.jwt.profile.enabled`	boolean	Specifies whether the JWT Profile authentication is available. Supported values: • `true` • `false` Default value: `true`	`true`	Pool specific	Not applicable
`security.jwks.allowed.urls`	string	Schema Level JWT Profile JWK URLs are restricted to public remote URLs by default. To enable local or restricted URLs, provide the exact JWK URLs as a comma-separated list.	`https://localhost/admin/v1/SigningCert/jwk`	Pool specific	Not applicable

`security.oauth.implicitGrantEnabled`	boolean	Enables the OAuth implicit grant flow at runtime. The default value is `false`. When the value is set to `false`, ORDS rejects OAuth implicit grant requests that specify `response_type=token`.
		Authorization code and other supported OAuth flows are not affected. This setting can be enabled at the pool level and overridden for individual ORDS-enabled schemas.
			`false`	Pool specific	`BEGIN`
					
					`ords.set_property(`
					
					`p_key => 'security.oauth.implicitGrantEnabled',`
					`p_value => 'true'`
					`);`
					`COMMIT;`
					`END;`
					`/`

`security.jwks.size`	numeric	Specifies the maximum number of bytes read from the JWK url. Default value: `100000` bytes .	`100000`	Pool specific	JWKS Size
`security.jwks.connection.timeout`	duration	Specifies the maximum amount of time before timing-out when accessing a JWK url. Default value: 5 seconds	`5s`	Pool specific	JWKS Connection Timeout

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`security.jwks.read.timeout`	duration	Specifies the maximum amount of time reading a response from the JWK url before timing-out. Default value: 5 seconds.	`5s`	Pool specific	JWKS Read Timeout
`security.jwks.refresh.interval`	duration	Specifies the minimum interval between refreshing the JWK cached value.	`10s`	Pool specific	JWKS Refresh Interval
`security.jwt.allowed.skew`	duration	Specifies the maximum skew the JWT time claims are accepted. This is useful if the clock on the JWT issuer and ORDS differs by a few seconds. By default, it is disabled.	`10s`	Pool specific	JWT Allowed Skew
`security.jwt.allowed.age`	duration	Specifies the maximum allowed age of a JWT in seconds, regardless of expired claim. The age of the JWT is taken from the JWT issued at claim. By default, it is disabled.	`1h`	Pool specific	JWT Allowed Age
`security.jwt.profile.mode`	Not applicable	Specifies the setting to enable or disable the JWT profile feature. Values can be one of `DISABLED`, `POOL` or `SCHEMA`. • If the value is`SCHEMA`, then JWT profile is defined by OAUTH.CREATE_JWT_PROFILE. • If the value is`POOL`, then JWT profile defined at the pool level is used. • If the value is`DISABLED`, then JWT profile defined at the pool level is used. Default value : `SCHEMA`	`POOL`	Pool specific	Not applicable
`security.jwt.profile.audience`	string	Specifies the expected audience for the JWT token. This value is used to validate the `aud` claim in the JWT token.	`myJWTaudience`	Pool specific	Not applicable
`security.jwt.profile.issuer`	string	Specifies the issuer of the JWT token. This value is used to validate the "`iss`" claim in the JWT token.	`https://identity.oraclecloud.com/`	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`security.jwt.profile.jwk.url`	string	Specifies the URL of the JSON Web Key (JWK) that is used to verify the signature of the JWT token. The JWT URL must use https.	`https://idcs-xxxxxxxxxxxxxxx.identity.oraclecloud.com/admin/v1/SigningCert/jwk`	Pool specific	Not applicable
`security.jwt.profile.role.claim.name`	string	Specifies the JSON pointer to the claim in the JWT token that contains the roles of the users. This is used for Role-Based Access Control (RBAC) with JWT profiles. The role claim name must be a valid JSON pointer and must start with '/ '.	`/roles`	Pool specific	Not applicable
`security.jwt.profile.allowed.roles`	string	Comma-separated list of supported built-in ORDS roles that can be accepted from JWT role claims. By default, built-in ORDS roles are not accepted from JWT role claims, and only an explicitly supported subset can be enabled. In `POOL` mode, this setting is pool-specific. In `SCHEMA` mode, it can be overridden at the schema level.	SQL Developer, SODA Developer	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`security.jwt.profile.allowed.scopes`	string	Comma-separated list of supported built-in ORDS privileges that can be accepted from JWT scope claims. By default, built-in ORDS privileges are not accepted from JWT scope claims, and only an explicitly supported subset can be enabled. In `POOL` mode, this setting is pool-specific. In `SCHEMA` mode, it can be overridden at the schema level.	`oracle.dbtools.auth.privileges.builtin.ResourceModules, oracle.soda.privilege.developer`	Pool specific	Not applicable
`security.credentials.attempts`	numeric	Specifies the maximum number of unsuccessful password attempts allowed. Enabled by setting a positive integer value. Defaults to `-1`.	3	Global	Max Unsuccessful Attempts
`security.credentials.file`	string	Specifies the file where credentials are stored.	Not applicable	Global	Credentials File
`security.credentials.lock.time`	duration	Specifies the period to lock the account that has exceeded maximum attempts. Defaults to `10m` (10 minutes)	`15m`	Global	Account Lock Time
`security.validationFunctionType`	string	Indicates the type of `security.requestValidationFunction: javascript` or plsql. Defaults to `plsql`.	Not applicable	Pool specific	Validation Function Type
`standalone.access.log`	string	Specifies the path to the folder to store HTTP request access logs. If not specified, then no access log is generated.	Not applicable	Global	Access Log Path
`standalone.binds`	string	Specifies the comma separated list of host names or IP addresses to identify a specific network interface on which to listen. Default value: `0.0.0.0`	Not applicable	Global	Bind Addresses
`standalone.context.path`	string	Specifies the context path where ords is located. Defaults to `/ords`	Not applicable	Global	Context Path

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`standalone.doc.root`	string	Points to the location where static resources to be served under the / root server path are located.	Not applicable	Global	Not applicable
`standalone.http.port`	numeric	Specifies the HTTP listen port. Default value: `8080`	`8777`	Global	Not applicable
`standalone.https.cert`	string	Specifies the SSL certificate path. If you are providing the SSL certificate, then you must specify the certificate location.	Not applicable	Global	Not applicable
`standalone.https.cert.key`	string	Specifies the SSL certificate key path. If you are providing the SSL certificate, you must specify the certificate key location.	Not applicable	Global	Not applicable
`standalone.https.host`	string	Specifies the SSL certificate hostname to use when you are generating a Self-Signed Certificate. Once a certificate exists, you cannot change the configuration setting to generate a new certificate.	Not applicable	Global	HTTPS Host
`standalone.https.port`	numeric	Specifies the HTTPS listen port. Default value: `8443`	Not applicable	Global	Not applicable
`standalone.https.san`	boolean	Specifies if the self-signed certificate generated by ORDS in a standalone mode includes a SAN extension. SAN has the same value as defined for the setting `standalone.https.host`.	`false`	Global	HTTPS SAN
`standalone.static.context.path`	string	Specifies the Context path where APEX static resources are located. Default value: `/i`	Not applicable	Global	Not applicable
`standalone.static.path`	string	Specifies the path to the folder containing static resources required by APEX.	Not applicable	Global	Not applicable
`standalone.stop.timeout`	duration	Specifies the period for Standalone Mode to wait until it is gracefully shutdown. Default value: `10s` (10 seconds)	`15s`	Global	Stop Timeout
`apex.docTable`	string	This parameter is deprecated, instead use `owa.docTable` parameter.	`MYDOCTABLE`	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`cache.metadata.timeout`	Duration	Specifies the setting to determine for how long a metadata record remains in the cache. Longer duration means, it takes longer to view the applied changes. The formats accepted are based on the ISO-8601 duration format. Default value: `1s` (1 second)	`5m`	Global	Metadata Cache Timeout
`cache.metadata.enabled`	boolean	Specifies the setting to enable or disable metadata caching. Supported values: • `true` • `false` Default value: `true`	`false`	Global	Metadata Caching Enabled
`database.api.enabled`	boolean	Specifies whether the Database API is enabled. Supported values: • `true` • `false` Default value: `false`	Not applicable	Global	Not applicable
`db.connectionType`	string	The type of connection. Supported values: • `basic` • `tns` • `customurl` • `databaseToolsConnection` for Database tools connections	`databaseToolsConnection`	Pool specific	Not applicable
`db.databaseToolsConnection`	string	Specifies the OCID of the pre-configured OCI database tools connection. Required for database tools type of connections.	`ocid1.databasetoolsconnection.oc1.eu-frankfurt-1.amaaaaaa....`	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
db.authProvider	string	Specifies the method of authentication to use. It can be one of oci-instance-principal or oci-profile. Defaults to oci-instance-principal. Optional for database tools type of connections.	oci-profile	Pool specific	Not applicable
db.ociProfile	string	Specifies the name of the profile in your OCI CLI configuration file for authentication. Defaults to DEFAULT. Optional for database tools type of connections, only used when the value of db.authProvider is set to oci-profile.	DEFAULT	Pool specific	Not applicable
db.customURL	string	Specifies the JDBC URL connection to connect to the database.	jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=myhost)(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=ora111.example.com)))	Pool specific	Not applicable
db.hostname	string	Specifies the host system for the Oracle database.	myhostname	Pool specific	Not applicable
db.password	string	Specifies the password of the specified database user	Not applicable	Pool specific secure setting	Not applicable
db.port	numeric	Specifies the database listener port.	1521	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`db.adminUser.password`	string	Specifies the password for the database account that ORDS uses for administration operations in the database.	Not applicable	Pool specific secure setting	Not applicable
`db.cdb.adminUser.password`	string	Specifies the password for the database account that ORDS uses for the Pluggable Database Lifecycle Management.	Not applicable	Pool specific secure setting	Not applicable
`db.servicename`	string	Specifies the network service name of the database.	`ora111.example.com`	Pool specific	Not applicable
`db.serviceNameSuffix`	string	Specifies that the pool points to a CDB, and that the PDBs connected to that CDB should be made addressable by Oracle REST Data Services (see Making All PDBs Addressable by Oracle REST Data Services (Pluggable Mapping)).	`apex_pu`	Pool specific	Service Name Suffix
`db.sid`	string	Specifies the name of the database.	`ora111`	Pool specific	Not applicable
`db.tnsAliasName`	string	Specifies the TNS alias name that matches the name in the `tnsnames.ora` file.	`MY_TNSALIAS`	Pool specific	Not applicable
`db.tnsDirectory`	string	The directory location of your `tnsnames.ora` file.	`C:\ORACLE\NETWORK\ADMIN`	Pool specific	Not applicable
`db.username`	string	Specifies the name of the database user for the connection.	`APEX_PUBLIC_USER`	Pool specific	Not applicable
`debug.printDebugToScreen`	boolean	Specifies whether to display error messages on the browser. Supported values: • `true` • `false` Default value: `false`	`false`	Global	Print Debug to Screen

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`debug.printOWADebug`	boolean	Specifies whether to display errors in PL/SQL gateway requests on the browser. If the value is set to `true`, then it overrides `debug.printDebugToScreen` value for PL/SQL gateway requests. Supported values: • `true` • `false` Default value: `false`	`false`	Pool specific	Print OWA Debug
`error.responseFormat`	string	Specifies how the HTTP error responses must be formatted. Supported values: • `html` - Force all responses to be in HTML format • `json` - Force all responses to be in JSON format • `auto` - Automatically determines most appropriate format for the request (default) • `GraphQL` - Force all responses to be formatted according to the GraphQL specification	`json`	Global	Error Response Format
`error.externalPath`	string	Specifies the path to a folder that contains the custom error page.	`/path/to/error/pages/folder/`	Global	External Error Path
`icap.port`	numeric	Specifies the Internet Content Adaptation Protocol (ICAP) port to virus scan files. Either `icap.port` or `icap.secure.port` are required to have a value.	`1344`	Global	ICAP Port

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`icap.secure.port`	numeric	Specifies the Internet Content Adaptation Protocol (ICAP) port to virus scan files. Either `icap.port` or `icap.secure.port` are required to have a value. If values for both `icap.port` and `icap.secure.port` are provided, then the value of `icap.port` is ignored.	`1344`	Global	ICAP Secure Port
`icap.server`	string	Specifies the Internet Content Adaptation Protocol (ICAP) server name or IP address to virus scan files. The `icap.server` is required to have a value.	`servername`	Global	ICAP Server
`jdbc.DriverType`	string	Specifies the JDBC driver type. Supported values: • `thin` • `oci8` Default value: `thin`	`thin`	Pool specific	JDBC Driver Type
`jdbc.InactivityTimeout`	numeric	Specifies how long an available connection can remain idle before it is closed. The inactivity connection timeout is in seconds. Defaults to `1800`.	`1800`	Pool specific	JDBC Inactivity Timeout
`jdbc.InitialLimit`	numeric	Specifies the initial size for the number of connections that will be created. Defaults to `0`. (The default is low, and should probably be set higher in most production environments.)	`10`	Pool specific	Not applicable
`jdbc.MaxConnectionReuseCount`	numeric	Specifies the maximum number of times to reuse a connection before it is discarded and replaced with a new connection. Defaults to `1000`.	`1000`	Pool specific	Max Connection Reuse Count
`jdbc.MaxLimit`	numeric	Specifies the maximum number of connections. Defaults to `10`. (Might be too low for some production environments.)	`20`	Pool specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`jdbc.auth.enabled`	boolean	Specifies if the PL/SQL Gateway calls can be authenticated using database users. If the value is `true` then this feature is enabled. If the value is `false`, then this feature is disabled. The default value is `false`. Oracle recommends not to use this feature. This feature used only to facilitate customers migrating from mod_plsql.	`false`	Pool specific	Basic Auth Enabled
`jdbc.MaxStatementsLimit`	numeric	Specifies the maximum number of statements to cache for each connection. Defaults to `10`.	`10`	Pool specific	Max Statements
`jdbc.MinLimit`	numeric	Specifies the minimum number of connections. Defaults to 2.	`1`	Pool specific	Not applicable
`jdbc.statementTimeout`	numeric	Specifies a timeout period on a statement. An abnormally long running query or script, executed by a request, may leave it in a hanging state unless a timeout is set on the statement. Setting a timeout on the statement ensures that all the queries automatically timeout if they are not completed within the specified time period. Defaults to `900`.	`900`	Pool specific	Statement Timeout
`misc.defaultPage`	string	Specifies the default page to display. The Oracle REST Data Services Landing Page.	`apex`	Pool specific	Default Page
`misc.pagination.maxRows`	numeric	Specifies the maximum number of rows that will be returned from a query when processing a RESTful service and that will be returned from a nested cursor in a result set. Affects all RESTful services generated through a SQL query, regardless of whether the resource is paginated. Defaults to `10000`.	`300`	Pool specific	Pagination Max Rows

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`owa.docTable`	string	Specifies the name of the document table used by the file upload. Defaults to `FLOWS_FILES.WWV_FLOW_FILE_OBJECTS$` value. **Note** For APEX 4.x and above this parameter should not be used.	`MYDOCTABLE`	Pool specific	Document Table
`procedure.postProcess`	string	Specifies the procedure name(s) to execute after executing the procedure specified on the URL. Multiple procedure names must be separated by commas.	`SCHEMA1.SUBMIT.REQUEST,FINISHTASK`	Pool specific	Post-Process Procedure
`procedure.preProcess`	string	Specifies the procedure name(s) to execute prior to executing the procedure specified on the URL. Multiple procedure names must be separated by commas.	`SCOTT.PREPROC1, INITIALIZE, PKG1.PROC`	Pool specific	Pre-Process Procedure

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`procedure.rest.preHook`	string	Specifies the function to be invoked prior to dispatching each Oracle REST Data Services based REST Service. The function can perform configuration of the database session, perform additional validation or authorization of the request. The function is invoked only after authorization has been granted to a protected resource. If the function returns `true`, then processing of the request continues. If the function returns `false`, then processing of the request is aborted and an `HTTP 403 Forbidden` status is returned. If authorization to the protected resource fails, the pre-hook function is not invoked. Instead, ORDS returns an `HTTP 401 Unauthorized Response` status code.	`MYAPP.VALIDATE_REST_CALL`	Pool specific	REST Pre-Hook Procedure

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`security.disableDefaultExclusionList`	boolean	If this value is set to `true`, then the Oracle REST Data Services internal exclusion list is not enforced. **Note**: The Oracle REST Data Services internal exclusion list blocks the users from accessing the following sys packages: • sys.* • dbms_* • utl_* • owa_* • owa.* • htp.* • htf.* • wpg_docload.* Supported values: • `true` • `false` Default value: `false` Oracle recommends that you do not set this value to `true`. That is, do not disable the default internal exclusion list. The only possible exception is temporarily disabling the internal exclusion list for debugging purposes.	`false`	Global	Disable Default Exclusion List
`security.exclusionList`	string	Specifies a pattern for procedures, packages, or schema names which are forbidden to be directly executed from a browser. Procedure names can contain the wildcard characters asterisk (*) or question mark (?). Use an asterisk (*) to substitute zero or more characters and a question mark (?) to substitute for any one character. **Note**: Separate multiple patterns using commas.	`customer_account,bank*,employe?`	Global	Exclusion List
`security.externalMappingPathPrefix`	boolean	Enables the use of pool identifiers of a tenant database directly within the URL of the request when trying to access it through a remote configuration.	Not applicable	Global specific	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`security.inclusionList`	string	Specifies a pattern for procedures, packages, or schema names which are allowed to be directly executed from a browser. Procedure names can contain the wildcard characters asterisk (*) or question mark (?). Use an asterisk (*) to substitute zero or more characters and a question mark (?) to substitute for any one character. **Note**: Separate multiple patterns using commas.	`apex, p, v, f, wwv_*, y*, c*`	Global	Inclusion List
`security.maxEntries`	numeric	Specifies the maximum number of cached procedure validations. Defaults to `2000`. Set this value to 0 to force the validation procedure to be invoked on each request.	`2000`	Global	Max Cached Procedure Validations
`security.requestAuthenticationFunction`	string	Specifies an authentication function to determine if the requested procedure in the URL should be allowed or disallowed for processing. The function should return `true` if the procedure is allowed; otherwise, it should return false. If it returns false, Oracle REST Data Services will return `WWW-Authenticate` in the response header.	`owa_custom.authorize`	Pool specific	Request Authentication Function
`security.requestValidationFunction`	string	Specifies a validation function to determine if the requested procedure in the URL should be allowed or disallowed for processing. The function should return true if the procedure is allowed; otherwise, return false.	`CHECK_VALID_PROCEDURE`	Pool specific	Request Validation Function
`security.verifySSL`	boolean	Specifies whether HTTPS is available in your environment. Supported values: • `true` • `false` Default value: `true` If you change the value to `false`, see Using OAuth2 in Non-HTTPS Environments.	`true`	Global	Verify SSL

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`restEnabledSql.active`	boolean	Specifies whether the REST-Enabled SQL service is active. Supported values: • `true` • `false` Default value: `false`	`true`	Pool specific	Not applicable
`apex.security.developer.roles`	string	Comma delimited list of additional roles to assign authenticated APEX developer type users. Default value: SQL Developer, OAuth Client Developer	Not applicable	Pool specific	APEX Security Developer Roles
`db.idlePoolTimeout`	duration	Specifies for how long to wait before destroying an idle pool.	Not applicable	Global	Idle Pool Timeout
`feature.sdw.selfServiceSchema`	boolean	Specifies enable self service schema feature. Default value: `false`	Not applicable	Pool specific	Self Service Schema Enabled
`http.cookie.filter.byValue`	string	Specifies a comma separated list of HTTP Cookie values to exclude when initializing an Oracle Web Agent environment.	Not applicable	Pool specific	HTTP Cookie Filter by Value
`jdbc.ConnectionWaitTimeout`	numeric	Specifies the wait time (in seconds) for a used connection to be released by a client. Default value: 3	Not applicable	Pool specific	JDBC Connection Wait Timeout
`jdbc.MaxConnectionReuseTime`	duration	Specifies the maximum time of a connection that can be reused before it is discarded and replaced with a new connection. Default value: `60s`	Not applicable	Pool specific	Max Connection Reuse Time
`jdbc.driverName`	string	Specifies the name of the JDBC driver to use.	Not applicable	Pool specific	Driver Name
`jdbc.ucp.enableJMX`	boolean	Enables JMX-Based Management in UCP. Default value: `true`	Not applicable	Pool specific	UCP JMX Enabled
`json.sdo.geometry.output.geojson`	boolean	Specifies that SDO Geometry data must be returned in GeoJSON format. Default value: `false`	Not applicable	Pool specific	SDO Geometry Output GeoJSON
`mongo.access.log`	string	Specifies the path to the folder to store API for MongoDB access logs. If `mongo.access.log` is not specified, then the Mongo access log is not generated.	`/path/to/mongo/logs`	Global	MongoDB Access Log

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`mongo.enabled`	boolean	Enables the API for MongoDB when set to true. Default value: `false`	`true`	Global	MongoDB Enabled
`mongo.host`	string	Comma separated list of host names or IP addresses to identify a specific network interface on which to listen. Default value: `0.0.0.0`	`example.com`	Global	MongoDB Host
`mongo.idle.timeout`	duration	Specifies the maximum idle time for a connection. Default value (in minutes): `30m`	45m	Global	MongoDB Idle Timeout
`mongo.op.timeout`	duration	The maximum time a single request will wait on a database operation. Default value (in minutes): `10m`	15m	Global	MongoDB Operation Timeout
`mongo.port`	numeric	Specifies listen port of API for MongoDB Wire protocol. Default value: `27017`	Not applicable	Global	MongoDB Port
`mongo.tls`	boolean	Enables TLS for the API for MongoDB. Note, disabling TLS is strongly discouraged as third parties may be able to observe credentials transmitted by MongoDB clients Default value: `true`	Not applicable	Global	MongoDB TLS
`oracle.jdbc.vectorDefaultGetObjectType`	string	Sets the default return type of `getObject()` methods when the column type is VECTOR. See Oracle JDBC documentation for more information. Default value: `String`	Not applicable	Pool specific	Vector Default Get Object Type
`public.properties.url`	string	Specifies the URL to retrieve the global public properties.	Not applicable	Global	Public Properties URL
`security.externalHostMappingHeader`	string	Indicates the header name for identifying a tenant database when trying to access it through a remote configuration.	Not applicable	Global	Not applicable
`security.externalSessionTrustedOrigins`	string	Specifies a comma separated list of origins that are trusted to make Cross-Origin Resource Sharing (CORS) request. If this setting is empty or not configured, then no CORS requests are allowed for the PL/SQL gateway and results in a 403 Unauthorized status.	Not applicable	Global	External Session Trusted Origins

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`security.httpsHeaderCheck`	string	Specifies the HTTP request header and value that indicates to ORDS the initial client request was received by a Load Balancer / Reverse Proxy over HTTPS even if the request received by ORDS was sent over HTTP.	`X-Forwarded-Proto: https`	Global	HTTPS Header Check
`security.host.headers`	string	Specifies the comma-delimited list of the allowed host names for the requests made to ORDS.	Not applicable	Global	Allowed Host Headers
`security.par.enabled`	boolean	Enables or disables the PAR feature. Default value: `true`	Not applicable	Pool specific	PAR Enabled
`standalone.access.log.retainDays`	numeric	Specifies the number of days before the rotated access log files are deleted. Default value (in days): `90`	Not applicable	Global	Access Log Retention Days
`feature.mcp`	boolean	Specifies whether the ORDS MCP endpoint is enabled. Supported values: • `true` • `false` Default value: `false`	`true`	Global	Not applicable
`mcp.security.jwt.profile.issuer`	string	Specifies the issuer for MCP JWT access tokens. ORDS validates this value against the token `iss` claim.	`https://idp.example.com`	Global	Not applicable
`mcp.security.jwt.profile.audience`	string	Specifies the intended audience of MCP JWT access tokens. The audience is typically the ORDS MCP resource URI.	`https://ords.example.com/mcp`	Global	Not applicable
`mcp.security.jwt.profile.jwk.url`	string	Specifies the URL of the JSON Web Key Set (JWKS) used to validate MCP JWT access token signatures.	`https://idp.example.com/.well-known/jwks.json`	Global	Not applicable

Copyright © 2011, 2026, Oracle and/or its affiliates.

Table D-1 (Cont.) Oracle REST Data Services Configuration Settings

Key	Type	Description	Example	Setting Type	OCI First Class Property or Label
`mcp.security.jwt.profile.authorization.server.url`	string	Specifies the optional authorization server URL used in OAuth protected-resource metadata for `/mcp`. If omitted, ORDS does not expose `authorization_servers`, preserving authorization server privacy while requiring MCP clients to obtain authorization server details through separate configuration.	`https://idp.example.com`	Global	Not applicable
`mcp.security.jwt.profile.role.claim.name`	string	Specifies the optional JSON Pointer for the JWT roles claim. Setting this property enables role mode; otherwise, ORDS MCP uses scope mode.	`/roles`	Global	Not applicable
`mcp.scope`	string	Specifies a pool-specific MCP scope. In scope mode, access to the corresponding direct pool is restricted to caller tokens that contain the scope. Pools configured with this property are dedicated to MCP and are not considered for REST URL mapping.	`urn:oracle:dbtools:ords:mcpserver:all`	Pool specific	Not applicable
`mcp.role`	string	Specifies a pool-specific MCP role. In role mode, access to the corresponding direct pool is restricted to caller tokens that contain the role. Pools configured with this property are dedicated to MCP and are not considered for REST URL mapping.	`ORDS_MCP_HR`	Pool specific	Not applicable
`db.description`	string	Specifies the optional description for the pool. ORDS MCP returns this value from `database_list` when the caller requests details.	`HR reporting database`	Pool specific	Not applicable

See Also

SODA Entries In ORDS Configuration File

Copyright © 2011, 2026, Oracle and/or its affiliates.

E

ORDS Central Configuration OpenAPI


```
{
  "openapi": "3.0.3",
  "info": {
    "title": "Central Configuration Server API",
    "version": "2024.07.01",
    "description": "<p>Last Update: July 2024</p>\n\nThe Central
Configuration Server API  provides comprehensive endpoints that allows ORDS
(Oracle Rest Data Services) to retrieve global and pool-specific
configurations from the central configuration server. Use this API to setup
and configure your own central configuration server for ORDS.<p> With
centralized configurations ORDS nodes will be configured to point to a
centralized configuration URL that provides access to the configuration
state, rather than reading the configuration state from a disk based folder.
</p> <p>The configuration server will be specified via a Java System property
named config.url .  Access to the centralized configuration URL will need to
be secured. Another system property config.wallet will be used to provide the
credentials to be used when connecting to the centralized configuration URL.</
p>",
    "license": {
      "name": "ORDS License",
      "url": "https://www.oracle.com/rest"
    }
  },
  "externalDocs": {
    "description": "Find out more about ORDS",
    "url": "https://www.oracle.com/rest"
  },
  "servers": [
    {
      "url": "https://central-config.example.com",
      "description": "Central Configuration Server"
    }
  ],
  "tags": [
    {
      "name": "General",
      "description": "General Endpoints"
    }
  ],
  "paths": {
    "/central/v1/config": {
      "get": {
        "tags": [
          "General"
        ],
        "summary": "Retrieve global configuration",
        "description": "<p>Retrieves the global configuration stored in the
central configuration server. The global configuration contains settings that
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
apply across the entire ORDS instance.</p><p> Configuration use the standard
JSON properties format, where each configuration setting contains a key and a
corresponding value.</p>",
        "operationId": "getGlobalConfig",
        "parameters": [
          {
            "$ref": "#/components/parameters/AcceptHeader"
          },
          {
            "$ref": "#/components/parameters/AuthorizationHeader"
          },
          {
            "$ref": "#/components/parameters/RequestIdHeader"
          }
        ],
        "responses": {
          "200": {
            "description": "Global configuration returned successfully",
            "content": {
              "application/json": {
                "schema": {
                  "$ref": "#/components/schemas/GlobalConfigResponse"
                }
              }
            }
          },
          "401": {
            "description": "Unauthorized - Invalid or missing
authentication.Bearer access token was not provided in Authorization header,
token could not be decoded, or token was invalid.",
            "content": {
              "application/json": {
                "example": {
                  "message": "string",
                  "code": "string"
                }
              }
            }
          },
          "404": {
            "description": "Not Found - Global configuration not found",
            "content": {
              "application/json": {
                "example": {
                  "message": "string",
                  "code": "string"
                }
              }
            }
          }
        }
      }
    },
    "/central/v1/config/pool/{host}": {
      "get": {
        "tags": [
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
            "General"
          ],
          "summary": "Retrieve pool-specific configuration",
          "description": "<p> Retrieves the configuration for a specific pool
from the central server.The pool configuration file contains informations
about a pool. Configuration files use the standard JSON properties format,
where each configuration setting contains a key and a corresponding
value.</p> <p>Note that the response should include an ETag which changes if
the configuration changes.</p>",
          "operationId": "getPoolConfig",
          "parameters": [
            {
              "$ref": "#/components/parameters/AcceptHeader"
            },
            {
              "$ref": "#/components/parameters/AuthorizationHeader"
            },
            {
              "$ref": "#/components/parameters/RequestIdHeader"
            },
            {
              "$ref": "#/components/parameters/IfNoneMatchHeader"
            },
            {
              "name": "host",
              "in": "path",
              "required": true,
              "description": "Identifier for the pool whose configuration is to
be retrieved.",
              "schema": {
                "type": "string",
                "example": "mydatabase"
              }
            }
          ],
          "responses": {
            "200": {
              "description": "Pool configuration returned successfully",
              "headers": {
                "ETag": {
                  "description": "ETag for the returned configuration",
                  "schema": {
                    "type": "string"
                  }
                }
              },
              "content": {
                "application/json": {
                  "schema": {
                    "$ref": "#/components/schemas/PoolConfigResponse"
                  }
                }
              }
            },
            "304": {
              "description": "Not Modified - The configuration has not changed
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
since the last request",
            "content": {
              "application/json": {
                "example": {
                  "message": "string",
                  "code": "string"
                }
              }
            }
          },
          "401": {
            "description": "Unauthorized - Invalid or missing
authentication.Bearer access token was not provided in Authorization header,
token could not be decoded, or token was invalid.",
            "content": {
              "application/json": {
                "example": {
                  "message": "string",
                  "code": "string"
                }
              }
            }
          },
          "404": {
            "description": "Not Found - No matching mapping is found for the
requested pool",
            "content": {
              "application/json": {
                "example": {
                  "message": "string",
                  "code": "string"
                }
              }
            }
          }
        }
      }
    }
  },
  "components": {
    "parameters": {
      "AcceptHeader": {
        "name": "Accept",
        "in": "header",
        "required": false,
        "description": "Indicates the media type(s) that ORDS is able to
understand. The central configuration server uses this header to select an
appropriate response format.",
        "schema": {
          "type": "string",
          "example": "application/json"
        }
      },
      "AuthorizationHeader": {
        "name": "Authorization",
        "in": "header",
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
        "required": true,
        "description": "Contains the credentials to authenticate ORDS with
the server. Typically used to pass a Bearer token.",
        "schema": {
          "type": "string",
          "example": "Bearer abcdef123456"
        }
      },
      "RequestIdHeader": {
        "name": "Request-Id",
        "in": "header",
        "required": true,
        "description": "A unique identifier for the request, which can be
used for tracking and debugging purposes. This ID helps correlate ORDS
requests with server logs.",
        "schema": {
          "type": "string",
          "example": "123e4567-e89b-12d3-a456-426614174000"
        }
      },
      "IfNoneMatchHeader": {
        "name": "If-None-Match",
        "in": "header",
        "required": true,
        "description": "Used to make conditional requests based on the ETag
(entity tag) of a resource. If the ETag matches the current version of the
resource, the server responds with a 304 Not Modified status instead of
returning the resource.",
        "schema": {
          "type": "string",
          "example": "34f7a3113803f8ed3b8fd7ce5656ebec"
        }
      }
    },
    "schemas": {
      "GlobalConfigResponse": {
        "type": "object",
        "properties": {
          "settings": {
            "type": "object",
            "additionalProperties": {
              "type": "string"
            },
            "description": "Key-value pairs of configuration settings. Under
\"Settings\", you can define your custom settings. Please refer to the
following documentation for a better understanding of what configuration
settings are available for the global configuration file [Oracle REST Data
Services Configuration Files](https://docs.oracle.com/en/database/oracle/
oracle-rest-data-services/24.2/ordig/about-REST-configuration-files.html).",
            "example": {
              "restEnabledSql.active": true,
              "feature.sdw": true,
              "security.externalHostMappingHeader": "poolname"
            }
          },
          "links": {
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
            "type": "array",
            "items": {
              "type": "object",
              "properties": {
                "rel": {
                  "type": "string",
                  "description": "Indicates the relationship of the link to
the current context."
                },
                "href": {
                  "type": "string",
                  "description": "Specifies the URL associated with the link."
                },
                "templated": {
                  "type": "boolean",
                  "description": "Specifies whether the URL is templated
(contains placeholders) or not."
                }
              },
              "required": [
                "rel",
                "href"
              ],
              "example": [
                {
                  "rel": "collection",
                  "href": "http://central-config.example.com:8585/central/v1/
config/"
                },
                {
                  "rel": "self",
                  "href": "http://central-config.example.com:8585/central/v1/
config/"
                },
                {
                  "rel": "search",
                  "href": "http://central-config.example.com:8585/central/v1/
config/pool/{host}",
                  "templated": true
                }
              ]
            },
            "description": "Under \"links\" (Array), each entry represents a
hyperlink related to the configuration, with specific relationships (e.g.,
\"collection\", \"self\", \"search\") and associated URLs. The \"search\"
link is marked as templated, indicating that it contains a placeholder {host}
which can be replaced with an actual value when making a request.\nThis array
contains links relevant to the configuration\n- **rel** (String) Indicates
the relationship of the link to the current context.\n- **href** (String)
Specifies the URL associated with the link.\n- **templated** (Boolean)
Specifies whether the URL is templated (contains placeholders) or not.\n"
          }
        }
      },
      "PoolConfigResponse": {
        "type": "object",
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

```
        "properties": {
          "database": {
            "type": "object",
            "properties": {
              "pool": {
                "type": "object",
                "properties": {
                  "name": {
                    "type": "string"
                  },
                  "settings": {
                    "type": "object",
                    "additionalProperties": {
                      "type": "string"
                    },
                    "example": {
                      "db.wallet.zip":
"UEsDBBQACAgIAOMid1IAAAAAAAAAAAAAAAALAAAAY3dhbGxldC5zc28BdRuH4TjA...",
                      "feature.sdw": true,
                      "plsql.gateway.mode": "proxied",
                      "restEnabledSql.active": true
                    },
                    "description": "Please refer to the following
documentation for better understanding of what configuration settings are
available for the pool configuration file [Understanding the Configurable
Settings](https://docs.oracle.com/en/database/oracle/oracle-rest-data-
services/24.2/ordig/about-REST-configuration-files.html)\n"
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```


Copyright © 2011, 2026, Oracle and/or its affiliates.

Index

A

Apache Tomcat, *3*
  about, *12*
  configuring Oracle REST Data Services
    images, *12*
  deploying to, *11*
  downloading, *12*

C

configuration file editable parameters
  jdbc.MaxConnectionReuseCount, *D-16*
  jdbc.MinLimit, *D-17*
  security.requestValidationFunction, *D-21*
  security.verifySSL, *D-21*
configuration files, *D-1*
configuration folder
  structure of, *D-1*

D

database users, *8*
deploy options
  Apache Tomcat, *11*
deployment options
  Oracle WebLogic Server, *6*
DER
  converting private key to DER, *4*
downloading
  Apache Tomcat, *12*
  Oracle WebLogic Server, *6*

J

Java EE application servers
  about supported, *3*

Java EE application servers *(continued)*

M

multiple database configuration
  about the request URL, *1*
  configuring additional databases, *2*

O

OAuth2, default behavior, *16*
Oracle Cloud Infrastructure
  about, *14*
Oracle REST Data Services
  configuration files, *D-1*
  configuring, *1*
  database users, *8*
  Deploying, *1*
  system requirements, *1*
Oracle WebLogic Server, *3*
  about, *6*
  deploy to, *6*
  downloading, *6*
  installing, *6*
  installing the deployment, *8*

P

private key
  converting to DER, *4*

S

structure of configuration folder, *D-1*
supported Java EE application servers, *3*
system requirements, *1*

Copyright © 2011, 2026, Oracle and/or its affiliates.