Oracle® Cloud

Oracle Machine Learning Administration Guide

G15788-03
May 2025

Oracle Cloud Oracle Machine Learning Administration Guide,

G15788-03

Primary Author: Moitreyee Hazarika

Contributors: Mark Hornick, Sherry LaMonica, Philip Cannata

Contents

Part I

Administering OML in Oracle Autonomous Database

Oracle Machine Learning is managed at the system level and at the application level by an administrator.

Topics include:

- Overview
- Administrative Tasks for Oracle Machine Learning on Autonomous Database
  Lists the administrative tasks related to Oracle Machine Learning on Autonomous Database. These tasks require the `ADMIN` role.

Overview

- What is Oracle Autonomous Database?
  Autonomous Database provides an easy-to-use, fully autonomous database that scales elastically and delivers fast query performance. As a service, Autonomous Database does not require database administration.
- Key Features of Autonomous Database
  Provides information on key features of Autonomous Database, an affordable, feature-rich service in the cloud.

What is Oracle Autonomous Database?

Autonomous Database provides an easy-to-use, fully autonomous database that scales elastically and delivers fast query performance. As a service, Autonomous Database does not require database administration.

With Autonomous Database you do not need to configure or manage any hardware or install any software. Autonomous Database handles provisioning the database, backing up the database, patching and upgrading the database, and growing or shrinking the database. Autonomous Database is a completely elastic service.

At any time you can scale, increase or decrease, either the compute or the storage capacity. When you make resource changes for your Autonomous Database instance, the resources automatically shrink or grow without requiring any downtime or service interruptions.

Autonomous Database is built upon Oracle Database, so that the applications and tools that support Oracle Database also support Autonomous Database. These tools and applications connect to Autonomous Database using standard SQL*Net connections. The tools and applications can either be in your data center or in a public cloud. Oracle Analytics Cloud and other Oracle Cloud services provide support for Autonomous Database connections.

Autonomous Database provides the following:

- **Increase your app development velocity**: Build and deploy quickly on unified platform supporting vector, JSON, graph, and more. Leverage scalable, cost-effective solutions with flexible cloud, multi-cloud, or on-premises deployment, advanced security, and self-service tools.
- **Fully automated database service**: Supports all modern Oracle data types, workloads including transaction processing, AI and analytics, reducing the need for multiple specialty databases.
- **Maximizes data security and availability**: With Oracle's automation services, you can keep your mission critical applications running by leveraging 99.995% availability.
- **Allows you to bring AI to your data**: Use LLMs with built-in AI Vector Search and your proprietary data to get more accurate answers. There's no need to duplicate data to a separate vector database.

Autonomous Database also includes the following:

- **Oracle APEX**: a low-code development platform that enables you to build scalable, secure enterprise apps with world-class features.

  See About Oracle APEX for more information.
- **Oracle REST Data Services (ORDS)**: a Java Enterprise Edition based data service that makes it easy to develop modern REST interfaces for relational data and JSON Document Store.

  See Developing RESTful Services in Autonomous Database for more information.
- **Database Actions**: is a web-based interface that uses Oracle REST Data Services to provide development, data tools, administration, and monitoring features for Autonomous Database.

  See About Database Actions for more information.
- **Oracle Machine Learning**: is a set of components supporting data scientists, ML engineers, and data analysts, as well as SQL, R, and Python users. Components include SQL, R, and Python APIs, a built-in notebook interface, and no-code interfaces for AutoML, data and model monitoring, and model deployment, along with model management, deployment, and monitoring via REST endpoints.

  See Oracle Machine Learning for more information.

Key Features of Autonomous Database

Provides information on key features of Autonomous Database, an affordable, feature-rich service in the cloud.

Key Features

- **Managed**: Oracle simplifies end-to-end management of the database:
  - Provisioning new databases
  - Growing or shrinking storage and compute resources
  - Patching and upgrades
  - Backup and recovery
- **Fully Tuned**: "Load and go":
  - Define tables, load data, run queries
  - Provides good performance out of the box

  - Run your queries using any business analytics tool or cloud service
  - Built-in SQL worksheet and notebook also included
- **Fully elastic scaling**: Scale compute and storage independently to fit your database workload with no downtime:
  - Size the database to the exact compute and storage required
  - Scale the database on demand: Independently scale compute or storage
  - Shut off idle compute to save money
- **Auto scaling**: Allows your database to use more CPU and IO resources or to use additional storage automatically when the workload or storage demand requires additional resources:
  - Specify the number of CPUs for your Autonomous Database workload.
  - Use compute auto scaling to allow the database to use up to three times more CPU and IO resources, depending on workload requirements. Compute auto scaling is enabled by default when you create an Autonomous Database.
  - Use storage auto scaling to allow the database to expand to use up to three times the reserved base storage, depending on your storage requirements. Storage auto scaling is disabled by default when you create an Autonomous Database.
  - Manage auto scaling from the Oracle Cloud Infrastructure Console to enable or disable compute auto scaling or storage auto scaling for your Autonomous Database.
- **Autonomous Database supports**:
  - Existing applications, running in the cloud or on premises
  - Connectivity via SQL*Net, JDBC, ODBC

    See Connect to Autonomous Database for more information.
  - Third-party data-integration tools
  - Oracle cloud services: Oracle Analytics Cloud, Oracle GoldenGate Marketplace, and others
- **High-performance queries and concurrent workloads**: Optimized query performance with preconfigured resource profiles for different types of users.
- **Oracle SQL**: Autonomous Database is compatible with existing applications that support Oracle Database.
- **Built-in web-based data analysis tool**: Web-based notebook tool for designing and sharing SQL based data-driven, interactive documents.
- **Database migration utility**: Easily migrate from MySQL, Amazon AWS Redshift, PostgreSQL, SQL Server, and other databases.

Simple Cloud-based Data Loading

Autonomous Database provides:

- Fast, scalable data-loading from Oracle Cloud Infrastructure Object Storage, Azure Blob Storage or Azure Data Lake Storage, Amazon S3, Amazon S3-Compatible, GitHub Repository, Google Cloud Storage, or on-premises data sources.

Oracle Database Actions

Database Actions is a web-based interface that uses Oracle REST Data Services to provide development, data tools, and administration and monitoring features for Autonomous Database, including the following:

- **Development Tools**
  - SQL Navigator and Worksheet: view objects and enter and run SQL and PL/SQL statements, and create database objects
  - Data Modeler: provides an integrated version of Oracle SQL Developer Data Modeler with basic reporting features. You can create diagrams from existing schemas, retrieve data dictionary information, generate DDL statements, and export diagrams
  - REST: An IDE for your REST APIs that enables you to manage templates, handlers and OAuth clients, generate API documentation, and test APIs.
  - LIQUIBASE: View ChangeLogs applied to your schema.
  - JSON: Create collections, upload documents, query and filter your data, create diagrams for your JSON document structures, and create relational views and indexes.
  - Charts: Use SQL queries to build rich charts and dashboards containing multiple charts.
  - Scheduling: An interface for `DBMS_SCHEDULER` that enables you to monitor jobs, view execution history, forecast upcoming jobs, and visualize scheduler chains.
  - Oracle Machine Learning: provides several components accessible through a common user interface. OML Notebooks supports Python, R, SQL, PL/SQL, and Markdown interpreters, with access to in-database ML through OML4Py, OML4R, and OML4SQL. OML Models supports managing and deploying in-database models. OML AutoML UI provides a no-code user interface to build, evaluate, and deploy in-database models using automated machine learning.
  - APEX: Login to APEX, develop and run rich, low-code web applications.
  - Graph Studio: Oracle Graph Studio lets you create property graph databases and automates the creation of graph models and in-memory graphs from database tables.
- **Data Studio**
  - Data Load: load or access data from local files or remote databases.
  - Catalog: understand data dependencies and the impact of changes.
  - Data Insights: discover anomalies, outliers and hidden patterns in your data.
  - Data Analysis: analyze your data
  - Data Transforms: transform data for analysis and other applications.
- **Administration and Monitoring**
  - Manage users
  - Database Dashboard: Monitor database activity charts such as CPU usage, number of executing SQL statements, and wait events formerly found on your Autonomous Database Service Console.
  - Performance Hub: Access SQL Monitoring reports and Active Session History (ASH) Analytics.

Disaster Recovery Options

Autonomous Data Guard Autonomous Database provides Autonomous Data Guard to enable a standby (peer) database to provide data protection and disaster recovery for your Autonomous Database instance.When you add an Autonomous Data Guard standby database, the system creates a standby database that continuously gets updated with the changes from the primary database. You can use Autonomous Data Guard with a standby in the current region, a local standby, or with a standby in a different region, a cross-region standby. You can also use Autonomous Data Guard with both a local standby and a cross-region standby.

Backup-Based Disaster Recovery uses backups to instantiate a peer database at the time of switchover or failover. This enables you to have a lower cost and higher Recovery Time Objective (RTO) disaster recovery option for your Autonomous Database, as compared with Autonomous Data Guard. For local backup-based disaster recovery, existing local backups are utilized. There are no additional costs for a local Backup-Based Disaster Recovery. Cross-Region Backup-Based Disaster Recovery incurs an additional cost.

SQL Developer Support

Using Autonomous Database with SQL Developer you can do the following:

- Connect to Autonomous Database
- Create tables, indexes, and materialized views in Autonomous Database
- Load data into an Autonomous Database
- Copy tables to Autonomous Database
- Transfer a schema to Autonomous Database

Business Intelligence Tools Support

Autonomous Database is compatible with a number of business intelligence and data visualization tools from Oracle and from trusted third parties:

- Oracle Analytics Cloud
- Oracle Analytics Desktop
- Third-party Business Intelligence tools

Administrative Tasks for Oracle Machine Learning on Autonomous Database

Lists the administrative tasks related to Oracle Machine Learning on Autonomous Database. These tasks require the `ADMIN` role.

The Admin creates and manages user accounts for Oracle Machine Learning, creates users on Autonomous Database, manages user profiles on Autonomous Database, and manages user roles and privileges.

Table Administrative Tasks for OML on Autonomous Database

Tasks	Interfaces/Tools	Links
Access Oracle Machine Learning User Management from Command Line	Oracle Cloud Infrastructure (OCI) Command Line Interface (CLI)	• Access OML User Management from Command Line
Create and Manage OML users for Oracle Machine Learning	Oracle Machine Learning User Management interface	• Create Users for Oracle Machine Learning • Add Existing Database Users to Oracle Machine Learning
Grant and Manage User Roles and Privileges	Database Actions - Autonomous Database	• Manage Users and User Roles on Autonomous Database - Connecting with Database Actions • Create and Manage Directories • Managing Users and User Roles on Autonomous Database with Database Actions • Manage User Privileges on Autonomous Database - Connecting with a Client Tool • Managing User Privileges on Autonomous Database
Manage Admin Account	Oracle Cloud Infrastructure Console	• Set the ADMIN Password in Autonomous Database • Unlock the ADMIN Account in Autonomous Database • Use OCI Vault Secret for ADMIN Password
Manage User Data	Admin interface in Oracle Machine Learning User Interface	• User Data • Reassign
Manage Compute Resource	Admin interface in Oracle Machine Learning User Interface	• Compute Resource • Oracle Resource
Manage Connection Groups	Admin interface in Oracle Machine Learning User Interface	• About Connection Groups • About Global Connection Group • Edit Oracle Database Interpreter Connection
Manage notebook sessions	Admin interface in Oracle Machine Learning User Interface	Manage notebook sessions
Manage Email addresses for Job notifications	Jobs interface Oracle Machine Learning User Interface	Change the variable `oml.emails.maxRecipients` in the `oml.conf` file. By default, a user can send email notifications to 3 email addresses.

Part II
Administering OML for Base Database

Oracle Base Database Service enables you to maintain absolute control over your data while using the combined capabilities of Oracle Database and Oracle Cloud Infrastructure.

Topics include:

- Overview
  Oracle Base Database Service enables you to maintain absolute control over your data while using the combined capabilities of Oracle Database and Oracle Cloud Infrastructure.
- Administrative Tasks for Oracle Machine Learning on Base Database
  Lists the administrative tasks related to Oracle Machine Learning on Oracle Base Database. These tasks require the `ADMIN` role.

Overview

Oracle Base Database Service enables you to maintain absolute control over your data while using the combined capabilities of Oracle Database and Oracle Cloud Infrastructure.

Oracle Base Database Service offers database systems (DB systems) on virtual machines. They are available as single-node DB systems and multi-node RAC DB systems on Oracle Cloud Infrastructure (OCI). You can manage these DB systems by using the OCI Console, the OCI API, the OCI CLI, the Database CLI (DBCLI), Enterprise Manager, or SQL Developer.

> **Note:**
>
> This documentation is intended for Oracle Database administrators and assumes familiarity with Oracle Database and tools.

- About Oracle Base Database Service
  Oracle Base Database Service enables you to maintain absolute control over your data while using the combined capabilities of Oracle Database and Oracle Cloud Infrastructure.
- About Virtual Machine DB Systems
  Oracle Cloud Infrastructure (OCI) offers DB systems on virtual machines.
- Connecting to a DB System
  Oracle Base Database Service offers database systems (DB systems) on virtual machines. This topic shows how to connect to a DB system from Oracle Base Database Service using the OCI console.

About Oracle Base Database Service

Oracle Base Database Service enables you to maintain absolute control over your data while using the combined capabilities of Oracle Database and Oracle Cloud Infrastructure.

Oracle Base Database Service offers database systems (DB systems) on virtual machines. They are available as single-node DB systems and multi-node RAC DB systems on Oracle

Cloud Infrastructure (OCI). You can manage these DB systems by using the OCI Console, the OCI API, the OCI CLI, the Database CLI (DBCLI), Enterprise Manager, or SQL Developer.

> **Note:**
>
> This documentation is intended for Oracle Database administrators and assumes familiarity with Oracle Database and tools.

About Virtual Machine DB Systems

Oracle Cloud Infrastructure (OCI) offers DB systems on virtual machines.

There are two types of database systems (DB systems) on virtual machines:

- Single-node DB system: A 1-node DB system consists of one virtual machine.
- Multi-node RAC DB system: A 2-node DB system consists of two virtual machines.

If you must provision a DB system for development or testing purposes, a special fast-provisioning single-node DB system is available.

When you create a DB system, you select the Oracle Database edition and version that applies to the database on that DB system. You cannot change the selected edition. Depending on your selected Oracle Database edition and version, your DB system can support multiple pluggable databases (PDB). See the following Oracle Database licensing topic for information about the maximum number of pluggable and container databases (CDB) available for your selected Oracle Database version.

- Oracle Database 19c: Permitted Features, Options, and Management Packs by Oracle Database Offering

A DB system can have only a single Database Home, which in turn can have only a single database. A DB system database uses OCI block storage instead of local storage. You specify a storage size when you create the DB system, and you can scale up the storage as required at any time. To change the number of CPU cores on an existing DB system, you must change the shape of that DB system. For more information, see *Change the Shape of a DB System*.

> **Note:**
>
> The shape change operation takes place in a rolling fashion for multi-node RAC DB systems, enabling you to change the shape with no database downtime.

Connecting to a DB System

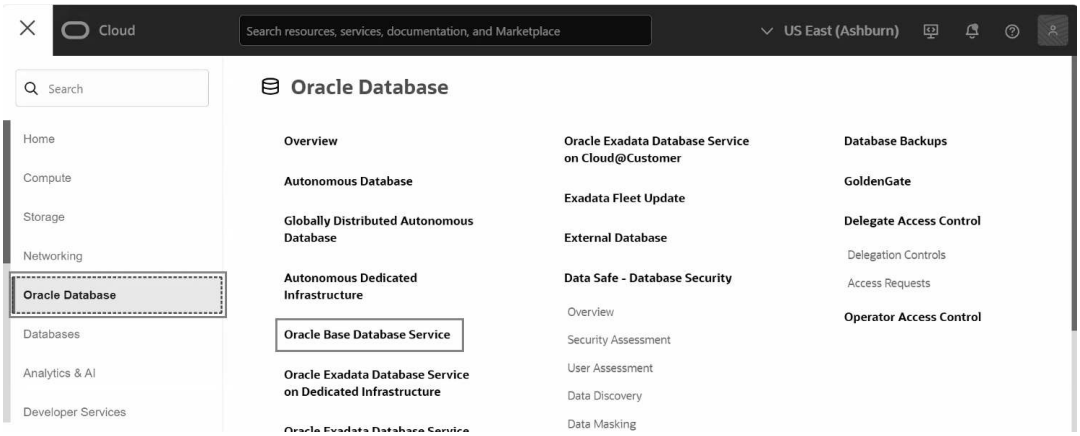
Oracle Base Database Service offers database systems (DB systems) on virtual machines. This topic shows how to connect to a DB system from Oracle Base Database Service using the OCI console.

DB systems are available as single-node DB systems and multi-node RAC DB systems on Oracle Cloud Infrastructure (OCI). You can manage these DB systems by using the OCI Console, the OCI API, the OCI CLI, the Database CLI (DBCLI), Enterprise Manager, or SQL Developer.

To connect to a DB System from Oracle Base Database Service using the OCI console:

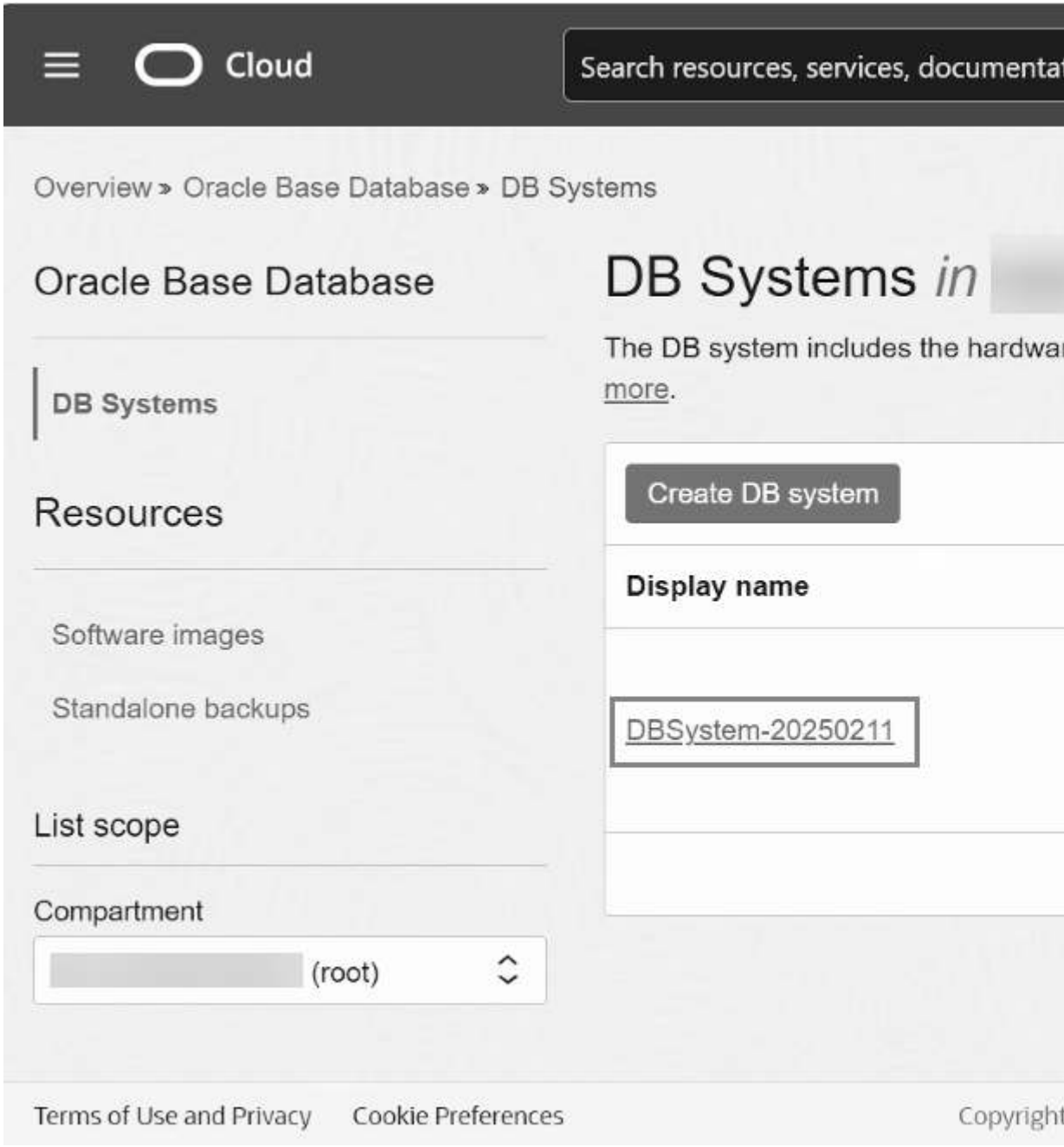
1. Go to your OCI console and on the top left, click on the **Navigation menu**.
2. On the left navigation menu, click **Oracle Database** and then click **Oracle Base Database Service.** This lists the DB systems in the Oracle Base Database that you are accessing.

Figure Oracle Base Database Service



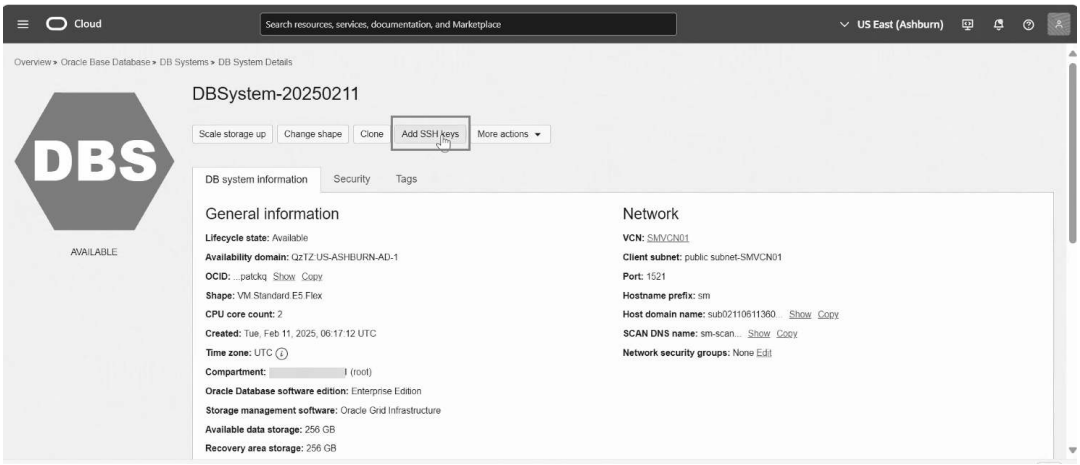
3. Select your compartment on the left pane. On the right, click the DB system that you want to access.

Figure List of DB systems



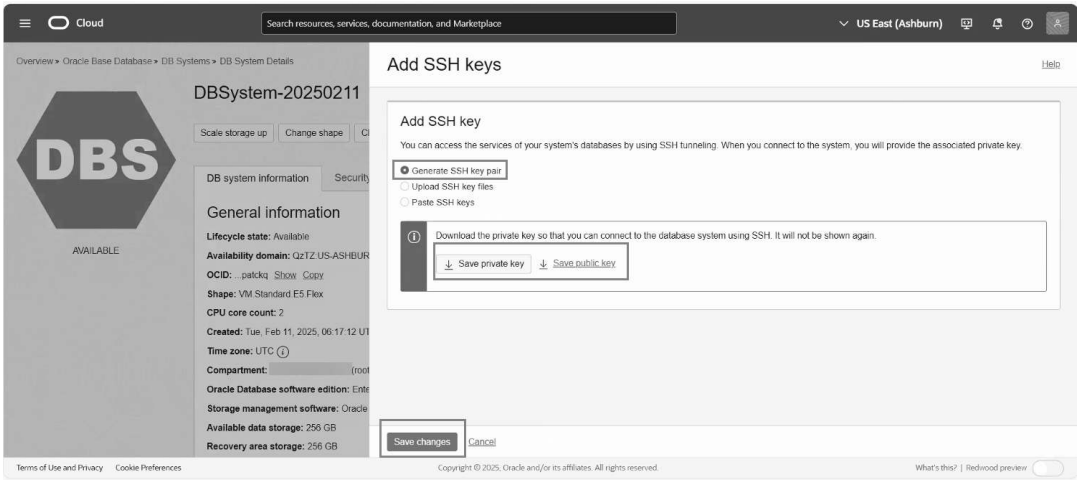
4. The DB System information page opens. On the top, click **Add SSH Keys.** The Add SSH Keys pane opens.

Figure Add SSH Keys option on DB Systems Information page



5. On the Add SSH Keys pane, click **Generate SSH Key Pairs.**

Figure Add SSH Keys pane



- **Generate SSH key pair:** Use this option to create a new SSH key pair. Click both **Save private key** and **Save public key** when using this option. The private key is downloaded to your local machine, and should be stored in a safe location.

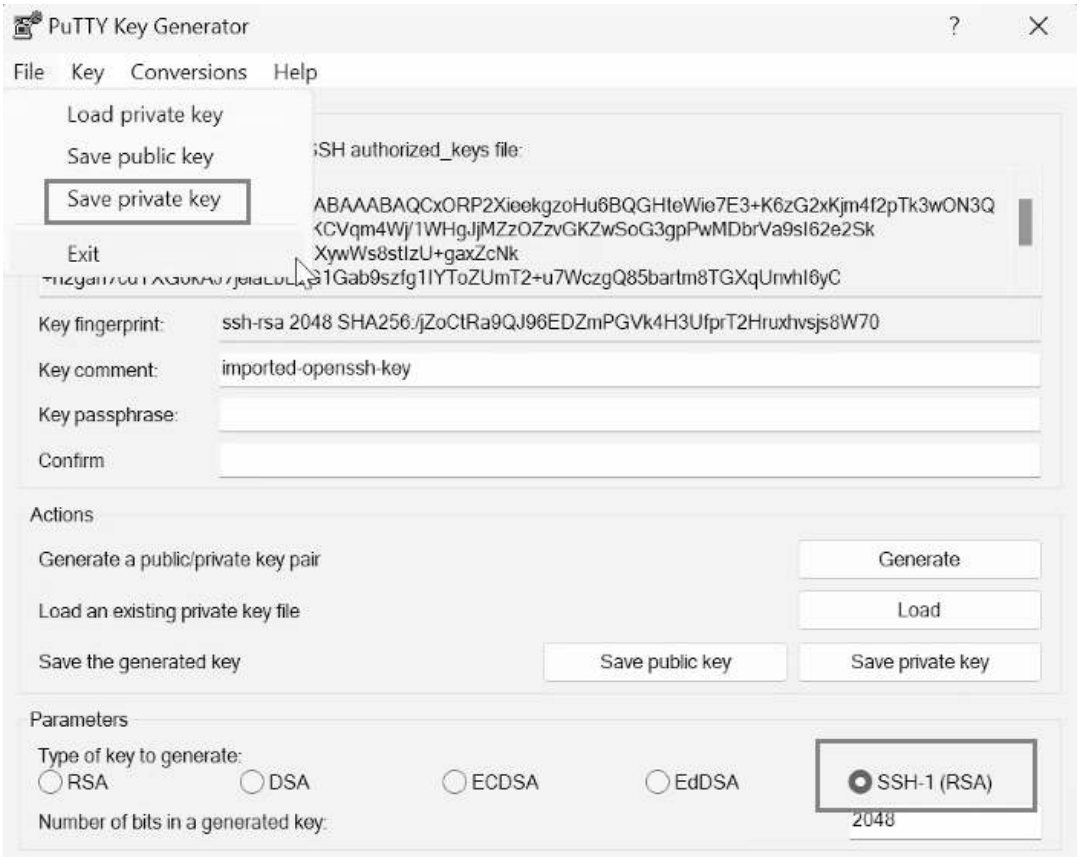
> **Note:**
>
> You cannot download another copy of the private key generated during this operation after completing the operation.

- **Upload SSH key files:** Select this option to browse or drag and drop `.pub` files.
- **Paste SSH keys:** Select this option to paste in individual public keys. To paste multiple keys, click + Another SSH key, and supply a single key for each entry.

6. Click **Save Changes.**
7. Now, we will import the private key that you downloaded and use PuTTY Gen for secure and passwordless authentication when connecting to the DB System using PuTTY. On

PuTTy Gen, click **Conversions** and then click **Import key.** This opens the Load Private Key window.

8. Click **Browse** and select the private key file that you have downloaded. Click **Open.**
9. On PuTTY Gen, select **SSH-1(RSA)** under **Parameters**.



10. Click **File** and then click **Save Private key.** Once again, go to File and click **Exit** to close.
11. Now, open PuTTY and under **Category**, click **Sessions** to create a new session to connect to the DB System. Enter the following:
    - **Host Name (or IP Address):** Type in the host name of the DB system.

      > **Note:**
      >
      > The host name and IP address of the DB System is available on the DB System information page.

    - **Connection Type:** Select **SSH.**
    - **Saved Sessions:** If you already save a session saved, select it and click **Load.**
12. Click **Save.**
13. Under Category, go to **Connections** and expand **SSH** and click **Auth** and then click **Credentials.**
14. In the **Private Key File for Authentication** field, click **Browse.** Browse and locate the private key file that you downloaded and saved.

15. Now, go to **Sessions** under Category and click **Open.**
16. This opens a PuTTy terminal.

> **Note:**
>
> In OCI environments, you are logged in an `opc` user. This is the default user with `sudo` privileges.

You must connect to the DB system as a `root` user. The `root` user has superuser privileges.

- To switch from `opc` user to `root` user, run

  ```
  sudo su -
  ```

- To switch from `root` user to `opc` user, run

  ```
  sudo su - opc
  ```

- To switch from `opc` user to `oracle` user, run

  ```
  sudo su - oracle
  ```

- To check your current user, run

  ```
  whoami
  ```

- To verify sudo permissions, use run

  ```
  sudo -l
  ```

This completes the task of connecting to a DB system as a root user.

Administrative Tasks for Oracle Machine Learning on Base Database

Lists the administrative tasks related to Oracle Machine Learning on Oracle Base Database. These tasks require the `ADMIN` role.

Table Administrative Tasks for OML on Base Database

Tasks	Interfaces/Tools	Links
Create a Database System	Oracle Cloud Infrastructure (OCI) Console	• Overview of Creating a DB System • Create a DB System Using the Console • Create a DB System from a Backup Using the Console

Table (Cont.) Administrative Tasks for OML on Base Database

Tasks	Interfaces/Tools	Links
Install OML4Py	• DB system on OCI Console • `$HOME/python` directory on the on premises Oracle Database. The On premises Oracle Database includes the Python 3.12.1.	1. Connecting to a DB System 2. Build and Install Python for Linux for On-Premises Databases
Install OML4R	• DB system on OCI Console • `/etc/yum.repos.d` directory on the Linux server	1. Connecting to a DB System 2. Install Oracle R Distribution on Oracle Linux 8 using Yum or Dnf

Part III

Administering OML for On Premises Oracle Database

Oracle Machine Learning related administrative tasks for on premises Oracle Database are categorized for OML4Python, OML4R, and OML4SQL.

Topics include:

- Administering Oracle Machine Learning for Python
- Administering Oracle Machine Learning for R
- Administering Oracle Machine Learning for SQL

Administering Oracle Machine Learning for Python

- Overview
  OML4Py is a Python module that enables Python users to manipulate data in database tables and views using Python syntax. OML4Py functions and methods transparently translate a select set of Python functions into SQL for in-database execution.
- Database and System Requirements for OML4Py
  Lists the Oracle Database and system requirements for Oracle Machine Learning for Python.
- Oracle Machine Learning for Python Architecture for Oracle Database
  Oracle Machine Learning for Python has a client server architecture based on Oracle Database and Oracle Client.
- Client and Server Components of Oracle Machine Learning for Python for On Premises Database
  The client and server components of OML4Py for on premises database is illustrated here.
- Administrative Tasks for Oracle Machine Learning for Python
  Lists the administrative tasks related to Oracle Machine Learning for Python. These tasks require the `ADMIN` role.

Overview

OML4Py is a Python module that enables Python users to manipulate data in database tables and views using Python syntax. OML4Py functions and methods transparently translate a select set of Python functions into SQL for in-database execution.

- What Is Oracle Machine Learning for Python
  Oracle Machine Learning for Python (OML4Py) enables you to run Python commands for data transformations and for statistical, machine learning, and graphical analysis on data stored in or accessible through an Oracle database using a Python API. The OML4Py supports running user-defined Python functions through the database spawned and

controlled Python engines, with optional built-in data-parallelism and task-parallelism. This embedded execution functionality enables invoking user-defined functions from SQL, and on ADB, REST. The OML4Py supports Automated Machine Learning (AutoML) for algorithm and feature selection, and model tuning and selection. You can augment the Python included functionality with third-party packages from the Python ecosystem.

- Advantages of Oracle Machine Learning for Python
  Using OML4Py to prepare and analyze data in or accessible to an Oracle database has many advantages for a Python user.
- Manipulate database tables and views using familiar Python functions and syntax
  With the transparency layer classes, you can manipulate database tables and views using familiar Python functions and syntax, For example, using DataFrame proxy objects that map to database data, users can invoke overloaded Pandas functions that transparently generate SQL that runs in the database, using the database as a high-performance compute engine.
- About the Python Components and Libraries in OML4Py
  OML4Py requires an installation of Python, the specified Python libraries, as well as the OML4Py components.

What Is Oracle Machine Learning for Python

Oracle Machine Learning for Python (OML4Py) enables you to run Python commands for data transformations and for statistical, machine learning, and graphical analysis on data stored in or accessible through an Oracle database using a Python API. The OML4Py supports running user-defined Python functions through the database spawned and controlled Python engines, with optional built-in data-parallelism and task-parallelism. This embedded execution functionality enables invoking user-defined functions from SQL, and on ADB, REST. The OML4Py supports Automated Machine Learning (AutoML) for algorithm and feature selection, and model tuning and selection. You can augment the Python included functionality with third-party packages from the Python ecosystem.

OML4Py is a Python module that enables Python users to manipulate data in database tables and views using Python syntax. OML4Py functions and methods transparently translate a select set of Python functions into SQL for in-database execution.

OML4Py is available in the following Oracle database environments:

- OML4Py is available in the Python interpreter in Oracle Machine Learning Notebooks in your Oracle Autonomous Database. For more information, see Use the Python Interpreter in a Notebook Paragraph in *Using Oracle Machine Learning Notebooks*.
- An OML4Py client connection to OML4Py in an on-premises Oracle Database instance.

  For this environment, you must install Python, the required Python libraries, and the OML4Py server components in the database, and you must install the OML4Py client. See Install OML4Py for On-Premises Databases.

Designed for problems involving both large and small volumes of data, OML4Py integrates Python with the database. With OML4Py, you can do the following:

- Run overloaded Python functions and use native Python syntax to manipulate in-database data, without having to learn SQL.
- Use Automated Machine Learning (AutoML) to enhance user productivity and machine learning results through automated algorithm and feature selection, as well as model tuning and selection.
- Use Embedded Python Execution to run user-defined Python functions in Python engines spawned and managed by the database environment. The user-defined functions and data

are automatically loaded to the engines as required, and when data-parallel and task-parallel execution is enabled. Develop, refine, and deploy user-defined Python functions and machine learning models that leverage the parallelism and scalability of the database to automate data preparation and machine learning.

- Use a natural Python interface to build in-database machine learning models.

Advantages of Oracle Machine Learning for Python

Using OML4Py to prepare and analyze data in or accessible to an Oracle database has many advantages for a Python user.

With OML4Py, you can do the following:

- **Operate on database data without using SQL**

  OML4Py transparently translates many standard Python functions into SQL. With OML4Py, you can create Python proxy objects that access, analyze, and manipulate data that resides in the database. OML4Py can automatically optimize the SQL by taking advantage of column indexes, query optimization, table partitioning, and database parallelism.

  OML4Py overloaded functions are available for many commonly used Python functions, including those on Pandas data frames for in-database execution.

  See Also: Manipulate database tables and views using familiar Python functions and syntax

- **Automate common machine learning tasks**

  By using Oracle's advanced Automated Machine Learning (AutoML) technology, both data scientists and beginner machine learning users can automate common machine learning modeling tasks such as algorithm selection and feature selection, and model tuning and selection, all of which leverage the parallel processing and scalability of the database.

  See Also: About Automated Machine Learning

- **Minimize data movement**

  By keeping data in the database whenever possible, you eliminate the time involved in transferring the data to your client Python engine and the need to store the data locally. You also eliminate the need to manage the locally stored data, which includes tasks such as distributing the data files to the appropriate locations, synchronizing the data with changes that are made in the production database, and so on.

  See Also: About Moving Data Between the Database and a Python Session

- **Keep data secure**

  By keeping the data in the database, you have the security, scalability, reliability, and backup features of the database for managing the data.

- **Use the power of the database**

  By operating directly on data in the database, you can use the memory and processing power of the database and avoid the memory constraints of your client Python engine.

- **Use current data**

  As data is refreshed in the database, you have immediate access to current data.

- **Save Python objects to a datastore in the database**

  You can save Python objects to an OML4Py datastore for future use and for use by others.

  See Also: About OML4Py Datastores

- **Build and store native Python models in the database**

  Using Embedded Python Execution, you can build native Python models and store and manage them in an OML4Py datastore.

  You can also build in-database models, with, for example, an `oml` class such as the Decision Tree class `oml.dt`. These in-database models have proxy objects that reference the actual models. Keeping with normal Python behavior, when the Python engine terminates, all in-memory objects, including models, are lost. To prevent an in-database model created using OML4Py from being deleted when the database connection is terminated, you must store its proxy object in a datastore.

  See Also: About Machine Learning Classes and Algorithms

- **Score data**

  For most of the OML4Py machine learning classes, you can use the `predict` and `predict_proba` methods of the model object to score new data.

  For these OML4Py in-database models, you can also use the SQL `PREDICTION` function on the model proxy objects, which scores directly in the database. You can use in-database models directly from SQL if you prepare the data properly. For open source models, you can use Embedded Python Execution and enable data-parallel execution for performance and scalability.

- **Run user-defined Python functions in embedded Python engines**

  Using OML4Py Embedded Python Execution, you can store user-defined Python functions in the OML4Py script repository, and run those functions in Python engines spawned by the database environment. When a user-defined Python function runs, the database starts, controls, and manages one or more Python engines that can run in parallel. With the Embedded Python Execution functionality, you can do the following:

  – Use a select set of Python packages in user-defined functions that run in embedded Python engines

  – Use other Python packages and third-party package in user-defined Python functions that run in embedded Python engines

  – Operationalize user-defined Python functions for use in production applications and eliminate porting Python code and models into SQL, and on ADB, REST; avoid reinventing code to integrate Python results into existing applications

  – Seamlessly leverage your Oracle database as a high-performance computing environment for user-defined Python functions, providing data parallelism and resource management

  – Perform parallel simulations, for example, Monte Carlo analysis, using the `oml.index_apply` function

  – Generate JSON images, PNG images and XML representations of both structured and image data, which can be used by Python clients and SQL-based applications. PNG images and structured data can be used for Python clients and applications that use REST APIs.

  See Also: About Embedded Python Execution

Manipulate database tables and views using familiar Python functions and syntax

With the transparency layer classes, you can manipulate database tables and views using familiar Python functions and syntax, For example, using DataFrame proxy objects that map to

database data, users can invoke overloaded Pandas functions that transparently generate SQL that runs in the database, using the database as a high-performance compute engine.

The OML4Py transparency layer does the following:

- Enables creating tables and views from `pandas.DataFrame` and getting proxy objects to tables and views.
- Overloads specific Python functions that transparently translate functionality to SQL
- Leverages proxy objects for database data
- Uses familiar Python syntax to manipulate database data

The following table lists the transparency layer functions for getting and creating proxy objects and tables/views.

Table Transparency Layer Functions for getting and creating proxy objects and tables/views

Function	Description
`oml.create`	Creates a table in a the database schema from a Python data set.
`oml_object``.pull`	Creates a local Python object that contains a copy of data fetched from database object referenced by the `oml` object.
`oml.push`	Pushes data from a Python session into an object in a database schema.
`oml.sync`	Creates a `DataFrame` proxy object in Python that represents a database table or view.
`oml.dir`	Return the names of `oml` objects in the Python session workspace.
`oml.drop`	Drops a persistent database table, view or in-database model.

Transparency layer proxy classes map SQL data types or objects to corresponding Python types. The classes provide Python functions and operators that are the same as those on the mapped Python types. The following table lists the transparency layer data type classes.

Table Transparency Layer Data Type Classes

Class	Description
`oml.Boolean`	A boolean series data class that represents a single column of 0, 1, and NULL values in database data.
`oml.Bytes`	A binary series data class that represents a single column of `RAW` or `BLOB` database data types.
`oml.Float`	A numeric series data class that represents a single column of `NUMBER`, `BINARY_DOUBLE`, or `BINARY_FLOAT` database data types.
`oml.String`	A character series data class that represents a single column of `VARCHAR2`, `CHAR`, or `CLOB` database data types.
`oml.DataFrame`	A tabular `DataFrame` class that represents multiple columns of `oml.Boolean`, `oml.Bytes`, `oml.Float`, `oml.String`, and `oml.Vector` data.
`oml.Integer`	A data class that represents a single column of `NUMBER(*,0)` data in the database.

Table (Cont.) Transparency Layer Data Type Classes

Class	Description
`oml.Datetime`	A series date class that represents a single column of `TIMESTAMP` or `TIMESTAMP WITH TIME ZONE` in Oracle Database. `oml.Timezone` A time class that is used with `oml.Datetime` to support `TIME STAMP WITH TIME ZONE`. `oml.Timedelta` A time class that represents a single column series of differences between two dates or times, or `INTERVAL DAY TO SECOND` in Oracle Database.
`oml.Timezone`	A time class that is used with `oml.Datetime` to support `TIME STAMP WITH TIME ZONE`.
`oml.Timedelta`	A time class that represents a single column series of differences between two dates or times, or `INTERVAL DAY TO SECOND` in Oracle Database.
`oml.Vector`	A vector series data class that represents a single column of `VECTOR` data in Oracle Database.

The following table lists the mappings of Python data types for both the reading and writing of data between Python and the database.

Table Python and SQL Data Type Equivalencies

Database Read	Python Data Types	Database Write
N/A	`Bool`	If `oranumber == True`, then `NUMBER` (the default), else `BINARY_DOUBLE`.
`BLOB`	`bytes`	`BLOB`
`RAW`		`RAW`
`BINARY_DOUBLE`	`float`	If `oranumber == True`, then `NUMBER` (the default), else `BINARY_DOUBLE`.
`BINARY_FLOAT`		
`NUMBER`		
`CHAR`	`str`	`CHAR`
`CLOB`		`CLOB`
`VARCHAR2`		`VARCHAR2`
`NUMBER(*,0)`	`int`	`NUMBER(*,0)`
`TIMESTAMP` or `TIMESTAMP WITH TIME ZONE`	`datetime.datetime`	`TIMESTAMP` or `TIMESTAMP WITH TIME ZONE`
`TIMESTAMP WITH TIME ZONE`	`datetime.timezone`	`TIMESTAMP WITH TIME ZONE`
`INTERVAL DAY TO SECOND`	`datetime.timedelta`	`INTERVAL DAY TO SECOND`

About the Python Components and Libraries in OML4Py

OML4Py requires an installation of Python, the specified Python libraries, as well as the OML4Py components.

- In Oracle Autonomous Database, OML4Py is already installed. The OML4Py installation includes Python, additional required Python libraries, and the OML4Py server components. A Python interpreter is included with Oracle Machine Learning Notebooks in Autonomous Database.

- You can install third-party Python libraries in a conda environment through a conda interpreter for use within OML Notebooks sessions and OML4Py embedded execution invocations.
- You can install OML4Py in an on-premises Oracle Database. In this case, you must install Python, the additional required Python libraries, the OML4Py server components, and an OML4Py client. See Install OML4Py for On-Premises Databases.

Python Version in Current Release of OML4Py

To determine the current version of python used by OML4Py, run the following command:

```
import sys
sys.version
```

This version is in the current release of Oracle Autonomous Database.

Required Python Libraries

The following Python libraries must be included.

- `oracledb 2.2.0`
- `cycler 0.10.0`
- `joblib 1.1.0`
- `kiwisolver 1.1.0`
- `matplotlib 3.8.4`
- `numpy 1.26.4`
- `pandas 2.1.1`
- `Pillow-8.2.0`
- `pyparsing 2.4.0`
- `python-dateutil 2.8.1`
- `pytz 2022.1`
- `scikit-learn 1.4.1.post1`
- `scipy 1.12.0`
- `six 1.13.0`
- `threadpoolctl 3.1.0`

All the above libraries are included with Python in the current release of Oracle Autonomous Database.

For an installation of OML4Py in an on-premises Oracle Database, you must install Python and additionally the libraries listed here. See Install OML4Py for On-Premises Databases.

Database and System Requirements for OML4Py

Lists the Oracle Database and system requirements for Oracle Machine Learning for Python.

Table Database and Systems Requirements for OML4Py

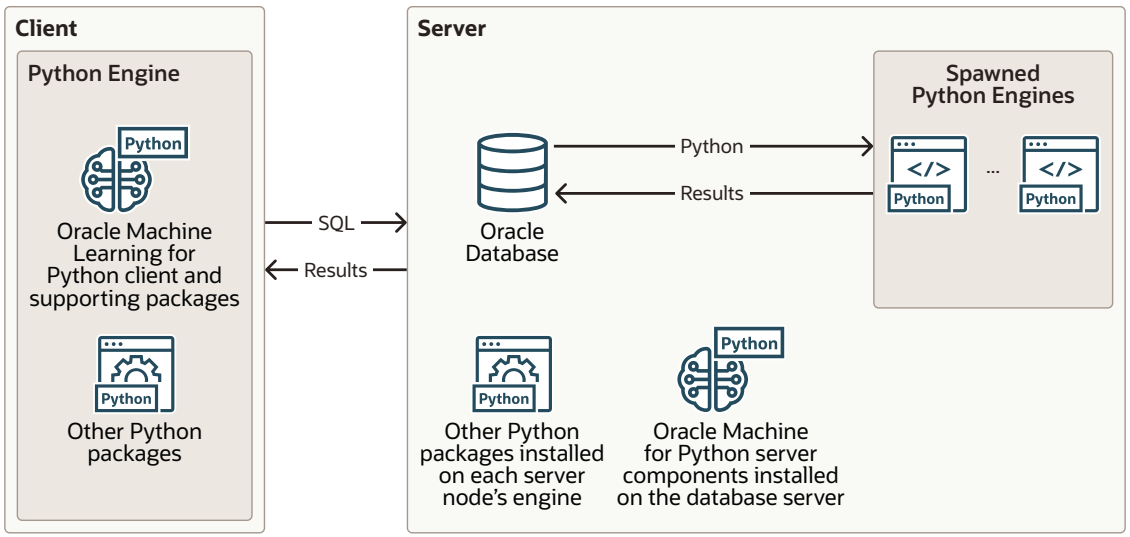
OML4Py Version	Oracle Autonomous Database Serverless			Oracle Database		
	Python Version	Database Version	Operating System	Python Version	Database Version	Operating System
2.1	3.12.6	23ai	Linux X64 Container (OL7)	3.12.6	23ai	Linux X64 (OL8)
2.0.1	3.12.3	23ai	Linux X64 Container (OL7)	3.12.3	23ai	Linux X64 (OL8)
2.0	3.12.0	23ai	Linux X64 Container (OL7)	3.12.0	23ai	Linux X64 (OL8)
	3.12.0	21c, 19c	Linux X64 Container (OL7)	3.12.0	21c, 19c	Linux X64 (OL7, OL8)
1.0	NA	NA	NA	3.9.5	21c, 19c	Linux X64 (OL7, OL8)

Oracle Machine Learning for Python Architecture for Oracle Database

Oracle Machine Learning for Python has a client server architecture based on Oracle Database and Oracle Client.

This diagram depicts the client server architecture of Oracle Machine Learning for Python.

Figure OML4Py Client Server Architecture



The client and the server communication can take place through an OML4Py SQL API. For example, SQL Developer. It can also be a configured OML4Py Python client.

The OML4Py engines spawned from the database are Python contained inside an external procedure managed by the Oracle Database. The external procedure is a C program called by the database. It acts as a bridge to trigger the Python engine and return the results back to the database.

Client and Server Components of Oracle Machine Learning for Python for On Premises Database

The client and server components of OML4Py for on premises database is illustrated here.

- **OML4Py Client Components:**
  - Oracle Instant Client
  - OML4Py Client
- **OML4Py Server Components:**
  - Oracle Database with schema objects and shared libraries for supporting OML4Py clients
  - OML4Py packages and supporting packages

Administrative Tasks for Oracle Machine Learning for Python

Lists the administrative tasks related to Oracle Machine Learning for Python. These tasks require the `ADMIN` role.

Table Administrative Tasks for OML4Py

Tasks	Interfaces/Tools	Link
Create	• On premises Oracle Database. • The subdirectory under the directory containing the extracted OML4Py server installation files. **Note:** You must have the `PYQADMIN` database role to perform these tasks.	• Create New Users for on pemises Database • Grant Users the Required Privileges for on pemises Oracle Database

Table (Cont.) Administrative Tasks for OML4Py

Tasks	Interfaces/Tools	Link
Install	• On premises Oracle Database • OML4Py Server • OML4Py Client **Note:** For Pluggable Database, you must run the `pyqcfg.sql` script at `CDB$ROOT` level. Else, you must run have the `sysdba` role to run the script. Here, is the dire	• Install OML4Py Server for Linux for on pemises Oracle Database 23ai • Build and Install Python for Linux for on pemises Databases • Install OML4Py Server for on pemises Oracle Database 19c/21c • Install OML4Py Client for on pemises Database

Table (Cont.) Administrative Tasks for OML4Py

Tasks	Interfaces/Tools	Link
	ctory where the script is located: `$ORACLE_HOME/oml4py/server/pyqcfg.sql`	
Post Installation	• On premises Oracle Database • OML4Py Server	• Verify OML4Py Server Installation for on pemises Database
Install Third-party packages	• OML Notebooks • The Object Storage bucket corresponding to the tenancy and database.	• Administrative task to create and save the conda environments • Conda commands • Create a Conda Environment for Python and Install Python Packages • Upload the Conda environments to an Object Storage bucket associated with the Autonomous Database
Manage	• Network Access Control Lists (ACL) API • Python system and process utilities library `psutil`	• Access and Authorization Procedures and Functions (Autonomous Database) • Manage System Processes in OML4Py Using psutil
Upgrade	• On premises Oracle Database • OML4Py Server • OML4Py Client On premises Oracle Database	• Upgrade the OML4Py Client for on pemises Databases • Upgrade OML4Py from 1.0 to 2.0 in DB 19c standalone

Table (Cont.) Administrative Tasks for OML4Py

Tasks	Interfaces/Tools	Link
Uninstall	• On premises Oracle Database • OML4Py Server • OML4Py Client	• Uninstall the OML4Py Server for on premises Database • Uninstall the OML4Py Client for on pemises Databases

Administering Oracle Machine Learning for R

- Overview
  Oracle Machine Learning for R is a set of R packages and database features that enable an R user to operate on database-resident data without using SQL and to run R scripts in one or more embedded R engines that run in the database environment.
- Oracle Machine Learning for R Architecture for Oracle Database
  Oracle Machine Learning for R has a client/server architecture based on Oracle Database and Oracle Client.
- page 15
- Client and Server Components of Oracle Machine Learning for R for On Premises Database
  Lists the client and server components of OML4R for on premises Oracle database.
- Database and System Requirements for Oracle Machine Learning for R
  Lists the Oracle Database and system requirements for Oracle Machine Learning for R.
- Administrative Tasks for Oracle Machine Learning for R
  Lists the administrative tasks related to Oracle Machine Learning for R. These tasks require the `ADMIN` role.

Overview

Oracle Machine Learning for R is a set of R packages and database features that enable an R user to operate on database-resident data without using SQL and to run R scripts in one or more embedded R engines that run in the database environment.

- What Is Oracle Machine Learning for R?
  Oracle Machine Learning for R (Oracle Machine Learning for R) is a comprehensive, database-centric environment for end-to-end analytical processes in R, with immediate deployment to production environments.

What Is Oracle Machine Learning for R?

Oracle Machine Learning for R (Oracle Machine Learning for R) is a comprehensive, database-centric environment for end-to-end analytical processes in R, with immediate deployment to production environments.

Oracle Machine Learning for R is a set of R packages and database features that enable an R user to operate on database-resident data without using SQL and to run R scripts in one or more embedded R engines that run in the database environment.

Using Oracle Machine Learning for R from your R session, you have easy access to data in a database instance. You can create and use R objects that correspond to database tables and views - referred to as "proxy objects" that enable in-database data exploration and preparation. Oracle Machine Learning for R has overloaded functions that translate R operations into SQL that runs in the database that benefits from the query. The database consolidates the SQL and can use in the database that benefits from the query optimization, parallel processing, and scalability features of the database. The database returns the results as R objects, which themselves can be proxy objects.

Oracle Machine Learning for R 2.0 is available in the R interpreter in Oracle Machine Learning Notebooks in your Oracle Autonomous Database instance. For more information, see Get Started with Notebooks for Data Analysis and Data Visualization. In this environment, all the required components are included, including R, required R libraries, and the R interpreter in OML Notebooks.

Embedded R execution provides some of the most significant advantages of using Oracle Machine Learning for R. Using Embedded R execution, you can store and run R scripts in the database through either an R interface or a SQL interface or both. The Embedded R execution can be run on Autonomous Database using both SQL and REST APIs. You can use the results of R scripts in SQL-enabled tools for structured data, R objects, and images.

Oracle Machine Learning for R Architecture for Oracle Database

Oracle Machine Learning for R has a client/server architecture based on Oracle Database and Oracle Client.

R engines run on the server computer and on each client computer. Oracle Machine Learning for R supports three key capabilities:

- **R Transparency**

  Oracle Machine Learning for R packages on the client support R transparency, which enables Oracle tables to appear "transparently" as native R objects. Oracle Machine Learning for R packages provide transparent access to Oracle Database tables and views, enabling users to invoke standard R functions, which are translated into SQL transparently to the user for in-database execution.

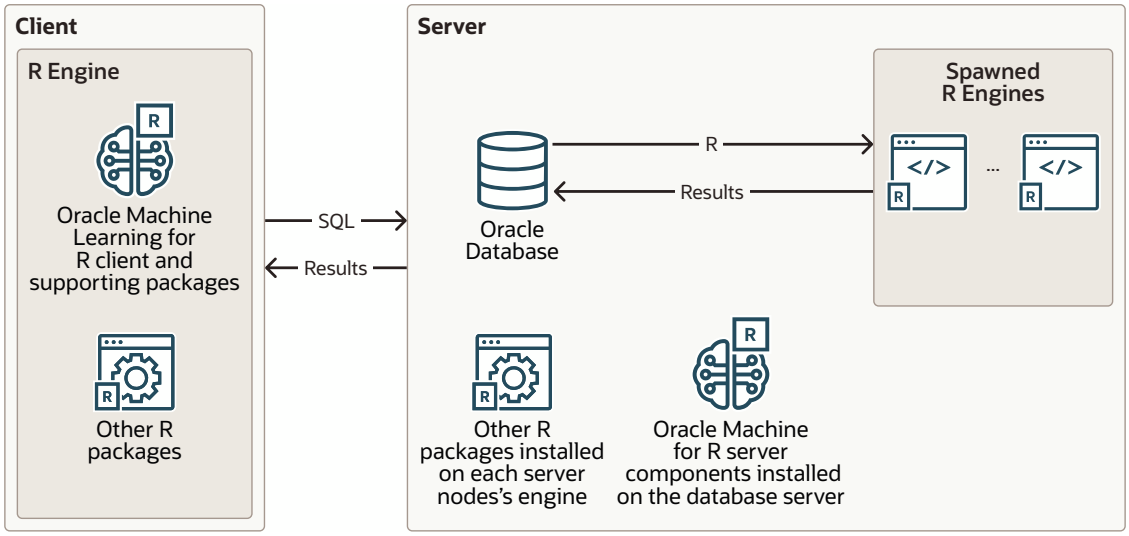
- **Predictive Analytics and Machine Learning**

  Oracle Machine Learning for R supports a wide range of parallel and distributed algorithms supporting predictive analytics and machine learning. This enables both scalability and improved performance, while leveraging a convenient R interface to in-database and database server-side algorithms.

- **Embedded R Execution**

  Oracle Machine Learning for R packages, libraries, and R and SQL APIs on the server support the execution of user-defined R functions within SQL queries and PL/SQL statements. Embedded R execution spawns R engines that can run in parallel, for data-parallel and task-parallel execution. With embedded R execution, you can run user-defined R functions, possibly leveraging third-party packages. With facilities like the `DBMS_SCHEDULER` database package, you can schedule the execution of user-defined R functions for lights-out processing.

Figure OML4R Client Server Architecture



The client and the server communication can take place through an OML4R SQL API. For example, SQL Developer. It can also be a configured OML4R R client.

The OML4R engines spawned from the database are R contained inside an external procedure managed by the Oracle Database. The external procedure is a C program called by the database. It acts as a bridge to trigger the R engine and return the results back to the database.

The image shows the client server architecture of OML4R for an on premises database. The client is depicted by the box on the left, and the server is depicted by the box on the right. The client and the server communicates using an OML4R SQL API. For example, SQL Developer. It can also be a configured OML4R R client.

Inside the client is shown the R engine. The R engine comprises:

- OML4R client and supporting packages
- Other R packages

The server is depicted by the box on the right. Inside the server are the following components:

- Oracle Database
- Spawned R engines
- OML4R server components installed on the Database server
- Other R packages installed on each server node's engines

Client and Server Components of Oracle Machine Learning for R for On Premises Database

Lists the client and server components of OML4R for on premises Oracle database.

- **OML4R Client Components**:

  - Oracle Database Client
  - OML4R packages and supporting packages
- **OML4R Server Components:**
  - Oracle Database with schema objects and shared libraries for supporting OML4R clients
  - OML4R packages and supporting packages

Database and System Requirements for Oracle Machine Learning for R

Lists the Oracle Database and system requirements for Oracle Machine Learning for R.

> **Note:**
>
> Oracle Database offers a range of editions, including XE, SE2, EE, EE-CS, DBCS-SE, DBCS-EE, DBCS EE-HP, DBCS EE-EP, ExaCS/CE, and Oracle Autonomous Database - Shared / Serverless.

Table Database and System Requirements for OML4R

OML4R Versions	Open Source R or Oracle R Distribution	Supported versions in Oracle Database Releases	Oracle Database Operating System
2.0	4.0.5	19c, 21c, 23ai	Linux x64 (OL7 and OL8)
1.5.1	3.6.1	18c, 19c, 21c	• Linux x64 (OL6, OL7, OL8) • IBM AIX 5.3, 6 and 7 64-bit • Solaris SPARC 64-bit • Microsoft Windows x64 (64-Bit)
1.5.1	3.3.0	11.2.0.4, 12.1.0.1, 12.1.0.2, 12.2.0.1	• Linux x64 (OL6, OL7, OL8) • Microsoft Windows x64 (64-Bit) • IBM AIX 5.3, 6 and 7 64-bit • Solaris x86 64-bit • Solaris SPARC 64-bit

Table (Cont.) Database and System Requirements for OML4R

OML4R Versions	Open Source R or Oracle R Distribution	Supported versions in Oracle Database Releases	Oracle Database Operating System
1.4.1	3.0.1, 3.1.1	11.2.0.3, 11.2.0.4, 12.1.0.1, 12.1.0.2	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)
1.4	3.0.1, 3.1.1	11.2.0.3, 11.2.0.4, 12.1.0.1	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)
1.3.1	2.15.1, 2.15.2, 2.15.3	11.2.0.3, 11.2.0.4, 12.1.0.1	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)
1.3	2.15.1	11.2.0.3, 11.2.0.4, 12.1.0.1	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)
1.2	2.15.1	11.2.0.3, 11.2.0.4, 12.1.0.1	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)

Table (Cont.) Database and System Requirements for OML4R

OML4R Versions	Open Source R or Oracle R Distribution	Supported versions in Oracle Database Releases	Oracle Database Operating System
1.1	2.13.2	11.2.0.3, 11.2.0.4, 12.1.0.1	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)
1.0	2.13.2	11.2.0.3, 11.2.0.4, 12.1.0.1	• Linux x64 (OL5 and OL6) • Solaris x86 64-bit • Solaris SPARC 64-bit • IBM AIX 5.3 and 6 64-bit • Microsoft Windows x64 (64-Bit)

Administrative Tasks for Oracle Machine Learning for R

Lists the administrative tasks related to Oracle Machine Learning for R. These tasks require the `ADMIN` role.

Table Administrative Tasks for OML4R

Tasks	Interfaces/Tools	Links
Create	• On premises Oracle Database • `$ORACLE_HOME/R/server` on the Oracle On premises Database • Oracle Wallet Manager **Note:** You must have the `RQADMIN` role to perform these tasks.	• Create a Database User for Oracle Machine Learning for R – About the RQADMIN Role • Create an Oracle Wallet for an Oracle Machine Learning for R Connection
Install	• On premises Oracle Database • OML4R Server • OML4R Client	• Install Oracle Machine Learning for R Server for Oracle Database • Install Oracle R Distribution on Oracle Linux 8 using Yum or Dnf • Install Oracle R Distribution on Oracle Linux 7 Using Yum • Install Oracle R Distribution on Oracle Linux Using RPMs • Install Oracle R Distribution on Red Hat Enterprise Linux • Install Oracle Instant Client from a Zip File • Install Oracle Instant Client on Linux from RPMs • Install the Supporting Packages on Linux – R Package Installation Tips

Table (Cont.) Administrative Tasks for OML4R

Tasks	Interfaces/Tools	Links
Post-installation	• On premises Oracle Database • OML4R Server • OML4R Client	• Verify the Oracle Machine Learning for R Server Installation • Connect Oracle Machine Learning for R Client to Oracle Machine Learning for R Server
Install third-party packages	• OML Notebooks • The Object Storage bucket corresponding to the tenancy and database.	• Administrative task to create and save the conda environments
Configure	• On premises Oracle Database • OML4R Server • OML4R Client	• Enable MKL Support for Oracle R Distribution on a Linux Client • Enable MKL Support for Oracle R Distribution on a Windows Client • Modifying the Number of Threads for MKL on Linux • Modify the Number of Threads for MKL on Windows
Migrate	• On premises Oracle Database • OML4R Server	• Migrate Oracle Machine Learning for R Data • Migrate Oracle Machine Learning for R After Database Upgrade
Uninstall	• On premises Oracle Database • OML4R Server • OML4R Client	• Uninstall OML4R Server from Oracle Database 23ai • Uninstall OML4R Server from Oracle Database 18c or Later • Uninstall OML4R Server from Oracle Database 12c and Earlier • Uninstall OML4R Client • Uninstall Oracle R Distribution on Linux
Upgrade	• On premises Oracle Database • OML4R Server • OML4R Client	• Upgrade Oracle Machine Learning for R • Upgrade Oracle Machine Learning for R from 1.5.1 release to 2.0
Downgrade	• On premises Oracle Database • OML4R Server	• Downgrade Oracle Machine Learning for R

Administering Oracle Machine Learning for SQL

Overview

Oracle Machine Learning for SQL (OML4SQL) provides scalable in-database machine learning algorithms through PL/SQL and SQL APIs. The algorithms are fast and scalable, support algorithm-specific automatic data preparation, and can score in batch or real-time.

About Oracle Machine Learning for SQL

Oracle Machine Learning for SQL (OML4SQL) provides scalable in-database machine learning algorithms through PL/SQL and SQL APIs. The algorithms are fast and scalable, support algorithm-specific automatic data preparation, and can score in batch or real-time.

OML4SQL provides a powerful, state-of-the-art machine learning capability within Oracle Database. The parallelized algorithms in the database keep data under database control. There is no need to extract data to separate machine learning engines, which adds latency to data access and raises concerns about data security, storage, and recency. The algorithms are fast and scalable, support algorithm-specific automatic data preparation, and can score in batch or real-time. You can use OML4SQL to build and deploy predictive and descriptive machine learning applications, to add intelligent capabilities to existing applications, and to generate predictive queries for data exploration. OML4SQL provides explanatory prediction details when scoring data, so you can understand why an individual prediction is made.

OML4SQL offers a broad set of in-database algorithms for performing a variety of machine learning tasks, such as classification, regression, anomaly detection, feature extraction, clustering, and market basket analysis. The algorithms can work on standard case data, transactional data, star schemas, and unstructured text data. OML4SQL is uniquely suited to the analysis of very large data sets.

Oracle Machine Learning for SQL, along with Oracle Machine Learning for R and Oracle Machine Learning for Python, is a component of Oracle Machine Learning that provides three powerful APIs for in-database machine learning, among other features.

Database and System Requirements for OML4SQL

Lists the Oracle Database and system requirements for Oracle Machine Learning for SQL.

> **Note:**
>
> Oracle Database offers a range of editions, including XE, SE2, EE, EE—CS, DBCS—SE, DBCS—EE, DBCS EE—HP, DBCS EE—EP, ExaCS/CE, and Oracle Autonomous Database—Serverless.

Table Database and System Requirements for OML4SQL

Database Offerings	Database Versions	System Requirements for 23ai	System Requirements for 21c and 19c	System Requirements for 18c
Free	18c, 19c, 21c, 23ai	• Windows x64 • Linux x64 (OL9) • Linux x64 (OL8) • Red Hat Enterprise Linux 8 • Red Hat Enterprise Linux 9 For downloads, see software downloads page.	• Windows x64 • Linux x64 (OL9) • Linux x64 (OL8) • Linux x64 (OL7) For more information, see 21c requirements and 19c requirements.	• Windows x64 • Linux x64 (OL7) • Linux x64 (OL6) See 18c requirements.
SE2	18c, 19c, 21c, 23ai	• Windows x64 • Linux x64 (OL9) • Linux x64 (OL8) For downloads, see software downloads page.	• Windows x64 • Linux x64 (OL9) • Linux x64 (OL8) • Linux x64 (OL7) See 21c requirements and 19c requirements.	• Windows x64 • Linux x64 (OL7) • Linux x64 (OL6) See 18c requirements.
EE	18c, 19c, 21c, 23ai	NA	NA	NA
EE—CS	18c, 19c, 21c, 23ai	NA	NA	NA
BaseDB—SE	18c, 19c, 21c, 23ai	NA	NA	NA
BaseDB—EE	18c, 19c, 21c, 23ai	NA	NA	NA
BaseDB EE—HP	18c, 19c, 21c, 23ai	NA	NA	NA
BaseDB EE—EP	18c, 19c, 21c, 23ai	NA	NA	NA
ExaCS/CE	18c, 19c, 21c, 23ai	NA	NA	NA
Oracle Autonomous Database—Serverless	18c, 19c, 21c, 23ai	NA	NA	NA

Administrative Tasks for Oracle Machine Learning for SQL

Lists the administrative tasks related to Oracle Machine Learning for SQL. These tasks require the `ADMIN` role.

Table Administrative Tasks for OML4SQL

Tasks	Interfaces/Tools	Links
Install and Configure a Database for Oracle Machine Learning for SQL	• Oracle SQL Developer • Oracle Data Miner	• Install and Configure a Database for Oracle Machine Learning for SQL • Database Tuning Considerations for Oracle Machine Learning for SQL • System Privileges for Oracle Machine Learning for SQL • Object Privileges for Oracle Machine Learning for SQL Models
Create OML user	• SQL*Plus **Note:** You must have the `SYSDBA` role to create a user.	• Create an Oracle Machine Learning for SQL User
Grant and Manage Privileges	• Oracle SQL Developer	• Grant Privileges for Oracle Machine Learning for SQL

Table (Cont.) Administrative Tasks for OML4SQL

Tasks	Interfaces/Tools	Links
Export and Import Oracle Machine Learning for SQL Models	• Export and Import functions of Oracle Database • `DBMS_DATA_MINING` package • Command-line clients of Oracle Data Pump	• Export and Import Oracle Machine Learning for SQL Models • Options for Exporting and Importing Oracle Machine Learning for SQL Models • Use EXPORT_MODEL and IMPORT_MODEL • EXPORT and IMPORT Serialized Models • Directory Objects for EXPORT_MODEL and IMPORT_MODEL • Import From PMML
Audit and Add Comments to Oracle Machine Learning for SQL Models	• SQL `COMMENT` statement • Oracle Database auditing system	• Add a Comment to a Oracle Machine Learning for SQL Model • Audit and Add Comments to Machine Learning for SQL Models
Upgrade OML4SQL	• Database Upgrade Assistant	• Pre-Upgrade Steps • Use Database Upgrade Assistant to Upgrade Oracle Machine Learning for SQL • Use Export/Import to Upgrade Machine Learning for SQL Models • Post Upgrade Steps
Downgrade OML4SQL	• Database Upgrade Assistant	• Downgrade Oracle Machine Learning for SQL

Glossary

ACL List

A Network Access Control List is a set of rules or permissions that define what network traffic is allowed to pass through a network device, such as a router, firewall, or gateway. ACLs are used to control and filter incoming and outgoing traffic based on various criteria such as IP addresses, port numbers, and protocols. They play a crucial role in network security by enabling administrators to manage and restrict network traffic to prevent unauthorized access, potential attacks, and data breaches.

ADMIN

A role that provides the highest level of access. You must have this role to create and manage database users, grant privileges, perform other administrative tasks, and manage the overall security and configuration of the database. This role grants full control over the database.

Bucket

Buckets are logical containers for storing objects. Users or systems create buckets as needed within a region. A bucket is associated with a single compartment that has policies that determine what actions a user can perform on a bucket and on all the objects in the bucket.

Client Server Architecture

In the Oracle Database system environment, the database application and the database are separated into two parts: a front-end or client portion, and a back-end or server portion. Therefore, the term client server architecture. The client runs the database application that accesses database information and interacts with a user. The server runs the Oracle software and handles the functions required for concurrent, shared data access to an Oracle database.

Conda

Conda is a cross-platform package and environment management system. It is used to install python and R packages in OML notebooks.

DBA

A predefined `DBA` role is automatically created with every Oracle Database installation. This role contains most database system privileges. Therefore, you should grant the `DBA` role only to

actual database administrators. The `DBA` role does not include the `SYSDBA` or `SYSOPER` system privileges.

Module

A module is a python library or extension that typically gets installed through an OML notebook using the conda utility.

Object

Any type of data, regardless of content type, is stored as an object. An object is composed of the object itself and metadata about the object. Each object is stored in a bucket.

Object Storage

Object Storage is part of OCI (Oracle Cloud Infrastructure). It is an internet-scale, high-performance storage platform that offers reliable and cost-efficient data durability. The Object Storage service can store a large amount of unstructured data of any content type, including analytic data and rich content, like images and videos.
Object Storage is a regional service and is not tied to any specific compute instance. You can access data from anywhere inside or outside the context of the Oracle Cloud Infrastructure, as long you have internet connectivity and can access one of the Object Storage endpoints.

PQADMIN

A database role that allows you to store and manage user-defined Python functions in the OML4Py script repository.

PSUTIL

A cross-platform library for retrieving information on running processes and system utilization, such as CPU, memory, disks, network, and sensors, in Python. It is useful for system monitoring, profiling, limiting process resources, and the management of running processes.

Root user

The root user has unrestricted access to all files and directories on the system, allowing it to perform any action, including system administration tasks.

RQADMIN

A database role that allows you to create and manage user-defined R functions for embedded R execution.

SYS

When you create an Oracle database, the `SYS` user is automatically created and granted the `DBA` role.

- The SYS user is the most powerful user and should only be used for critical tasks, typically with the SYSDBA privilege.
- The SYS user owns the data dictionary and contains the core internal tables that the database needs to function.

All base tables and views for the database data dictionary are stored in the schema SYS. These base tables and views are critical for the operation of Oracle Database. To maintain the integrity of the data dictionary, tables in the SYS schema are manipulated only by the database. They should never be modified by any user or database administrator. Also, you should not create any tables in the schema of user SYS, although you can change the storage parameters of the data dictionary settings if necessary.

SYSDBA

A system privilege assigned to the SYS user. It allows high-level administrative tasks like starting and shutting down the database.

SYSTEM

When you create an Oracle Database, the user SYSTEM is also automatically created and granted the DBA role.

- The SYSTEM user can create additional tables and views that display administrative information as well as internal tables and views used by various Oracle Database options and tools.
- The SYSTEM user also has DBA privileges but is intended for administrative tasks like creating user-defined tables and views.
- Unlike SYS, the SYSTEM role is considered safe for routine DBA work.

Never use the SYSTEM schema to store tables of interest to nonadministrative users. A predefined DBA role is automatically created with every Oracle Database installation. This role contains most database system privileges. Therefore, you should grant the DBA role only to actual database administrators. The DBA role does not include the SYSDBA or SYSOPER system privileges.

> **Note:**
>
> SYSDBA and SYSOPER are administrative privileges required to perform basic database operations such as creating the database and instance startup and shutdown. Depending upon the level of authorization you require, you must have one of these privileges granted to you.

Wallet

A wallet is a secure password-protected container that provides a simple and easy way to manage database credentials across multiple domains. It allows you to update database

credentials by updating the Wallet instead of having to change individual datasource definitions.

Wallet Manager

Oracle Wallet Manager is an application that wallet owners use to manage and edit the security credentials in their Oracle wallets. A wallet is a password-protected container used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL.