Oracle® Cloud

Administering Oracle Database Cloud Service

E48368-59
February 2019

Oracle Cloud Administering Oracle Database Cloud Service,

E48368-59

Contents

5 Accessing Database Cloud Service

6 Backing Up and Restoring Databases on Database Cloud Service

7 Patching Database Cloud Service

8 Configuring Database Features, Database Options, and Companion Products

9 Migrating Oracle Databases to Database Cloud Service

10 Frequently Asked Questions for Database Cloud Service

11 Troubleshooting Database Cloud Service

A Characteristics of a Newly Created Deployment

B Oracle Cloud Pages for Administering Database Cloud Service

C The oracle-dbcs-cli Utility

D The dbaascli Utility

E The raccli Utility

F The dbpatchmdg Utility

G Using Oracle DBaaS Monitor

Preface

This document describes how to manage and monitor Oracle Database Cloud Service and provides references to related documentation.

Topics

Audience

This document is intended for Oracle Cloud users who want to manage and monitor Oracle Database Cloud Service.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see these Oracle resources:

- *Getting Started with Oracle Cloud*
- *Using Oracle Cloud Infrastructure Object Storage Classic*
- *Using Oracle Cloud Infrastructure Compute Classic*

Conventions

The following text conventions are used in this document:

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

1

Getting Started with Database Cloud Service

This section describes how to get started with Oracle Database Cloud Service for administrators and application owners.

Topics

About Oracle Database Cloud Service

Oracle Database Cloud Service provides you the ability to deploy Oracle databases in the Cloud, with each database deployment containing a single Oracle database or an Oracle Data Guard configuration consisting of a primary Oracle database and a standby Oracle database. You have full access to the features and operations available with Oracle Database, but with Oracle providing the computing power, physical storage and (optionally) tooling to simplify routine database maintenance and management operations.

When you create a database deployment, Database Cloud Service creates compute nodes to host the database, using computing, storage and networking resources provided by various Oracle Cloud infrastructure services.

The deployment includes Oracle Database and supporting software. The software is installed for you, an Oracle database is created using values you provide when creating the database deployment, and the database is started. Additionally, you can direct Database Cloud Service to set up automatic backups. Finally, the deployment includes cloud tooling that simplifies backup, recovery, patching and upgrade operations. You have root privilege, so you can load and run software in the compute environment. You have full administrative privileges for the Oracle database. You are responsible for making any changes to the automated maintenance setup, and you are responsible for recovery operations in the event of a failure.

Like many Oracle Cloud platform services, Database Cloud Service relies on an underlying component of Oracle Cloud named Platform Service Manager (PSM) to provide its service console and its REST API. As a result, you will find that the Oracle

Database Cloud Service console has the same “look and feel” as the service consoles for other platform services like Oracle GoldenGate Cloud Service and Oracle Java Cloud Service, and you will find that the endpoint structure and feature set of the Database Cloud Service REST API is very similar to those of the REST APIs for other platform services.

The Platform Service Manager (PSM) component of Oracle Cloud uses SSH to access the compute nodes that comprise your database deployments, in order to perform predefined Platform Service actions like backup and patching. You initiate these Platform Service actions from the web console or REST API. A separate SSH key pair is used for each database deployment to perform this internal communication. This SSH key is not available for ad hoc usage. You cannot delete this key from compute nodes or it will cause these Platform Service actions to fail. The key is only used under programmatic control and cannot be directly accessed by Oracle employees. All SSH actions performed by the Platform Service Manager (PSM) component on your compute nodes are logged and can be audited. The Oracle Cloud Operations team does not have access to any SSH keys residing on your compute nodes and has no way to access your compute nodes, unless you explicitly provide access to the keys for troubleshooting purposes.

Cloud Tooling for Database Cloud Service

In addition to the capabilities of the web-based Oracle Database Cloud Service console, Database Cloud Service offers the following tools on the database delpoyment's compute nodes:

- Simple Automated Backups: use the `bkup_api` utility (`raccli` on deployments that use Oracle Real Application Clusters) to perform on-demand backups and to change how automatic backups are configured. See Backing Up and Restoring Databases on Database Cloud Service.
- Simple Automated Recovery: use the `orec` subcommand of the `dbaascli` utility (`raccli` on deployments that use Oracle Real Application Clusters) to restore from backups. See Backing Up and Restoring Databases on Database Cloud Service.
- Simple Automated Patching: use the `patch` subcommand of the `dbaascli` utility (`raccli` on deployments that use Oracle Real Application Clusters) to apply patches. See Patching Database Cloud Service.
- SQL Developer Web: use the Oracle SQL Developer Web browser-based application to monitor the Oracle database and computing resources. See Using Oracle SQL Developer Web in Database Cloud Service. Oracle SQL Developer Web is not available on deployments that use Oracle Real Application Clusters.

About Database Cloud Service Database Deployments

When you create a new database deployment on Oracle Database Cloud Service, you use the Create Instance wizard, which steps you through the process of making the choices that produce a database deployment tailored to your needs. These choices include:

- Oracle Database Software Release
- Oracle Database Software Edition
- Oracle Database Type
- Computing Power

- Database Storage
- Automatic Backup Configuration

Oracle Database Software Release

When creating a database deployment on Oracle Database Cloud Service, you choose one of the following Oracle Database software releases:

- **Oracle Database 11g Release 2**
- **Oracle Database 12c Release 1**
- **Oracle Database 12c Release 2**
- **Oracle Database 18c**

> **Note:**
>
> The Oracle Database Cloud Service supports several database versions as part of the service provisioning process. The 12c and 18c database versions as licensed perpetually or by term are under normal support and maintenance and these 12c and 18c versions selected for provisioning included in the Oracle Database Cloud Service are fully supported as part of the service subscription. Oracle Database 11.2.0.4 as licensed perpetually or by term is also selectable for provisioning in the Oracle Database Cloud Service and Oracle will continue to support this release version as part of the service subscription for the duration of the Extended Support period for 11.2 as defined in Oracle's Lifetime Support Policy.

Oracle Database Software Edition

When creating a database deployment on Oracle Database Cloud Service, you choose one of the following Oracle Database software editions. For detailed information about the included Oracle Database features, options and packs, see the Permitted Features section of *Oracle Database Licensing Information User Manual*.

- **Standard Edition**—Oracle Database Standard Edition, which delivers unprecedented ease of use, power, and performance for workgroup, department-level, and Web applications. It includes all the facilities necessary to build business-critical applications.
- **Enterprise Edition**—Oracle Database Enterprise Edition, which provides the performance, availability, scalability, and security required for mission-critical applications such as high-volume online transaction processing (OLTP) applications, query-intensive data warehouses, and demanding Internet applications.
- **Enterprise Edition - High Performance**—provides all the features of Enterprise Edition, plus many of the available Oracle Database options and management packs.
- **Enterprise Edition - Extreme Performance**—provides all the features of Enterprise Edition, plus all of the Oracle Database options and management packs appropriate for use in Oracle Database Cloud Service.

> **Note:**
>
> If you choose Enterprise Edition or Enterprise Edition - High Performance, all available database enterprise management packs and Enterprise Edition options are included in the database deployment. The packs and options that are not part of the software edition you chose are available to you for use on a trial basis.

Oracle Database Type

When creating a database deployment on Oracle Database Cloud Service, you choose one of the following database types:

- **Single Instance**—A single Oracle Database instance and database data store hosted on one compute node. For more information about this type, see Characteristics of a Single Instance Database Deployment.
- **Database Clustering with RAC**—A two-node clustered database using Oracle Real Application Clusters technology; two compute nodes each host an Oracle Database instance, and the two instances access the same shared database data store. For more information about this type, see Characteristics of a Database Clustering with RAC Database Deployment.

  (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer)
- **Single Instance with Data Guard Standby**—Two single-instance databases, one acting as the primary database and one acting as the standby database in an Oracle Data Guard configuration. For more information about this type, see Characteristics of a Single Instance with Data Guard Standby Database Deployment.
- **Database Clustering with RAC and Data Guard Standby**—Two two-node Oracle RAC databases, one acting as the primary database and one acting as the standby database in an Oracle Data Guard configuration. For more information about this type, see Characteristics of a Database Clustering with RAC and Data Guard Standby Database Deployment.

  (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer)
- **Data Guard Standby for Hybrid DR** — Single-instance database acting as the standby database in an Oracle Data Guard configuration. The primary database is on your own system.

  (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer)

Not all types are available with all software editions:

- Single Instance is the only type supported by the Standard Edition software edition.
- The two types that use Oracle RAC are available only with Enterprise Edition - Extreme Performance software edition.

Computing Power

When creating a database deployment on Oracle Database Cloud Service, you choose the computing power for the associated compute node or nodes from a list of

supported OCPU (Oracle CPU) and processor RAM combinations. The combinations you can choose from depend on which infrastructure you are using for the deployment:

- For deployments on Oracle Cloud Infrastructure:

  View a list of currently available shapes by going to the Pricing page for Oracle Cloud Infrastructure Compute on cloud.oracle.com. The “BM.Standard” shapes in the **Compute - Bare Metal Instances** table and the “VM.Standard” shapes in the **Compute - Virtual Machine Instances** table are available in Oracle Database Cloud Service.

- For deployments on Oracle Cloud Infrastructure Classic:

  View a list of currently available shapes by going to the Pricing page for Oracle Cloud Infrastructure Compute Classic - General Purpose Compute on cloud.oracle.com. The shapes in the **General Purpose Shapes** and the **High Memory Shapes** tables are available in Oracle Database Cloud Service.

> **Note:**
>
> Not all shapes are available in all regions or sites.

Database Storage

When creating a database deployment on Oracle Database Cloud Service, you choose the amount of usable data storage you want for your database. After you create the database deployment, you can add more data and local backup storage as needed to create much larger databases. For information, see Scaling Up the Storage for a Database Deployment.

How much storage you can allocate when creating a deployment and how much storage you can add later both depend on which infrastructure you are using for the deployment:

- For Oracle Cloud Infrastructure:
  - When creating a deployment: you can create a database of up to 9600 GB (9.3 TB) with backups to both cloud and local storage or up to 16 TB with backups to cloud storage only or no backups.
  - By adding more storage: 28 scale-up operations, each of up to 16 TB, are supported. Thus, the deployment can accommodate a database of up to 158 TB with backups to both cloud and local storage or up to 386 TB with backups to cloud storage only or no backups. However, if you need databases of such large sizes, you should consider using Oracle Database Exadata Cloud Service instead of Oracle Database Cloud Service.
- For Oracle Cloud Infrastructure Classic:
  - When creating a deployment: you can create a database of up to 1200 GB with backups to both cloud and local storage or up to 2048 GB (2 TB) with backups to cloud storage only or no backups.
  - By adding more storage: 5 scale-up operations, each of up to 2 TB, are supported. Thus, the deployment can accommodate a database of up to 4.7 TB with backups to both cloud and local storage or up to 10 TB (7.7 TB in Oracle RAC deployments) with backups to cloud storage only or no backups.

Automatic Backup Configuration

When creating a new database deployment on Oracle Database Cloud Service, you choose whether you want automatic backups to be configured for the database. Your choices are:

- **Both Cloud Storage and Local Storage**—30 days' worth of backups are kept, with the 7 most recent days' worth available directly on the compute node's local storage.
- **Cloud Storage Only**—30 days' worth of backups are kept, with all backups on cloud storage.

> **Note:**
>
> This choice is not currently available for database deployments that use Oracle Real Application Clusters (Oracle RAC).

- **None**—automatic backups are not configured.

If you choose either of the cloud storage options, you must specify a cloud storage location:

- For deployments in Oracle Cloud Infrastructure, specify an existing Oracle Cloud Infrastructure Object Storage bucket.
- For deployments in Oracle Cloud Infrastructure Classic, specify an existing Oracle Cloud Infrastructure Object Storage Classic container or create one as part of the database deployment creation.

Guidelines for Administering Database Cloud Service Database Deployments

When you create a database deployment, Oracle Database Cloud Service creates a complete environment for you, including a running Oracle database and automatic scheduled backups, all configured according to values you provided when creating the deployment. You have root privilege to the compute nodes of the deployment and you have full administrative privileges for the Oracle database.

You are responsible for administering the database deployment. To simplify your administrative duties, Database Cloud Service provides service automation tooling (also called cloud tooling) for a variety of operations like backup, recovery, patching and upgrade. This cloud tooling takes two forms:

- The web-based Oracle Database Cloud Service console, which provides an easy-to-use graphical interface to perform database deployment lifecycle and management tasks.
- Utilities and scripts on the compute nodes of a database deployment. The Oracle Database Cloud Service console interacts with these utilities and scripts to perform management tasks, and you can use them as well. Examples of these utilities and scripts are `bkup_api`, `dbaascli` and `raccli`.

These on-node utilities and scripts are configured to work optimally with the deployment as it was initially configured. When you make changes to the deployment's configuration, you should always use the cloud tooling in favor of underlying Oracle Database utilities. In this way, the on-node tooling will remain in sync with the deployment's configuration.

Additionally, you should follow these guidelines to ensure that your deployment continues to run smoothly and as expected.

- Do not create additional installations of Oracle Database software.

  Each deployment comes with Oracle Database software already installed.

- Do not create additional Oracle databases.

  Each deployment is intended to house the one Oracle database (or, in the case of Oracle Data Guard configurations, the one primary Oracle database and one standby Oracle database) created when the deployment was created.

- Do not change fundamental characteristics of the database, such as the DB name (SID).

- For deployments that use Oracle Data Guard, use only cloud tooling to update the password of the SYS and SYSTEM database users.

  For Data Guard configurations of two single-instance databases, use dbaascli database changepassword. For Data Guard configurations of two Oracle RAC databases, use raccli update databasepassword.

- Apply **only** patches that are available through Database Cloud Service.

  Do **not** apply patches from any other source unless directed to by Oracle Support. This best practice applies to both database and OS patches.

- Apply database and OS patches when they become available through Database Cloud Service.

  For information about applying database patches, see Patching Database Cloud Service; for OS patches, see Applying Linux OS Security Patches.

- Update the cloud tooling on a deployment regularly.

  On a monthly basis you should check for and apply updates to the cloud tooling as described in Updating the Cloud Tooling on Database Cloud Service.

- Do not disable or close access to the SSH port (port 22).

  You can open other ports and protocols. See About Network Access to Database Cloud Service for information about the available options.

- Do not directly modify a compute node's user or SSH settings.

  Database Cloud Service configures default OS users and Secure Shell (SSH) access settings during the creation of a deployment. Do not modify these default users and use only the features of Database Cloud Service to modify the SSH keys for these users.

- Do not modify the default administration ports.

  Do not change the ports for Oracle Application Express, Oracle Net Listener, Enterprise Manager Database Express 18c, Enterprise Manager Database Express 12c, or Enterprise Manager 11g Database Control after the deployment is created.

- When specifying backups to cloud storage, use a different cloud storage container or bucket for each deployment.
- Do not manually remove files from a cloud storage container or bucket that is being used for backups.
- Always use the cloud tooling when changing the configuration of automatic backups. In particular, for deployments on Oracle Cloud Infrastructure Classic, always use the cloud tooling to change the password used to access the cloud storage container for backups, as described in Updating the Password for Backing Up to the Storage Cloud.
- Do not detach, change file access permissions for, or change the mount point of any storage volume that was attached to a compute node during the creation of your deployment. In particular, do not unmount or change the file access permissions of `/u01` through `/u05`.

  For details about these volumes, see Storage Volumes and File System Layout.
- When adding storage to a deployment configured for local backups, be sure add both FRA storage and Data storage to maintain a 1.7-to-1 ratio of FRA storage to Data storage. That is, the FRA storage size should be 1.7 times the Data storage size.
- When adding storage to a deployment, especially on Oracle Cloud Infrastructure Classic, avoid adding small amounts (under 10GB) of storage.

  Oracle Cloud Infrastructure Classic supports 10 storage volumes attached to a compute node, of which 5 are used when the database deployment is created. Thus, you have only 5 opportunities to scale up storage.
- When adding storage to a deployment on Oracle Cloud Infrastructure Classic, use the Oracle Database Cloud Service cloud tooling, as described in Scaling Up the Storage for a Database Deployment.

  Do not use the Oracle Cloud Infrastructure Compute Classic cloud tooling unless you are adding temporary storage to use for a specific short-term period and then remove, as described in Adding Temporary Storage to a Database Deployment.

Before You Begin with Database Cloud Service

Before you begin using Oracle Database Cloud Service, you should be familiar with the following technologies:

- Oracle Cloud

  See *Getting Started with Oracle Cloud*.

Before you create a Database Cloud Service database deployment:

- On Oracle Cloud, sign up for a free credit promotion or purchase a subscription. You cannot create a Database Cloud Service database deployment until you do so.
- If your Cloud account supports Oracle Cloud Infrastructure, perform the steps in Prerequisites for Oracle Platform Services on Oracle Cloud Infrastructure in the Oracle Cloud Infrastructure documentation.
- If your Cloud account supports Oracle Cloud Infrastructure Classic, set the replication policy in Oracle Cloud Infrastructure Object Storage Classic, as

described in Selecting a Replication Policy for Your Service Instance in *Using Oracle Cloud Infrastructure Object Storage Classic*.

- (Optional) Create a Secure Shell (SSH) public/private key pair. The SSH keys are used to facilitate secure access to the compute nodes that support your database deployments. See Generating a Secure Shell (SSH) Public/Private Key Pair.
- (Optional) Create a cloud storage backup location

  If you want to automatically back up your database to cloud storage, you must associate it with a cloud storage backup location. The type of location you specify depends on the infrastructure the deployment is built on:

  – **Oracle Cloud Infrastructure:** cloud backups are stored in an Oracle Cloud Infrastructure Object Storage bucket. You must create a storage bucket before you create database deployments configured to back up to cloud storage.

  – **Oracle Cloud Infrastructure Classic:** cloud backups are stored in an Oracle Cloud Infrastructure Object Storage Classic container. You can create the container beforehand and provide the wizard with information about it, or you can have the wizard create the container for you.

How to Begin with Database Cloud Service Subscriptions


This topic does not apply to Oracle Cloud at Customer.

Here's how to get started with Oracle Database Cloud Service free promotions and subscriptions:

1. Sign up for a free credit promotion or purchase a subscription.

   See Requesting and Managing Free Oracle Cloud Promotions or Buying an Oracle Cloud Subscription in *Getting Started with Oracle Cloud*.

2. Open the Oracle Database Cloud Service console.

   See Accessing the Oracle Database Cloud Service Console.

> **Note:**
>
> Be sure to review the prerequisites described in Before You Begin with Database Cloud Service before you create your first database deployment on Database Cloud Service . Depending on which Oracle Cloud infrastucture services are available in your account, you may have to perform some set-up steps before you create your first database deployment.

If you want to grant others access to Database Cloud Service, start by reviewing About Database Cloud Service Roles and Users. Then, create accounts for users and assign them appropriate privileges and roles. For instructions, see Adding Users and Assigning Roles in *Getting Started with Oracle Cloud*.

About Database Cloud Service Roles and Users

In addition to the roles and privileges described in Oracle Cloud User Roles and Privileges in *Getting Started with Oracle Cloud*, the **DBaaS Database Administrator** role is created for Oracle Database Cloud Service.

When the Database Cloud Service account is first set up, the service administrator is given this role. User accounts with this role must be added before anyone else can access and use Database Cloud Service.

The identity domain administrator can create more Database Cloud Service administrators by creating user accounts and assigning them the DBaaS Database Administrator role. See Managing User Accounts in *Managing and Monitoring Oracle Cloud*.

The following table summarizes the privileges given to the DBaaS Database Administrator role.

Description of Privilege	More Information
Can create and delete database deployments	Creating a Customized Database Deployment Deleting a Database Deployment
Can scale, patch, and back up or restore database deployments	Scaling a Database Deployment Patching Database Cloud Service Backing Up and Restoring Databases on Database Cloud Service
Can monitor and manage service usage in Oracle Cloud	Managing and Monitoring Oracle Cloud Services in *Managing and Monitoring Oracle Cloud*

Accessing the Oracle Database Cloud Service Console

To access the Oracle Database Cloud Service console:

1. Sign in to your Cloud Account and go to the My Services Dashboard.

   See Signing in to Your Cloud Account in *Getting Started with Oracle Cloud*.

2. Click the navigation menu in the top corner of the My Services Dashboard and then click **Database Classic**.

   The Oracle Database Cloud Service console opens.

3. If a Welcome page is displayed, go to the Instances page by clicking **Instances** next to “Database Cloud Service”.

Using the Database Cloud Service REST APIs

You can programmatically provision and manage Oracle Database Cloud Service instances and associated database deployments by using REST (REpresentational State Transfer) application programming interfaces (APIs).

Each REST API call maps to a HTTP request: getting an object (`GET`), adding an object (`POST`), updating an object (`PUT`), and deleting an object (`DELETE`). The HTTP response

code indicates whether the request was successful. Each object for which you can perform the `GET`, `POST`, `PUT`, and `DELETE` requests is identified uniquely by its URI.

To access Database Cloud Service by using the REST API you must use the REST endpoint URL that is associated with your service instance. For details, see *REST API for Oracle Database Cloud Service*.

Typical Workflow for Using Database Cloud Service

To start using Oracle Database Cloud Service, refer to the following tasks as a guide:

Task	Description	More Information
Sign up for a free credit promotion or purchase a subscription (Does not apply to Oracle Cloud at Customer)	Provide your information, and sign up for a free credit promotion or purchase a subscription to Oracle Database Cloud Service.	How to Begin with Database Cloud Service Subscriptions
Add and manage users and roles	Create accounts for your users and assign them appropriate privileges. Assign the necessary Database Cloud Service roles.	Adding Users and Assigning Roles in *Getting Started with Oracle Cloud*, and About Database Cloud Service Roles and Users
Create a database deployment	Use a wizard to create a new database deployment.	Creating a Customized Database Deployment
Enable network access	Permit access to network services associated with your database deployments.	About Network Access to Database Cloud Service
Load data into the database	Use standard Oracle Database tools to load data into your databases.	Loading Data into the Oracle Database on Database Cloud Service
Monitor database deployments	Check on the health and performance of individual database deployments.	Monitoring and Managing Oracle Database on Database Cloud Service
Monitor the service	Check on the day-to-day operation of your service, monitor performance, and review important notifications.	Managing and Monitoring Oracle Cloud Services in *Managing and Monitoring Oracle Cloud*
Patch a database deployment	Apply a patch or roll back a patch.	Patching Database Cloud Service
Back up a database deployment	Back up a database or restore a database from a backup.	Backing Up and Restoring Databases on Database Cloud Service

About Database Deployments in Oracle Cloud Infrastructure

You can create Oracle Database Cloud Service database deployments in Oracle Cloud Infrastructure and in Oracle Cloud Infrastructure Classic.

The Oracle Database environment that your database deployment provides in either type of infrastructure is substantially the same. A few differences exist in the underlying infrastructure components and in the supported capabilities. Awareness of these differences will help you choose an appropriate infrastructure when creating a database deployment.

- **Prerequisite steps before creating your first database deployment:** In Oracle Cloud Infrastructure, you must perform the steps described in Prerequisites for Oracle Platform Services on Oracle Cloud Infrastructure in the Oracle Cloud

Infrastructure documentation. In Oracle Cloud Infrastructure Classic, you must set the replication policy in Oracle Cloud Infrastructure Object Storage Classic before you create database deployments with backups to cloud storage, as described in Selecting a Replication Policy for Your Service Instance in *Using Oracle Cloud Infrastructure Object Storage Classic*.

- **Regions and availability domains:** While creating a database deployment, you select a region in Oracle Cloud Infrastructure Classic or Oracle Cloud Infrastructure. If you select a region in Oracle Cloud Infrastructure, then you also select an *Availability Domain*. A region in Oracle Cloud Infrastructure has multiple isolated availability domains, each with separate power and cooling. The availability domains within a region are interconnected using a low-latency network.
- **Subnets and IP networks:** In Oracle Cloud Infrastructure Classic, you can optionally attach a database deployment to an IP network that you define beforehand. In Oracle Cloud Infrastructure, you must attach each database deployment to a subnet, which is a part of a virtual cloud network that you create in Oracle Cloud Infrastructure.
- **Database types:** Currently, Oracle Cloud Infrastructure supports only the **Single Instance** and **Single Instance with Data Guard Standby** database types.
- **Compute shapes:** The range of compute shapes that you can select from when creating a database deployment is different for Oracle Cloud Infrastructure and Oracle Cloud Infrastructure Classic. For more information, see Computing Power.
- **Backups to cloud storage:** In Oracle Cloud Infrastructure, an Oracle Cloud Infrastructure Object Storage *bucket* is used to store backups to cloud storage. You must create a storage bucket before you create database deployments configured to back up to cloud storage. In Oracle Cloud Infrastructure Classic, an Oracle Cloud Infrastructure Object Storage Classic *container* is used to store backups to cloud storage. You can create a storage container before you create database deployments configured to back up to cloud storage, or you can have a storage container created at the same time as a database deployment.
- **IP reservations:** Currently, Oracle Cloud Infrastructure does not support IP reservations.
- **Network access to database deployments:** Regardless of the infrastructure that you create your database deployment in, the rules to provide network access to the deployment are preconfigured for you. The interfaces that you use to manage these rules depend on the infrastructure that the deployment is created in:
  - For deployments in Oracle Cloud Infrastructure, you configure the rules, called *security rules*, in the Oracle Cloud Infrastructure interfaces.
  - For deployments in Oracle Cloud Infrastructure Classic, you configure the rules, called *access rules*, in the Oracle Database Cloud Service interfaces. Note that these access rules prohibit access by default (with the exception of SSH access on port 22), and you must enable them to provide access to other ports.
- **Scaling database deployments:** In Oracle Cloud Infrastructure, you cannot scale the shape of a database deployment's compute nodes; you can scale only the storage. You can create new storage volumes of 50 GB to 2048 GB in 50 GB increments. You can scale up storage 26 times, for a total of 30 storage volumes attached to the deployment.

- **Snapshots and deployments cloned from snapshots:** Currently, Oracle Cloud Infrastructure does not support creating storage snapshots of single-instance database deployments and then creating cloned deployments from the snapshots.

2

Managing the Database Cloud Service Life Cycle

This section describes tasks to manage the life cycle of Oracle Database Cloud Service.

Topics

Creating a Database Deployment

There are several ways in which you can create a database deployment on Oracle Database Cloud Service, depending on your requirements and experience level.

Choose from one of the following deployment creation methods:

Create Method	More Information
The fastest and easiest way to create a database deployment is to use a QuickStart template.	Creating a QuickStart Database Deployment
The way to create a database deployment where you can customize all options is to use the Create Instance wizard.	Creating a Customized Database Deployment

Create Method	More Information
Ways to create a database deployment whose database is populated, or “instantiated”, using data from the cloud backup of another database.	Creating a Database Deployment Using a Cloud Backup
The way to create a clone of an existing database deployment is to use snapshots.	Creating and Managing Snapshots of a Database Deployment to create the snapshot, and then Creating a Clone Database Deployment from a Snapshot to create the clone
The way to create an Oracle Data Guard standby database for an on-premises primary database is to use Hybrid DR.	Creating a Hybrid DR Deployment

Creating a QuickStart Database Deployment

 This topic does not apply to Oracle Cloud Infrastructure.

For accounts that use the Universal Credits payment model, Oracle Database Cloud Service provides QuickStart templates that create database deployments of commonly used configurations. You simply pick a template and, if desired, change the default deployment name. Database Cloud Service then uses an Oracle Cloud Stack template to provide all the other configuration information.

> **Note:**
>
> QuickStart database deployments use the "Bring Your Own License" feature. This feature enables you to use an existing perpetual Oracle Database license to establish the right to use Oracle Database on a deployment. If you do not wish to use this feature, you must create a customized deployment. Follow the steps in Creating a Customized Database Deployment and, for the **License Type** option, select **Subscribe to a new Oracle Database software license and the Oracle Database Cloud Service.**

Procedure

1. Go to the QuickStarts page:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the **QuickStarts** link.

      The QuickStarts page is displayed.
2. If desired, change the **Instance Name** from the default one that’s provided. The name must start with a letter, and can contain up to 25 letters or numbers.
3. Review the information about the available templates and decide which one you want to use.

For more information about the database deployments these templates create, see Standard Edition, Enterprise Edition, and Extreme Performance.

4. Click the **Create** button below the template you want to use.

   The Confirmation window is displayed. Note that the **Create** button is disabled.

5. Click the **Download** link and save the zip file containing the SSH key pair and administrator password that will be used to create your deployment. You will need this information to access your database deployment after it is created.

   The **Create** button is now enabled.

6. Click **Create**.

   The Confirmation window closes, Database Cloud Service begins creating the deployment using theOracle Cloud Stack template, and the Services page is displayed.

7. Periodically refresh the Services page to monitor the creation of your new database deployment.

Standard Edition

When you use the Standard Edition QuickStart option, Database Cloud Service uses an Oracle Cloud Stack template to create a database deployment consisting of a single-instance Oracle Database 12.2 Standard Edition database housed on a single compute node. In brief, the template performs these actions:

- Creates a compute node of 1 OCPU with 7.5 GB RAM and its own public IP address.
- Installs Oracle Linux, Oracle Database 12.2.0.1 Standard Edition, and cloud tooling software on the compute node.
- Creates storage for 15 GB of database data and storage for the fast recovery area and redo logs needed for the database.
- Creates Oracle Cloud Infrastructure Compute Classic networking resources to provide access to the compute node, setting all except SSH access on port 22 to a disabled status.
- Creates an Oracle database with the SID (System ID) ORCL and a single PDB (pluggable database) named PDB1.
- Configures Oracle Net Services to listen for database connections on port 1521.
- Starts the database and starts the Oracle Net Services listener.

Note that the template **does not** configure automatic backups for the database deployment. To set up automatic backups, see Changing the Backup Configuration on Database Deployments Hosting Single-Instance Databases.

Enterprise Edition

When you use the Enterprise Edition QuickStart option, Database Cloud Service uses an Oracle Cloud Stack template to create a database deployment consisting of a single-instance Oracle Database 12.2 Enterprise Edition database housed on a single compute node. In brief, the template performs these actions:

- Creates a compute node of 2 OCPUs with 15 GB RAM and its own public IP address.

- Installs Oracle Linux, Oracle Database 12.2.0.1 Enterprise Edition, and cloud tooling software on the compute node.
- Creates storage for 512 GB of database data and storage for the fast recovery area and redo logs needed for the database.
- Creates Oracle Cloud Infrastructure Compute Classic networking resources to provide access to the compute node, setting all except SSH access on port 22 to a disabled status.
- Creates an Oracle database with the SID (System ID) ORCL and a single PDB (pluggable database) named PDB1.
- Configures Oracle Net Services to listen for database connections on port 1521.
- Starts the database and starts the Oracle Net Services listener.

Note that the template **does not** configure automatic backups for the database deployment. To set up automatic backups, see Changing the Backup Configuration on Database Deployments Hosting Single-Instance Databases.

Extreme Performance

When you use the Extreme Performance QuickStart option, Database Cloud Service uses an Oracle Cloud Stack template to create a database deployment consisting of a single-instance Oracle Database 12.2 Enterprise Edition database housed on a single compute node. In brief, the template performs these actions:

- Creates a compute node of 2 OCPUs with 15 GB RAM and its own public IP address.
- Installs Oracle Linux, Oracle Database 12.2.0.1 Enterprise Edition, and cloud tooling software on each compute node.
- Creates storage for 1 TB of database data and storage for the fast recovery area and redo logs needed for the database.
- Creates Oracle Cloud Infrastructure Compute Classic networking resources to provide access to the compute node, setting all except SSH access on port 22 to a disabled status.
- Creates an Oracle database with the SID (System ID) ORCL and a single PDB (pluggable database) named PDB1.
- Configures Oracle Net Services to listen for database connections on port 1521.
- Starts the database and starts the Oracle Net Services listener.

Note that the template **does not** configure automatic backups for the database deployment. To set up automatic backups, see Changing the Backup Configuration on Database Deployments Hosting Oracle RAC Databases.

Creating a Customized Database Deployment

To create a customized database deployment on Oracle Database Cloud Service, use the Create Instance wizard as described in the following procedure.

However, before using the Create Instance wizard, you need to make sure that you have all of the necessary information, as described in Before You Begin. Additionally, after your database deployment is created you need to perform a few follow-on tasks

to make sure your deployment is accessible and up-to-date, as described in After Your Database Deployment Is Created.

Before You Begin

When you create a database deployment, you provide information used to create the deployment itself and the Oracle database it hosts. In addition, you may need to provide information about other resources:

- An SSH public/private key pair (Optional)

  You must associate an SSH public key with the compute infrastructure supporting the deployment. An SSH public key is used for authentication when you use an SSH client to connect to a compute node associated with the deployment. When you connect, you must provide the private key that matches the public key.

  You can have the wizard create a public/private key pair for you, or you can create one beforehand and upload or paste its private key value. If you want to create a key pair beforehand, you can use a standard SSH key generation tool. See Generating a Secure Shell (SSH) Public/Private Key Pair.

- A cloud storage backup location (Optional)

  If you want to automatically back up your database to cloud storage, you must associate it with a cloud storage backup location. The type of location you specify depends on the infrastructure the deployment is built on:

  – **Oracle Cloud Infrastructure:** cloud backups are stored in an Oracle Cloud Infrastructure Object Storage bucket. You must create a storage bucket before you create database deployments configured to back up to cloud storage.

  – **Oracle Cloud Infrastructure Classic:** cloud backups are stored in an Oracle Cloud Infrastructure Object Storage Classic container. You can create the container beforehand and provide the wizard with information about it, or you can have the wizard create the container for you.

    The container becomes associated with Oracle Database Backup Cloud Service, which Database Cloud Service uses to perform backups to cloud storage. Once associated with Oracle Database Backup Cloud Service, the container becomes part of your Oracle Database Public Cloud Services subscription (or trial) rather than part of an Oracle IaaS Public Cloud Services subscription (or trial).

    Whether you create the container beforehand or have the wizard do it for you, you are prompted for the following information about the container:

    * The name of the container
    * The user name and password of a user who has read/write access to the container

- A existing cloud backup created using Oracle Database Backup Cloud Service (Optional)

  If you are creating a database deployment hosting a single-instance database, you can use the “instantiate from backup” technique to have the new database populated from the data stored in the Database Backup Cloud Service backup of another Oracle database. For information about this technique, its requirements on the cloud backup and the characteristics of the instantiated database, see Creating a Database Deployment Using a Cloud Backup.

Note that after using this instantiate-from-backup technique, Oracle Application Express, DBaaS Monitor and ORDS (Oracle REST Data Services) may not be accessible. To restore accessibility, see Application Express, DBaaS Monitor and ORDS inaccessible after creating a database deployment using a cloud backup in *Known Issues for Oracle Database Cloud Service*.

Procedure

To create a database deployment on Database Cloud Service:

1. Open the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. Click **Create Instance**.

   The Create Instance wizard starts and the Instance page is displayed.

3. On the Instance page, provide a name and description for the database deployment, and provide information about its high-level characteristics. When you are finished, click **Next** to advance to the Instance Details page.

Element	Description
Instance Name	The name for the new database deployment. The name: • Must not exceed 50 characters. • Must start with a letter. • Must contain only letters, numbers, or hyphens. • Must not contain any other special characters. • Must be unique within the identity domain.
Description	(Optional) A description for the new database deployment.
Notification Email	(Optional) An email address where you would like updates about the deployment-creation operation to be sent.
Region	(Available only if your identity domain is enabled for regions.) The region for the database deployment. If you choose a region that supports Oracle Cloud Infrastructure, the **Availability Domain** and **Subnet** fields are displayed, and the deployment will be created on Oracle Cloud Infrastructure. Otherwise, the deployment will be created on Oracle Cloud Infrastructure Classic. Choose **No Preference** to let Database Cloud Service choose an Oracle Cloud Infrastructure Classic region for you.
Availability Domain	(Available only on Oracle Cloud Infrastructure) The availability domain (within the region) where the database deployment will be placed.
Subnet	(Available only on Oracle Cloud Infrastructure) The subnet (within the availability domain) that will determine network access to the database deployment.

Element	Description
IP Network	(Available only if you have selected an Oracle Cloud Infrastructure Classic region and you have defined one or more IP networks created in that region using Oracle Cloud Infrastructure Compute Classic.) Select the IP network where you want the database deployment placed. Choose **No Preference** to use the default shared network provided by Oracle Cloud Infrastructure Compute Classic. For more information about IP networks, see these topics in *Using Oracle Cloud Infrastructure Compute Classic*: • About IP Networks • Creating an IP Network
Assign Public IP	(Available only if you have selected an IP network.) Choose whether to assign public IP addresses to the compute nodes in your database deployment. If you select this check box (default), then any node added during deployment creation, or later added as part of a scaling operation, will have a public IP address assigned to it. You will be able to directly access the nodes from the public Internet. If you deselect this check box, then any node added during deployment creation, or later added as part of a scaling operation, will not have a public IP address assigned to it. You will not be able to directly access the nodes from the public Internet. This selection is for use cases where you intend to access the nodes and the database only from within your IP network or from your on-premises data center over a VPN network.
License Type	(Available only in accounts that use the Universal Credits payment model) Controls how the right to use Oracle Database on the new deployment is established. To use the "Bring Your Own License" (BYOL) feature, which enables you to use an existing perpetual Oracle Database license to establish the right to use Oracle Database on a deployment, select **My organization already owns Oracle Database software licenses. Bring my existing database software license to the Oracle Database Cloud Service.** Your Oracle Cloud account will be charged a lesser amount for the new deployment because the right to use Oracle Database is covered by your perpetual license agreement. To use your Oracle Cloud account, select **Subscribe to a new Oracle Database software license and the Oracle Database Cloud Service.** Your account will be charged for the new deployment according to your Oracle Database Cloud Service agreement.
Service Level	(Available only in accounts that include Oracle Database Exadata Cloud Service or old accounts that predate the Universal Credits payment model.) The service level for the new deployment: • **Oracle Database Cloud Service** is the service level you should choose for Database Cloud Service. • **Oracle Database Cloud Service - Virtual Image** (Not available on Oracle Cloud Infrastructure)

Element	Description
Metering Frequency	(Available only in old accounts that predate the Universal Credits payment model.) The metering frequency for the new deployment: • **Hourly** • **Monthly**
Software Release	The release version of Oracle Database for the new deployment: • **Oracle Database 11g Release 2** • **Oracle Database 12c Release 1** • **Oracle Database 12c Release 2** • **Oracle Database 18c** See Oracle Database Software Release.
Software Edition	The Oracle Database software package for the new deployment: • **Standard Edition** • **Enterprise Edition** • **Enterprise Edition - High Performance** • **Enterprise Edition - Extreme Performance** See Oracle Database Software Edition.

Element	Description
Database Type	The type of deployment to create: • **Single Instance**—A single Oracle Database instance and database data store hosted on one compute node. For more information about this type, see Characteristics of a Single Instance Database Deployment. • **Database Clustering with RAC**—A two-node clustered database using Oracle Real Application Clusters technology; two compute nodes each host an Oracle Database instance, and the two instances access the same shared database data store. For more information about this type, see Characteristics of a Database Clustering with RAC Database Deployment. (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer) • **Single Instance with Data Guard Standby**—Two single-instance databases, one acting as the primary database and one acting as the standby database in an Oracle Data Guard configuration. For more information about this type, see Characteristics of a Single Instance with Data Guard Standby Database Deployment. • **Database Clustering with RAC and Data Guard Standby**—Two two-node Oracle RAC databases, one acting as the primary database and one acting as the standby database in an Oracle Data Guard configuration. For more information about this type, see Characteristics of a Database Clustering with RAC and Data Guard Standby Database Deployment. (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer) • **Data Guard Standby for Hybrid DR** — Single-instance database acting as the standby database in an Oracle Data Guard configuration. The primary database is on your own system. (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer) Not all types are available with all combinations of service level and software edition: • Single Instance is the only type supported by the Standard Edition software edition. • The two types that use Oracle RAC are available only with Enterprise Edition - Extreme Performance software edition.

4. On the Instance Details page, complete the **Database Configuration** section by providing information about the Oracle Database you want to create.

Element	Description
DB Name (SID)	The name for the database instance. The name: • Must not exceed 8 characters. • Must start with a letter. • Must contain only letters and numbers.

Element	Description
PDB Name	(Available only for Oracle Database 12c or later.) The name for the default pluggable database (PDB). The name: • Must not exceed 8 characters. • Must start with a letter. • Must contain only letters, numbers, or these symbols: _ (underscore), # (hash sign), or $ (dollar sign). This option is not available if Create Instance from Existing Backup is set to Yes.
Administration Password **Confirm Password**	The password for the following administrative users: • Oracle Database administrative users • Oracle Application Express admin user The password: • Must be 8 to 30 characters in length. • Must contain at least one lowercase letter • Must contain at least one uppercase letter • Must contain at least one number • Must contain at least one of these symbols: _ (underscore), # (hash sign), or $ (dollar sign). • Must not contain the word “oracle”.
Usable Database Storage (GB)	The amount of storage in GB for actual database data. Note that up to 8% of this storage will be used for file system constructs and other overhead.
Total Data File Storage (GB)	The computed amount of storage in GB that will be allocated to the deployment, including space for operating system and product binaries, supporting files, database data and configuration files, and so on.
Compute Shape	The number of Oracle Compute Units (OCPUs) and amount of memory (RAM) for each compute node of the new database deployment. Database Cloud Service offers several OCPU/RAM combinations, as described in Computing Power.
SSH Public Key **Edit**	The SSH public key to be used for authentication when using an SSH client to connect to a compute node that is associated with your database deployment. Click **Edit** to specify the public key. You can upload a file containing the public key value, paste in the value of a public key, or create a system-generated key pair. If you paste in the value, make sure the value does not contain line breaks or end with a line break.
Use High Performance Storage	(Available only if you have a metered subscription and you chose an Oracle Cloud Infrastructure Classic region on the wizard’s Instance page.) Controls the device type to be used for database block storage. By default, block storage is allocated on spinning devices. If you select this option, then block storage will be allocated on solid state devices, at an increased cost. For pricing details, refer to the Block Storage information at https://cloud.oracle.com/compute-classic/pricing.
Advanced Settings: Listener Port	The port number for the Oracle Net Listener. The port number must be between 1521 and 5499 (inclusive).

Element	Description
Advanced Settings: Timezone	The time zone for the new database deployment. The default is Coordinated Universal Time (UTC).
Advanced Settings: Character Set	The database character set for the database. The database character set is used for: • Data stored in SQL `CHAR` data types (`CHAR`, `VARCHAR2`, `CLOB`, and `LONG`) • Identifiers such as table names, column names, and PL/SQL variables • Entering and storing SQL and PL/SQL source code This option is not available if Create Instance from Existing Backup is set to Yes.
Advanced Settings: National Character Set	The national character set for the database. The national character set is used for data stored in SQL `NCHAR` data types (`NCHAR`, `NCLOB`, and `NVARCHAR2`). This option is not available if Create Instance from Existing Backup is set to Yes.
Advanced Settings: Enable Oracle GoldenGate	(Not available on Oracle Cloud at Customer) Configures the database for use as the replication database of an Oracle GoldenGate Cloud Service instance. See Using Oracle GoldenGate Cloud Service with Database Cloud Service.
Advanced Settings: Include "Demos" PDB	(Available only for Oracle Database 12c Release 1.) Controls whether the "Demos" PDB is to be included in the database. This PDB contains demos for many new features of Oracle Database 12c such as in-memory and multitenant. Usable Data File Storage must to be at least 25 GB to include this PDB.
Advanced Settings: IP Reservations	(Not available on Oracle Cloud Infrastructure) (Available only if you chose an Oracle Cloud Infrastructure Classic region on the wizard's Instance page and did not deselect the **Assign Public IP** option.) Specifies whether to use an IP reservation for this deployment. If you choose **Assign Automatically**, an IP reservation is not used and Database Cloud Service acquires a new IP address for use by the deployment. Otherwise, Database Cloud Service uses the IP reservation you choose.

5. On the Instance Details page, complete the **Backup and Recovery Configuration** section, choosing a backup option for the database deployment and, depending on your choice, providing information about the Oracle Storage Cloud Service container where cloud backups are to be stored.

Element	Description
Backup Destination	Controls how backups for the deployment are to be configured: • **Both Cloud Storage and Local Storage**—backups are configured to be created automatically and stored both on local storage and on cloud storage. If this choice is selected, the Cloud Storage Container, User Name and Password fields are displayed. • **Cloud Storage Only** — backups are configured to be created automatically and stored on cloud storage. If this choice is selected, the Cloud Storage Container, User Name and Password fields are displayed. **Note:** This choice is not currently available for database deployments that use Oracle Real Application Clusters (Oracle RAC). • **None**—Backups are not configured for the deployment. For more information about backups and backup configurations, see About Backing Up Database Deployments on Database Cloud Service.

Cloud Storage Container	The location where backups to cloud storage are to be stored:
	• For database deployments in Oracle Cloud Infrastructure, enter the URL of an existing Oracle Cloud Infrastructure Object Storage bucket. The URL is of the form:
	`https://swiftobjectstorage.region.oraclecloud.com/v1/namespace/bucket`
	For example:
	`https://swiftobjectstorage.us-phoenix-1.oraclecloud.com/v1/mycompany/mybucket`
	You must create this storage bucket before you begin creating the database deployment. See Object Storage API in Oracle Cloud Infrastructure documentation.
	• For database deployments in Oracle Cloud Infrastructure Classic, enter the location of an Oracle Cloud Infrastructure Object Storage Classic container using this form:
	`Storage-identity_domain/container`
	where *identity_domain* is the id of the identity domain, and *container* is the name of the container. If this container doesn't exist, use the **Create Cloud Storage Container** checkbox to create it.
	Note: In some Oracle Cloud Infrastructure Classic accounts, you cannot use the above form. If you get an error when you try to use this form, you must instead provide a full URL for the container using this form:
	rest-endpoint-url/container
	To determine the *rest-endpoint-url* value for your account, see About REST URLs for Oracle Cloud Infrastructure Object Storage Classic Resources in *Using Oracle Cloud Infrastructure Object Storage Classic*.

Element	Description
Cloud Storage User Name	A user with read/write (and creation, if necessary) access to the location specified in **Cloud Storage Container**: • For database deployments in Oracle Cloud Infrastructure, enter the user name you use to sign in to the Oracle Cloud Infrastructure console. • For database deployments in Oracle Cloud Infrastructure Classic, enter the Oracle Cloud user name of the administrator of the Oracle Cloud Infrastructure Object Storage Classic container specified in **Cloud Storage Container**. Usually, this is your Oracle Cloud user name.
Cloud Storage Password	The password necessary to access the location specified in **Cloud Storage Container**: • For database deployments in Oracle Cloud Infrastructure, enter your Swift password (auth token). • For database deployments in Oracle Cloud Infrastructure Classic, enter the password of the Oracle Cloud user specified in **Cloud Storage User Name**.
Create Cloud Storage Container	(Not available on Oracle Cloud Infrastructure) Create a new Oracle Cloud Infrastructure Object Storage Classic container as part of the database deployment creation. Specify the container name and the Cloud Storage user name and password in the preceding fields.
Total Estimated Monthly Storage	Storage for data files and backups.

6. On the Instance Details page, complete the **Initialize Data From Backup** section if you are having the new database populated, or “instantiated”, from the data stored in the Database Backup Cloud Service backup.

Element	Description
Create Instance from Existing Backup	Create a database deployment whose database is derived from a cloud backup created using Oracle Database Backup Cloud Service. The other fields and options in the Initialize Data From Backup section only display if Create Instance from Existing Backup is set to Yes.
On-Premises Backup	Indicates the origin of the source database backup. Select this option if the source database backup is not from another Database Cloud Service database deployment in the same identify domain. In this case, the following fields and options are displayed except for Source Service Name. Deselect this option if the source database backup is from another Database Cloud Service database deployment in the same identify domain. In this case, only the Source Instance Name and Backup Tag fields are displayed.
Database ID	The database identifier of the database from which the existing backup was created. You can get this value by using the following SQL query: `SQL> SELECT dbid FROM v$database;`

Element	Description
Decryption Method **Edit**	Specifies the information necessary to decrypt the source database backup. Click **Edit** to specify the necessary information. In the resulting dialog: • For a backup that uses Transparent Database Encryption (TDE), select **Upload Wallet File** then click **Browse** and specify a zip file containing the source database's TDE wallet directory, and the contents of that directory. **Note:** If the source database is from another Database Cloud Service database deployment, its TDE wallet directory is `/u01/app/oracle/admin/dbname/tde_wallet`. • For a backup that uses password encryption, select **Paste RMAN Key Value** and paste the password (key value) used to encrypt the backup.

Cloud Storage Container	The URL where the existing backup is stored:
	• The URL of an Oracle Cloud Infrastructure Object Storage bucket. The URL is of the form:
	`https://swiftobjectstorage.region.oraclecloud.com/v1/namespace/bucket`
	For example:
	`https://swiftobjectstorage.us-phoenix-1.oraclecloud.com/v1/mycompany/mybucket`
	• The URL of an Oracle Cloud Infrastructure Object Storage Classic container. The URL is of the general form:
	`rest-endpoint-url/container`

Username	The Oracle Cloud user name of the administrator of the Oracle Cloud Infrastructure Object Storage Classic container specified in **Cloud Storage Container**.
Password	The password of the user specified in **Username**.
Source Instance Name	From the list of possible alternatives, specify the database deployment whose source database backup you want to use.
Backup Tag	A list of backups available for the specified database deployment. The latest backup is selected by default, but you can choose an earlier backup.

7. On the Instance Details page, complete the **Standby Database Configuration** section. When you are finished, click **Next** to advance to the Confirmation page.

Element	Description
Standby Database Configuration	Controls where the standby database is placed in relation to the primary database: • **High Availability**—The standby database is placed in a different availability domain from the primary database, thus providing isolation at the infrastructure level. • **Disaster Recovery**—The standby database is placed in a different data center from the primary database, thus providing isolation at the infrastructure level and geographical separation to support availability despite catastrophic events. See Using Oracle Data Guard in Database Cloud Service for more information. If you choose this option, the Enable Oracle GoldenGate option is disabled.

8. On the Confirmation page, review the information listed. If you are satisfied with the information, click **Create**.

   If you need to change the information, use the navigation bar or **Back** button at the top of the wizard to step back through the pages in the wizard. Click **Cancel** to cancel out of the wizard without creating a new database deployment.

After Your Database Deployment Is Created

After your database deployment is created, you should perform the following actions:

- **Enable network access to the deployment**

  By default, strict security restrictions limit network access to database deployments. To open access to applications and management tools, you need to enable predefined network security rules or create and enable your own network security rules. See Enabling Access to a Compute Node Port.

- **Update cloud tooling**

  While the base images used to create Database Cloud Service database deployments are updated regularly, it is possible that even more recent updates to the cloud tooling are available. Therefore, you should check for and apply any updates to the cloud tooling. See Updating the Cloud Tooling on Database Cloud Service.

- **Apply database patches**

  While the base images used to create Database Cloud Service database deployments are updated regularly, it is possible that a newer patch set update (PSU) or bundle patch (BP) is available. Therefore, you should check for and apply any database patches that are available. See Applying a Patch.

Creating a Database Deployment Using a Cloud Backup

You can create an Oracle Database Cloud Service database deployment whose database is instantiated from a cloud backup of an Oracle Database Cloud Service database or an on-premises Oracle database.

In brief, you create a Database Cloud Service database deployment hosting a single-instance database and then you replace the newly created database using another

database's cloud backup. This technique is called "instantiate from backup" and the database from which the cloud backup was made is called the "source database".

You can use a cloud backup from a source database at version 18, 12.2.0.1, 12.1.0.2 or 11.2.0.4 to replace a newly created cloud database of the same version. You can use a cloud backup from a version 12.1.0.2 on-premises source database to replace a newly created version 18 or 12.2.0.1 cloud database. You can also use a cloud backup from a version 12.2.0.1 on-premises source database to replace a newly created version 18 cloud database on the Oracle Database Cloud Service; this feature is not supported on Oracle Database Exadata Cloud Service. Thus, you can instantiate from backup and upgrade your database in one operation.

Requirements and Results

The source database must meet the following requirements:

- The latest PSU (patch set update) must be applied to the source database.
- If the database is at version 12.1.0.2 or later, it must be multitenant (CDB). Database Cloud Service does not support non-CDB databases that are at version 12.1.0.2 or later.
- The database from which the backup was made uses File System or ASM as its storage method for data files.

If the source database is an on-premises database, it must meet these additional requirements:

- The on-premises database host must be a Linux X64 (OEL 6 or OEL 7) system.
- The database backup must contain archivelogs. Otherwise, it is not possible to recover the database.
- The database character sets of your on-premises database and the Oracle Database Cloud Service database that you intend to replace must be compatible.
- Non-Oracle software on the on-premises database host must meet the following minimum release requirements:
  - Java: Release 7 or higher. Java must be in the default path.
  - Python: Above Release 2.6 and below Release 3.0.

After completing the instantiate-from-backup tasks, you will have a Database Cloud Service database deployment with the following characteristics:

- A single-instance database with the SID you specified when creating the deployment, but containing the data from the backup. Additionally, the database ID will be different from the original database's database ID.
- Data files in `/u02/app/oracle/oradata/SID`.
- Redo logs in `/u04/app/oracle/redo`.
- Fast recovery area (FRA) at `/u03/app/oracle/fast_recovery_area`.
- Memory parameters set based on the Compute Shape (OCPUs and RAM) you specified when creating the deployment.
- Oracle Net Listener configured with services for the database (and PDBs if the Oracle database is version 12*c* or later).
- If the newly created database is at version 18 or 12.2.0.1, and the source database backup was created using Transparent Data Encryption (TDE), then all

tablespaces in the newly created database will be encrypted using TDE, except `SYSAUX`, `SYSTEM`, and any tablespace whose name contains the string `UNDO`.

Procedure

You perform an instantiate-from-backup operation by following these steps:

1. Ensure that you have a suitable backup of the source database.
   - If the source database is an Oracle Database Cloud Service database, ensure that the database deployment has been backed up to cloud storage. For more information, see About Backing Up Database Deployments on Database Cloud Service.
   - If the source database is an on-premises Oracle database, ensure that the database is suitable for instantiation in the cloud and then create a cloud backup. For instructions, see Creating a Cloud Backup of an On-Premises Database.
2. Create a Database Cloud Service database deployment and replace the newly created database using the backup. Choose one of the following methods:
   - If the source database is smaller than the maximum storage you can allocate when creating a deployment, you can use the Create Instance wizard to perform an instantiate-from-backup operation as it creates a database deployment. For information on the maximum storage you can allocate when creating a deployment, see Database Storage. For instructions on using the wizard to perform an instantiate-from-backup operation, see Creating a Customized Database Deployment, specifically the step about completing the **Initialize Data From Backup** section on the wizard's Instance Details page.
   - For source databases of any size, perform an instantiate-from-backup operation by following these steps:
     a. Create a Database Cloud Service database deployment hosting a single-instance database. Because you will be replacing the database using a cloud backup, some constraints apply to the choices you make when creating the database deployment:
        - **Software Release**. Choose the Oracle Database release corresponding to the source database.
        - **Software Edition**. Choose at the minimum a software edition that supports the options used by the source database.
        - **Database Type**. Choose Single Instance.
        - **DB Name (SID)** (on the Instance Details page). You can specify any SID you want, but if you want to use the same SID as the database from which the existing backup was made, you must make sure the SID you provide matches exactly, including the case of letters. For example, if the existing backup's database SID is `Orcl`, you must use `Orcl`, not `orcl` or `ORCL`.
        - **Usable Database Storage (GB)** (on the Instance Details page). Specify at the minimum the amount of storage needed to accommodate the source database. If the source database is bigger than you can specify when creating the database deployment, create the deployment at maximium size and then scale up the data and local backup storage as required by following the instructions in Scaling Up the Storage for a Database Deployment.

- **Enable Oracle GoldenGate** and **Include "Demos" PDB** (both on the Instance Details page). Do not choose either of these options, as the database is going to be replaced.
- **Backup Destination** (on the Instance Details page). Choose the option you want, even None. The instantiate-from-backup technique works even if the deployment is not being backed up.
- **Create Instance from Existing Backup** (on the Instance Details page). Choose No. You will be replacing the database after the deployment is created.

b. If necessary, scale up the deployment's storage to accommodate the source database.

   For instructions, see Scaling a Database Deployment.

c. Replace the database on the deployment using a backup:

   - You can perform this step by using the Oracle Database Cloud Service console. For instructions, see Replacing the Database by Using the Oracle Database Cloud Service Console.
   - You can perform this step by using the `dbaasapi` utility on the deployment's compute node to perform `ibkup` actions to start and then monitor the replacement. For instructions, see Replacing the Database by Using ibkup Actions.

> **Note:**
>
> After performing an instantiate-from-backup operation, Oracle Application Express, DBaaS Monitor and ORDS (Oracle REST Data Services) may not be accessible. To restore accessibility, see Application Express, DBaaS Monitor and ORDS inaccessible after creating a database deployment using a cloud backup in *Known Issues for Oracle Database Cloud Service*.

Creating a Cloud Backup of an On-Premises Database

Use the `ibackup` utility to create a backup of an on-premises Oracle Database, which can then be used to replace an Oracle Database Cloud Service database.

The `ibackup` utility enables you to:

- Perform a pre-check of the on-premises database to ensure that you can generate a backup that is suitable for replacing a cloud database.
- Generate an Oracle Database backup, as well as additional files, that you can use to replace the database on a Database Cloud Service database deployment as part of an instantiate-from-backup operation.

Prerequisites

Ensure that the on-premises database you intend to back up, as well as the Database Cloud Service database you intend to replace, meet the requirements described in Requirements and Results.

Procedure

Perform these tasks:

1. Download a zip file containing the `ibackup` utility to the on-premises database host. Use `wget` on the on-premises database host to download the `OracleCloud_ibackup_Setup.zip` file from Oracle Cloud Infrastructure Object Storage Classic:

   ```
   $ wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/
   ibackup/OracleCloud_ibackup_Setup.zip
   ```

2. On the on-premises database host:
   a. Log in as the `oracle` user.
   b. Unzip the `ibackup.zip` or `OracleCloud_ibackup_Setup.zip` file. Files are extracted into the `ibackup` directory.
   c. Switch to the `root` user and run the following command to set the ownership of the files in the `ibackup` directory:

      ```
      # chown -R oracle:oinstall ibackup
      ```

   d. Return to being the `oracle` user and navigate to the `ibackup` directory:

      ```
      $ cd ibackup
      ```

   e. Edit the `backup.cfg` file as follows:
      - Set the encryption mode for the database backup. Set `TDE=y` if the database uses Transparent Data Encryption (TDE). Set `TDE=n` to use RMAN key encryption.
      - Set the value for `target_db` to `12.2.0.1`, `12.1.0.2`, or `11.2.0.4`, depending on the version of the Database Cloud Service database deployment where you intend to instantiate the backup.
      - Set the value for `oss_user` to `Storageadmin`.
      - Set the value for `oss_url` to `https://storage.oraclecorp.com/v1/Storage-dbcsdev/IBKUP`.
      - If you enter a password value for `oss_passwd`, the password will be obfuscated the first time you run the `ibackup` tool. If you do not enter a password value, you will be prompted for the password when you run the tool.
      - If `TDE=n`, set the `rman_key` value to the RMAN encryption key. Otherwise, leave this value blank.
3. Run a pre-check on the source on-premises database. The pre-check does not generate a backup file.

   ```
   $ ./ibackup -d dbname
   ```

   In the above command, *dbname* is the name of the source database. Examine the pre-check results.

4. Generate a backup:

```
$ ./ibackup -d dbname -b -i
```

Optionally, you can use the –f option to ignore fix-up log failures when generating a backup:

```
$ ./ibackup -d dbname -b -i -f
```

In addition to the Oracle Database backup, the following files are also generated in the `/var/opt/oracle/ibackup/ibkup` directory:

- `tde_wallet.zip` — The TDE wallet directory. This file is generated only if TDE was enabled in the on-premises database. Copy this file to a secure and accessible location. This file is required to import the Oracle backup in an instantiate-from-backup operation.
- `TDE_README.txt` — Instructions on how to unzip the `tde_wallet.zip` file. This is important because the instantiate-from-backup operation expects a defined structure for the TDE wallet directory.
- `Import.json` — Template file to import the backup using `ibkup` actions with the `dbaasapi` utility.
- `oss_file.cfg` — Oracle Cloud Infrastructure Object Storage Classic information used to save the backup.

Use these files when replacing the database on a Database Cloud Service database deployment as part of an instantiate-from-backup operation.

Replacing the Database by Using the Oracle Database Cloud Service Console

You can use the Oracle Database Cloud Service console to replace the database for a Database Cloud Service database deployment using an instantiate-from-backup operation.

Before You Begin

If you wish to replace your database using a backup from another currently operational Database Cloud Service database deployment in the same identity domain, then you must specify the source database deployment by selecting from a list of the available deployments.

> **Caution:**
>
> If the database you want to replace was created from a backup, then you must first use the `dbaasapi ibkup restore` to make the cloud database ready to accept the data from the target backup. See Use ibkup restore.

If you wish to replace your database using any other backup, you are prompted for the following information:

- The database ID of the backed-up database.
- The decryption method for the backup, which is the password associated with the backup for backups that use password encryption, or a zip file containing the source database's wallet directory and contents for backups that use Transparent Data Encryption (TDE).
- The name of the Oracle Cloud Infrastructure Object Storage Classic container where the backup is stored.
- The user name and password of an Oracle Cloud user who has read access to the container.

Procedure

1. Open the Instances page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. Click the database deployment whose database you wish to replace.

   The Oracle Database Cloud Service Overview page is displayed.

3. From the action menu ( ) next to the database deployment name, choose **Replace Database using Backup**.

   The Replace Database using Backup window is displayed.

4. Specify attributes in the Replace Database using Backup window:

   On-Premises Backup? — use this option to indicate the origin of the source database backup.

   If you select this option you are indicating that the source database backup is not from another currently operational Database Cloud Service database deployment in the same identify domain. In this case, the following fields and options are displayed:

   - **Database ID** — enter the database id of the database from which the existing backup was created. You can get this value by querying the backup source database as follows:

     ```
     SQL> SELECT dbid FROM v$database;
     ```

   - **Decryption Method** — provide the information necessary to decrypt the existing backup:

     – For a backup that uses Transparent Database Encryption (TDE), select **Upload Wallet File** then click **Browse** and specify a zip file containing the source database's TDE wallet directory, and the contents of that directory.

     > **Note:**
     >
     > If the source database is from another Database Cloud Service database deployment, its TDE wallet directory is `/u01/app/oracle/admin/dbname/tde_wallet`.

     – For a backup that uses password encryption, select **Paste RMAN Key Value** and paste the password (key value) used to encrypt the backup.

> **Note:**
>
> For database deployments using Oracle Database 12c Release 2 (12.2), or later, only backups using TDE are supported.

- **Cloud Storage Container** — enter the name of the Oracle Cloud Infrastructure Object Storage Classic container where the existing backup is stored; use this format:

  ```
  instance-id_domain/container
  ```

  where *instance* is the name of the Oracle Cloud Infrastructure Object Storage Classic instance, *id_domain* is the id of the identity domain, and *container* is the name of the container.
- **Username** — enter the user name of an Oracle Cloud user who has read access to the container specified in **Cloud Storage Container**.
- **Password** — enter the password of the user specified in **Username**.
- **Administration Password** and **Confirm Password** — enter and then re-enter a password for the Oracle Database SYS and SYSTEM users in the newly replaced database.

If you deselect **On-Premises Backup?** you are indicating that the source database backup is from another currently operational Database Cloud Service database deployment in the same identity domain. In this case, the following fields are displayed:

- **Source Instance Name** — specify the database deployment whose database backup you want to use.
- **Backup Tag** — a list of backups available for the specified database deployment. The latest backup is selected by default, but you can choose an earlier backup.
- **Administration Password** and **Confirm Password** — enter and then re-enter a password for the Oracle Database SYS and SYSTEM users in the newly replaced database.

5. Click **Replace Database** and confirm that you want to replace the database when prompted.

   The database deployment is put into Maintenance status and the operation begins. The process is fully automated and takes some time to complete. You should not access or manipulate the database deployment until the process is completed.

Replacing the Database by Using ibkup Actions

You can perform an instantiate-from-backup operation to replace the database on a Database Cloud Service database deployment by using `ibkup` actions with the `dbaasapi` utility.

The `dbaasapi` utility operates by reading a json file containing instructions and other information and writing its results to a json file specified in the input file. In essence, it is a command-line utility that operates like a REST API endpoint, accepting a json "request body" and producing a json "response body". The `dbaasapi` utility checks that the operation being requested does not conflict with any operation already in progress

and then runs the operation asynchronously: that is, it starts the requested operation and then returns terminal control to you.

> **Caution:**
>
> If the database you want to replace was created using ibkup, then you must first use `ibkup restore` to make the cloud database ready to accept the data from the new database. See .

Here are the tasks you perform to replace the database by using `ibkup` actions:

1. Copy the TDE wallet from the source database to the Database Cloud Service deployment, if necessary.
2. Create `dbaasapi` input files for `ibkup begin` and `ibkup status` actions.
3. Run the `ibkup begin` action.
4. Run the `ibkup status` action to monitor progress of the `ibkup` operation.
5. Upon completion of the `ibkup` operation, confirm that the source database now resides on the Database Cloud Service deployment.

Copy the Source Database TDE Wallet

If the cloud backup you are using was created using Transparent Data Encryption (TDE) or dual-mode encryption, you need to copy the TDE wallet from the source database to the database deployment.

> **Note:**
>
> If the source database is from another Database Cloud Service database deployment, its backup was created using Transparent Data Encryption (TDE) because all cloud backups from Database Cloud Service use TDE as the backup encryption mode.

Here are high-level instructions:

1. Zip (or tar) up the source database's `tde_wallet` directory. (If the source database is from another Database Cloud Service database deployment, its `tde_wallet` directory is `/u01/app/oracle/admin/dbname/tde_wallet`.)
2. On the database deployment you created, make a directory where you'll store the various files you'll create and use in the coming steps. For example:

   ```
   # mkdir -p /home/oracle/ibkup
   ```

3. Use a secure copy utility like scp or WinSCP to copy the zip file to this directory.
4. Unzip (or untar) the file into the `tde_wallet` subdirectory.

Create `dbaasapi` Input Files

1. Use a secure shell utility like ssh or PuTTY to connect as the `opc` user to the compute node that is associated with your target database deployment. For instructions, see Connecting to a Compute Node Through Secure Shell (SSH).
2. The `dbaasapi` utility must be run as the `root` user. Start a root-user shell:

   ```
   $ sudo -s
   #
   ```

3. Navigate to the directory where you previously stored the source database TDE wallet.

   ```
   # cd /home/oracle/ibkup
   ```

   If you did not copy the source database TDE wallet, create a directory to store your request and response files and then navigate to it.

4. Create a `begin-request.json` file to pass to `dbaasapi` to perform the `ibkup begin` action.

   Here is an example that uses a password-encrypted backup:

   ```
   # cat begin-request.json
   {
     "object": "db",
     "action": "begin",
     "operation": "ibkup",
     "params": {
       "dbname": "crmdb",
       "dbid": "1428538966",
       "oss_url": "https://mystore.storage.oraclecloud.com/v1/Storage-mystore/
   IBKUP",
       "oss_user": "storageadmin",
       "oss_passwd": "pa55word",
       "decrypt_key": "backup",
       "passwd": "Welcome-1",
       "dbsize": "30GB"
     },
     "outputfile": "/home/oracle/ibkup/begin-response.json",
     "FLAGS": ""
   }
   ```

   The json object for the `ibkup begin` action supports the following parameters. All parameters are required except those identified as optional.

Parameter	Description
`object`	The value `"db"`.
`action`	The value `"begin"`.
`operation`	The value `"ibkup"`.

Parameter	Description

`params`	An object containing parameters that provide details for the `ibkup begin` action. This object has the following parameters:
	• `dbname`: The name of the target database that you are replacing. You can get this value by querying the target database:
	`SQL> SELECT name FROM v$database;`
	• `dbid`: The database id of the source database. You can get this value by querying the source database:
	`SQL> SELECT dbid FROM v$database;`
	• `oss_url`: The URL of the container where the source database's backup is stored.
	• `oss_user`: The user name of an Oracle Cloud user who has read privileges for the container where the source database's backup is stored.
	• `oss_passwd`: The password of the `oss_user` user.
	• `rman_handle`: (Optional) The RMAN handle of a targeted backup that contains controlfile and spfile backups. The `ibkup begin` action will use the controlfile and spfile in this backup.
	Use the `rman_tag` parameter to specify the RMAN tag of a backup supported by this controlfile and spfile. If you do not specify an RMAN tag, the latest backup supported by this controlfile and spfile will be used.
	Oracle recommends that you provide both a handle and a tag to use a specific backup or provide neither a handle nor a tag to use the latest backup.
	You can view RMAN handles and tags by using the RMAN `LIST BACKUP` command.
	• `rman_tag`: (Optional) The RMAN tag of a targeted full backup. The `ibkup begin` action will use this backup.
	Use the `rman_handle` parameter to specify the RMAN handle of a backup containing controlfile and spfile backups that support this RMAN tag. If you do not specify an RMAN handle, the latest controlfile and spfile will be used. If they do not support the specified RMAN tag, a “datafile not found” error will occur.
	Oracle recommends that you provide both a handle and a tag to use a specific backup or provide neither a handle nor a tag to use the latest backup.
	You can view RMAN handles and tags by using the RMAN `LIST BACKUP` command.
	• `decrypt_key`: (Optional) The key (password) used to encrypt the backup.
	Provide this parameter if you created the backup using password encryption or dual-mode encryption.
	Note: you cannot use this option when replacing the database on a database deployment using Oracle Database 12c Release 2 (12.2) or later, because only backups using TDE are supported for such deployments.
	• `decrypt_wallet`: (Optional) The fully qualified path of the wallet directory you copied from the source database

Parameter	Description
	to the DBCS deployment you created; for example: `/home/oracle/ibkup/tde_wallet`. Provide this parameter if you created the backup using Transparent Data Encryption (TDE) or dual-mode encryption. • `passwd`: The administrator (SYS and SYSTEM) password to use for the target database after the replacement operation concludes. • `dbsize`: The size of the source database.
`outputfile`	The fully qualified name of the output file for `dbaasapi` to use; for example: `"/home/oracle/ibkup/begin-response.json"`.
`FLAGS`	The value `""` (an empty string).

5. Create a `status-request.json` file to pass to `dbaasapi` to perform the `ibkup status` action. Here is an example:

```
# vim status-request.json
{
  "object": "db",
  "action": "status",
  "operation": "ibkup",
  "id": "TBD",
  "params": {
    "dbname": "crmdb"
  },
  "outputfile": "/home/oracle/ibkup/status-response.json",
  "FLAGS": ""
}
```

In this example, the value of the `id` parameter is `"TBD"` because the `ibkup begin` action whose status this action will check has not been run yet.

The json object for the `ibkup status` action supports the following parameters. All parameters are required.

Parameter	Description
`object`	The value `"db"`.
`action`	The value `"status"`.
`operation`	The value `"ibkup"`.
`id`	The ID number of the action you want status for.
`params`	An object containing parameters that provide details for the `ibkup status` action. This object has the following parameters: • `dbname`: The name of the database on the target database that is being replaced.
`outputfile`	The fully qualified name of the output file for `dbaasapi` to use; for example: `"/home/oracle/ibkup/status-response.json"`.
`FLAGS`	The value `""` (an empty string).

Run the `ibkup begin` Action

1. Use `dbaasapi` to run the `ibkup begin` action:

```
# /var/opt/oracle/dbaasapi/dbaasapi -i begin-request.json
```

> **Note:**
>
> The `begin` action deletes the input json file because it contains sensitive password information. If you are testing and want to save the file for debugging purposes, make a copy of it before you run the `ibkup begin` action; for example:
>
> ```
> # cp begin-request.json begin-request.json.keep
> ```

2. View the output file to confirm that the action has started and note the `id` of the action; for example:

```
# cat /home/oracle/ibkup/begin-response.json
{
   "msg" : "",
   "object" : "db",
   "status" : "Starting",
   "errmsg" : "",
   "outputfile" : "",
   "action" : "begin",
   "id" : "19",
   "operation" : "ibkup",
   "logfile" : "/var/opt/oracle/log/crmdb/dbaasapi/db/ibkup/19.log"
}
```

The key parameters in this response are as follows:

Parameter	Description
`status`	The status of the operation; one of: `"Error"`, `"Starting"`, `"InProgress"` or `"Success"`.
`action`	The value `"begin"`, which is the `ibkup` action you requested.
`id`	The ID number assigned to this action. Use this number in subsequent `ibkup status` actions to check the status of the overall `ibkup` operation.
`operation`	The value `"ibkup"`, which is the operation you requested.
`logfile`	The log file for the ibkup operation. You can poll this log file to monitor progress of the operation. However, you should run the `ibkup status` action to monitor progress because this provides additional status information along with a definitive indication of when the operation is finished.

Run the `ibkup status` Action to Monitor Progress

1. Update the `status-request.json` input file with `id` value of the `ibkup` operation that you have started. Edit the `status-request.json` file, replacing the `id`

parameter value of "TBD" with the ID number reported in the begin-response.json file.

2. Use dbaasapi to run the ibkup status action and view the response; for example:

```
# /var/opt/oracle/dbaasapi/dbaasapi -i status-request.json
# cat status-response.json
{
   "msg" : "  -> 15 03 * * 6 oracle /var/opt/oracle/cleandb/cleandblogs.pl\\n\
\n#### Completed OCDE Successfully ####",
   "object" : "db",
   "status" : "Success",
   "errmsg" : "",
   "outputfile" : "",
   "action" : "begin",
   "id" : "19",
   "operation" : "ibkup",
   "logfile" : "/var/opt/oracle/log/crmdb/dbaasapi/db/ibkup/19.log"
}
```

3. Rerun the ibkup status action regularly until the response indicates that the operation is finished.

Confirm Successful Completion

1. Connect as the **oracle** user to the compute node that is associated with your target database deployment.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Connect as SYSDBA to the database:

```
$ sqlplus / as sysdba
```

3. Check the database status:

```
SQL> select status from v$instance;
STATUS
------------
OPEN
```

4. Check the name and id of the database; for example:

```
SQL> select name, dbid from v$database;

NAME        DBID
--------- ----------
CRMDB      4023093732
```

5. If the Oracle Database version is 12c or later, view information about its PDBs; for example:

```
SQL> select name,open_mode,restricted from v$pdbs;

NAME                  OPEN_MODE  RES
--------------------- ---------- ---
PDB$SEED              READ ONLY  NO
PDBDBNOASM            READ WRITE NO
```

6. Exit SQL*Plus:

```
SQL> exit
$
```

7. Check that services provided by Oracle Net Listener include those from the source database; for example:

```
$ lsnrctl status
Services Summary...
Service "crmdb.localdomain" has 1 instance(s).
  Instance "crmdb", status READY, has 1 handler(s) for this service...
Service "dbnoasm.localdomain" has 1 instance(s).
  Instance "crmdb", status READY, has 1 handler(s) for this service...
Service "dbnoasm.localdomainXDB" has 1 instance(s).
  Instance "crmdb", status READY, has 1 handler(s) for this service...
Service "pdbdbnoasm.localdomain" has 1 instance(s).
  Instance "crmdb", status READY, has 1 handler(s) for this service...
The command completed successfully
```

8. Disconnect from the compute node:

```
$ exit
```

Use ibkup restore

The `restore` action takes an input file of the same format as the `begin` action, except that the value of the `action` parameter must be `"restore"`.

If a `begin` operation fails, you can use the `restore` action to reset the database deployment's environment so that you can attempt the `begin` operation again, after determining the cause of the failure and correcting the problem.

If you successfully used ibkup to create a cloud database, but now want to replace the database from a different backup, you must use the `restore` action before using the `begin` action.

After you use the `restore` action, you need to reboot the database deployment's compute node to ensure that the environment is completely reset. For instructions, see Rebooting a Compute Node.

The `restore` action does as follows:

1. Validates the format and completeness of the input file.
2. Creates the output file, which includes an ID number for use in subsequent `status` actions.
3. Releases terminal control.
4. Terminates any `begin` action that is in progress.
5. Kills all processes related to the `begin` action. If it cannot kill one or more processes, it exits with an error status.
6. Restores the database deployment environment to its state before the first `begin` action.

The steps below show an example a `restore` action, which is used before retrying a failed `begin` action or before replacing an existing database created with ibkup using a different backup.

1. Create a JSON file for the `restore` action as shown below:

```
{
    "object": "db",
    "action": "restore",
```

```
    "operation": "ibkup",
    "params": {
        "dbname":
        "<target_dbname>"
    },
    "outputfile": "<response_file>",
    "FLAGS": ""
}
```

You can also add a `status` action to monitor the `restore` process.

2. Save the file as `restore.json`.
3. Run the ibkup action:

```
dbaasapi -i restore.json
```

4. After the restore completes, you must reboot the VMs before attempting the `begin` action.

More About `ibkup` Actions

The preceding instantiate-from-backup tasks showed the use of two `ibkup` actions; `begin` and `status`. Here is more information about what these two actions do, along with information about two other `ibkup` actions; `prereqs` and `restore`.

- The `begin` action:
  1. Validates the format and completeness of the input file.
  2. Creates the output file, which includes an ID number for use in subsequent `status` actions.
  3. Releases terminal control.
  4. Performs the same value-validation checks that the `prereqs` action performs.
  5. Takes a backup of the current database deployment environment, should the need arise to restore the environment after a failed `ibkup` operation.
  6. Replaces the current database using the backup of the source database.
- The `status` action:
  1. Validates the format and completeness of the input file.
  2. Retrieves the current status of operation whose ID number was provided in the input file.
  3. Creates the output file, which contains the retrieved status information.
- The `prereqs` action takes an input file of the same format as the `begin` and `restore` actions, except that the value of the `action` parameter must be `"prereqs"`.

  You can use the `prereqs` action to test whether the input file you intend to use for either the `begin` action or the `restore` action is valid and that the backup specified in the file is available.

  The `prereqs` action does as follows:

  1. Validates the format and completeness of the input file.

2. Creates the output file, which includes an ID number for use in subsequent `status` actions.
3. Releases terminal control.
4. Checks that the values provided in the input file would be valid if used in the input file for a `begin` or `restore` action. It confirms access to the backup, including use of the decryption key and wallet as necessary, and that the backup's database ID matches the provided `dbid`.

Creating a Clone Database Deployment from a Snapshot


This topic does not apply to Oracle Cloud Infrastructure.

You can create an Oracle Database Cloud Service database deployment from a snapshot you have taken of another database deployment in the same identity domain. The resulting deployment is known as a linked clone because its storage is linked to the snapshot's storage.

> **Note:**
>
> Currently, you can only create snapshots of database deployments that host a single-instance database. Therefore, you can only create linked-clone database deployments that host a single-instance database.

When you create a linked clone deployment, Database Cloud Service creates a new database deployment whose storage volumes are cloned from the snapshot.

Using the "copy on write" technology that Oracle Compute Cloud Service supports for storage volume snapshots, the file data on the linked-clone deployment can change without changing the snapshot itself. Thus, you can create several linked clones from the same snapshot to use for application testing or branched application development work.

Procedure

1. Open the Snapshots page of the deployment you want to create a snapshot of:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the name of the database deployment whose snapshot you want to use as the basis for a linked clone deployment.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Snapshots tab.

      The Oracle Database Cloud Service Snapshots page is displayed. Any snapshots already created for the deployment are shown in the Available Storage Snapshots list.

2. In the Available Storage Snapshots list, locate the snapshot you want to create a linked-clone deployment from and choose **Create Database Clone** from that snapshot's

menu.

   The Subscription Type page of the Create Instance wizard is displayed.

3. Step through the pages of the wizard to provide information about the linked-clone deployment.

   As you step through, you will note that several options are not selectable; for example, Software Release and Software Edition. Such options are not selectable because their values are determined from the snapshot upon which the linked-clone is based.

   Also note that some options are required: you must provide a new service name, specify an SSH public key, and provide an administrator password. You can change the other selectable from their defaults if you want to; for example, Shape and Backup Destination.

4. After completing the wizard by clicking **Create** on the Confirmation page, the Services page is displayed, including notice that creation of the database deployment has begun.

Creating a Hybrid DR Deployment


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can create a Hybrid Disaster Recovery (DR) Oracle Database Cloud Service database deployment with a primary database on your own host system and a standby database in the cloud.

You can use the Oracle Database Cloud Service creation wizard to create the Hybrid DR deployment which uses Oracle Data Guard.

See Using Oracle Data Guard in Database Cloud Service for general information about using Oracle Data Guard in Oracle Database Cloud Service.

The on-premises database on your own system will be the primary database in the Hybrid DR deployment, which uses Oracle Data Guard. Your system and the primary database must meet certain requirements as follows:

- The owner of the Oracle Database software (dbowner) must be the `oracle` user.
- The database must be at version 11.2.0.4.0 or version 12.1.0.2.0 with the latest PSU (patch set update) applied.
- The database must use a file system as its storage method for database files. Oracle Automatic Storage Management (ASM) is not supported.
- The database cannot be an Oracle Real Application Clusters (RAC) database.
- The database must be smaller than 2 TB (1.2 TB if you plan to back up the database to both cloud and local storage when the database on the compute node is in the primary role).

- If it is an Oracle 12*c* database, it must be a multitenant container database (CDB). Database Cloud Service does not support non-CDB Oracle 12*c* databases.
- All pluggable databases (PDBs) in the Oracle 12*c* multitenant environment must be open in read/write mode.
- The database must be in `ARCHIVELOG` mode. See “Changing the Database Archiving Mode” in *Oracle Database Administrator’s Guide* for Release 12.1 or Release 11.2.
- The tablespaces of the database must be encrypted using Transparent Data Encryption (TDE), a feature of Oracle Advanced Security. See "Configuring Transparent Data Encryption" in *Oracle Database Advanced Security Guide* for Release 12.1 or "Securing Stored Data Using Transparent Data Encryption" in *Oracle Database Advanced Security Administrator's Guide* for Release 11.2
- The database on your own system cannot be part of another Oracle Data Guard configuration.
- Enable Oracle Flashback Database for easy reinstatement of the primary database after a failover operation. See “Using Flashback Database” in *Oracle Database Backup and Recovery User’s Guide* for Release 12.1 or Release 11.2 .
- Enable `FORCE LOGGING` mode to prevent the loss of data when statements in `NOLOGGING` mode are executed on the primary database. See “Specifying FORCE LOGGING Mode” in *Oracle Database Administrator’s Guide* for Release 12.1 or Release 11.2.
- Ensure the `listener.ora` file is configured for static service registration, a requirement of Oracle Data Guard.
- To protect plaintext from being visible on the WAN, enable Oracle Net encryption. Set parameters in the `$ORACLE_HOME/network/admin/sqlnet.ora` file:
  - `SQLNET.ENCRYPTION_SERVER = requested`
  - `SQLNET.ENCRYPTION_TYPES_SERVER = (RC4_256, AES256)`
  - `SQLNET.ENCRYPTION_CLIENT = requested`
  - `SQLNET.ENCRYPTION_TYPES_CLIENT = (RC4_256, AES256)`

  After editing the `sqlnet.ora` file, reload the listener.
- Ensure one of the following RPMs is installed, based on the Oracle Database release of your system:
  - Oracle Database 11*g* Release 2: `oracle-rdbms-server-11gR2-preinstall`
  - Oracle Database 12*c* Release 1: `oracle-rdbms-server-12cR1-preinstall`

  You can use this command as the `root` user to verify that the RPM is installed:

  ```
  # rpm -qa|grep oracle-rdbms-server
  ```
- Ensure the Netcat RPM is installed: nc-1.84-24.el6.x86_64 or higher.
- Ensure that the `/etc/hosts` file contains the IP address for your on-premises system, the host name with a fully qualified domain name, and a short host name.
- Ensure all TCP socket size maximum kernel parameters are set to `10 MB` (`10485760`) for optimal transport performance:
  - `net.core.rmem_max = 10485760`
  - `net.core.wmem_max = 10485760`

As the `root` user, execute these commands:

```
# sysctl -w net.core.rmem_max=10485760
# sysctl -w net.core.wmem_max=10485760
```

- The Oracle listener port and the Secure Shell (SSH) port must be open for remote access from the compute node.

  It is important that you secure port connectivity on the on-premises system. To enable SSH tunneling and also to make sure that only specific cloud IP addresses can access the listener port in the on-premises system, appropriate security rules need to be configured on the on-premises system. The on-premises firewall needs to have properly configured Access Control Lists to allow SSH and Oracle Net to be accessed from the on-premises system by the Oracle Cloud compute node. Because Oracle Data Guard in a Hybrid DR deployment requires access to the on-premises database from the cloud compute node, the primary database listener port must be opened with restricted access from the cloud IP address.

- Ensure that the non-Oracle software meets the following minimum release requirements

  – Java: Release 1.7 or higher. Java must be in the default path.

  – Perl: 5.10.1 or higher.

  – Python: 2.6.6 or higher.

You must have a valid Oracle Storage Cloud Service account and ensure that it is accessible from your on-premises system. You will create a container and then specify its name in the `setupdg.cfg` file and in the Oracle Database Cloud Service creation wizard.

Perform these steps to create the Hybrid DR configuration:

1. Create a container in Oracle Storage Cloud Service to be used only during the deployment creation. See Creating Containers in *Using Oracle Cloud Infrastructure Object Storage Classic*. You should delete this container after the database deployment is created.
2. Create an IP reservation that will be used for the standby database. See Creating an IP Reservation for detailed instructions.
3. If you have previously used the same on-premises system to set up a Hybrid DR configuration, perform these commands on the system to prepare it for reuse in this Hybrid DR configuration:

   ```
   % cd /var/opt/oracle/hdg/
   % rm -rf db_wallet
   % rm -rf hdgonpreminfo*
   ```

4. Using `wget`, download the `OracleCloud_HybridDR_Setup.zip` file from Oracle Storage Cloud Service to your own system and expand the zip file.

   ```
   % wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/hdg/
   DBCS_version/OracleCloud_HybridDR_Setup.zip
   ```

   where `DBCS_version` is the Database Cloud Service version, such as 17.3.1.

5. Follow the instructions in the `README` file from the `OracleCloud_HybridDR_Setup.zip` file to update the `setupdg.cfg` configuration file and back up your on-premises database.

6. Create the Hybrid DR deployment by using the Oracle Database Cloud Service creation wizard.

   See Creating a Customized Database Deployment for instructions on using the wizard.
   Be sure to specify the following values when using the wizard:

   - Service page:
     - **Region.** Select a region. Your identity domain must be enabled for regions to create a Hybrid DR deployment.
     - **IP Network**. If your region has IP networks, this field appears. You must select No Preference because Hybrid DR does not support IP networks.
     - **Software Release**. Choose the Oracle Database release corresponding to the primary database.
     - **Software Edition**. Choose at the minimum a software edition that supports the options used by the source database.
     - **Database Type**. Choose Data Guard Standby with Hybrid DR.
   - Service Details page:
     - **DB Name (SID)** (on the Service Details page). Specify the same SID as used by the database on your on-premises system. You must make sure the SID you provide matches exactly, including the case of letters.
     - **Administration Password**. Specify the same password that you use for the SYS user on your on-premises database.
     - **Usable Database Storage (GB)** (on the Service Details page). Specify the minimum the amount of storage needed to accommodate the source database.
     - **Advanced Settings: Listener Port**. Specify the same port number that you use for the database on your own system.
     - **Advanced Settings: IP Reservations**. Select the IP reservation you created in a previous step.
     - **Backup Destination.** The chosen destination is only applicable when the database on the compute node is the primary database. If you select None when you create the deployment, you can change the backup configuration as described in Enabling and Reconfiguring the Automatic Backups Feature. Be aware that if you change the backup destination, the Database Cloud Service Console is unaware of this change.
     - **Hybrid DR: Cloud Storage Container**. The name of the Oracle Storage Cloud Service container that contains the configuration information and backup of your on-premises database. This name should match the name you set in the `setupdg.cfg` file.
     - **Hybrid DR: Username**. The name of a user who has read/write access to the specified Oracle Storage Cloud Service container.
     - **Hybrid DR: Password**. The password of the user specified in **Username**.
7. After the database deployment creation is complete, change the permissions on the `/home/oracle/bkup` directory.
   a. Log in to the compute node as the **opc** user.
   b. Start a root-user command shell:

```
$ sudo -s
#
```

c. Change the permissions on the `/home/oracle/bkup` directory:

```
# chmod 755 /home/oracle/bkup
```

d. Exit the root-user command shell.

8. After the database deployment creation is complete, create operating system directories required for the standby database on the compute node.

a. Log in to the compute node as the `oracle` user.

b. Connect to the standby database as `SYSDBA`:

```
$ sqlplus / as sysdba
```

c. Query `V$DATAFILE` to see a list of files and identify the directory for each, as shown in the example:

```
SQL> SELECT name FROM v$datafile;
NAME
----------------------------------------------------------------------------
-
/u02/app/oracle/oradata/orcl/system01.dbf
/u02/app/oracle/oradata/orcl/sysaux01.dbf
/u02/app/oracle/oradata/orcl/undotbs01.dbf
/u02/app/oracle/oradata/orcl/pdbseed/system01.dbf
/u02/app/oracle/oradata/orcl/users01.dbf
/u02/app/oracle/oradata/orcl/pdbseed/sysaux01.dbf
/u02/app/oracle/oradata/orcl/PDB1/system01.dbf
/u02/app/oracle/oradata/orcl/PDB1/sysaux01.dbf
.
8 rows selected.
```

d. Exit from SQL*Plus.

e. Create the directories that you identified when you queried `V$DATAFILE`. For example:

```
$ mkdir -p /u02/app/oracle/oradata/orcl
$ mkdir -p /u02/app/oracle/oradata/orcl/pdbseed
$ mkdir -p /u02/app/oracle/oradata/orcl/PDB1
```

9. Make a copy of `/home/oracle/.ssh/` on your own system after the database backup is completed successfully.

When the Hybrid DR deployment is created, a new set of private and public keys is created in `/home/oracle/.ssh`. The original `.ssh` directory is stored as `/home/oracle/.ssh.bak`. Only the private and public keys generated during the deployment creation process are copied to the compute node. By making an extra copy of `/home/oracle/.ssh`, you can append the public keys from your on-premises host to the newly created `/home/oracle/.ssh` directory.

10. Delete the Oracle Storage Cloud Service container that you created in the first step.

The following limitations apply to a Hybrid DR deployment:

- You must use the `dataguard` subcommands of the `dbaascli` utility to perform operations on your Hybrid DR Data Guard configuration. You cannot use the Oracle Database Cloud Service console to perform operations on the on-premises primary database. See The dbaascli Utility for detail on the `dataguard` subcommands of the `dbaascli` utility.

- You cannot use the Oracle Database Cloud Service console or the `bkup_api` utility to take backups of your on-premises primary database. If the standby database on the compute node becomes the primary database in the Hybrid DR configuration, backups may be taken by using the Oracle Database Cloud Service console or the `bkup_api` utility depending on the backup configuration selected when the database deployment was created.
- You cannot use the Oracle Database Cloud Service console or the `patch` subcommand of the `dbaascli` utility to patch your on-premises primary database. See Patching a Hybrid DR Deployment for additional information on patching the databases in a Hybrid DR deployment.

Viewing All Database Deployments

From the Oracle Database Cloud Service Console, you can:

- View the total resources allocated across all Oracle Database Cloud Service database deployments.
- View the details for each deployment.
- Use the search field to filter the list to include only the deployments that contain a given string in their name.

To view all database deployments:

1. Open the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   The Oracle Database Cloud Service console opens and displays the Instances Page, which contains a list of database deployments.

> **Note:**
>
> If a Welcome page is displayed, click **Services** next to Database Cloud Service to display the Instances Page.

Viewing Detailed Information for a Database Deployment

From the Oracle Database Cloud Service Overview page, you can:

- View a summary of details for a database deployment on Oracle Database Cloud Service, such as description, subscription mode, and so on.
- View the total resources allocated to the deployment.
- View the details and status information for each node associated with the deployment.

To view detailed information for a database deployment:

1. Open the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. Click on the name of the database deployment for which you want to view more information.

   The Oracle Database Cloud Service Overview Page is displayed.

Viewing Activities for Database Deployments in an Identity Domain

Use the Activity page to view activities for database deployments on Oracle Database Cloud Service in your identity domain. You can restrict the list of activities displayed using search filters.

To view activities for your database deployments:

1. Open the Activity page:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click **Activity**.

      The Activity Page is displayed, showing the list of all activities started within the past 24 hours. You can use the Start Time Range field to specify a start time range other than the default of the previous 24 hours.
2. Use the options in the Search Activity Log section to filter the results to meet your needs. You can search on start time range, full or partial service name, activity status, and operation type. Click **Search**. View the results in the table that follows.

Stopping, Starting and Restarting a Database Deployment

From the Oracle Database Cloud Service console, you can stop, start and restart the compute nodes that are associated with a database deployment on Oracle Database Cloud Service.

Topics

- About Stopping, Starting and Restarting a Database Deployment
- Stopping a Database Deployment
- Starting a Stopped Database Deployment
- Restarting a Database Deployment
- Viewing Past Stop, Start and Restart Activity

About Stopping, Starting and Restarting a Database Deployment

About Stopping a Database Deployment

When you stop a Database Cloud Service database deployment, no access to it is possible and you can perform no management operations on it except to start it or to delete it.

Stopping a database deployment is similar to turning off your personal computer: it has no computing capabilities because the CPU and RAM have no power, but all its other

resources—disk drives and the data they contain, static IP reservations, and so on—remain and are ready to be put back into use when power is restored.

When database deployment is stopped, its CPU and RAM (an Oracle Compute Cloud Service instance) are stopped. As a consequence, it consumes no OCPU or memory resources and so metering and billing of these resources stop. However, all the other resources of the database deployment continue to exist and so continue to be metered and billed, including:

- Oracle Compute Cloud Service resources such as storage volumes and IP address reservations
- Oracle Storage Cloud Service storage space used by the database deployment's backups to the Oracle Cloud (if the database deployment was being backed up to cloud storage)

Additionally, when database deployment is stopped, backups of it are not performed.

About Starting a Stopped Database Deployment

When you start a stopped Database Cloud Service database deployment, access to it becomes possible again and you can perform management operations on it such as scaling and patching.

Starting a stopped database deployment is similar to turning your personal computer back on: its computing capabilities are restored because the CPU and RAM again have power, and all its other resources are put back into use.

When database deployment is started:

1. An Oracle Compute Cloud Service instance of the appropriate compute shape (OCPU and memory) is allocated to it.
2. All other Compute Cloud Service resources associated with it when it was created or as the result of a scaling operation are reattached to it.
3. The allocated Oracle Compute Cloud Service instance is started.

After these steps complete, the database deployment is running and available.

Because the started database deployment again consumes OCPU and memory resources, metering and billing of these resources resume.

> **Note:**
>
> Compute Cloud Service resources that were associated with the database deployment using the Oracle Compute Cloud Service console are ***not*** reattached when it is started. As a result, you must manually reattach the following kinds of Compute Cloud Service resources:
>
> - Storage volumes you created and attached using the Oracle Compute Cloud Service console.
>
> You must attach such storage volumes to the new Oracle Compute Cloud Service instance once the database deployment is started, and then connect to the compute node and mount them.
> - Security lists to which you added the database deployment's previous Oracle Compute Cloud Service instance.
>
> You must add such security lists to the new Oracle Compute Cloud Service instance once the database deployment is started.
>
> To manage custom network access to your service instance, you can use manually created security rules that refer to the database deployment's default security list instead of using manually created and added security lists. If you do so, you can avoid the need to add the manually created security lists to the new Oracle Compute Cloud Service instance once the database deployment is started. For more information, see Enabling or Restricting Port Access by Creating an Access Rule.

About Restarting a Database Deployment

When you restart a Database Cloud Service database deployment, it is stopped and then immediately started again. Thus, the information about what happens when stopping and starting a database deployment applies to restarting a database deployment as well, just in immediate succession.

> **Note:**
>
> Restarting a database deployment is different from rebooting a compute node of a database deployment. Rebooting a compute node, as described in Rebooting a Compute Node, does not restart the database deployment. It simply reboots the compute node.

Stopping a Database Deployment

In general, you stop a Database Cloud Service database deployment for one of these reasons:

- To prohibit access to it.
- To reduce its cost of operation.

Before You Begin

To learn what happens when you stop a Database Cloud Service database deployment, review About Stopping, Starting and Restarting a Database Deployment.

Procedure

To stop a database deployment:

1. View the Overview page for the database deployment:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment you want to stop.

      The Oracle Database Cloud Service Overview page is displayed.
2. From the menu for the database deployment's compute node, select **Stop**, and then confirm the action.

   The deployment first has a status of **Maintenance** and then **Stopped** in the Oracle Database Cloud Service console. Note that you cannot scale a stopped deployment.

Starting a Stopped Database Deployment

Before You Begin

To learn what happens when you start a stopped Database Cloud Service database deployment, review About Stopping, Starting and Restarting a Database Deployment.

Procedure

To start a stopped database deployment:

1. View the Overview page for the database deployment:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment you want to start.

      The Oracle Database Cloud Service Overview page is displayed.
2. From the menu for the database deployment's compute node, select **Start**, and then confirm the action.

   The deployment has a status of **Maintenance** in the Oracle Database Cloud Service console until it is fully started.

Restarting a Database Deployment

> **Note:**
>
> Restarting a database deployment is different from rebooting a compute node of a database deployment. Rebooting a compute node, as described in Rebooting a Compute Node, does not restart the database deployment. It simply reboots the compute node.

Before You Begin

To learn what happens when you restart a Database Cloud Service database deployment, review About Stopping, Starting and Restarting a Database Deployment.

Procedure

To restart a database deployment:

1. View the Overview page for the database deployment:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment you want to restart.

      The Oracle Database Cloud Service Overview page is displayed.
2. From the  menu for the database deployment's compute node, select **Restart**, and then confirm the action.

   The deployment has a status of **Maintenance** in the Oracle Database Cloud Service console until it is fully restarted.

Viewing Past Stop, Start and Restart Activity

You can see information about past stop, start and restart activity for a Database Cloud Service database deployment by viewing the activity log:

1. View the Overview page for the database deployment:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment whose past activity you want to view.

      The Oracle Database Cloud Service Overview page is displayed.
2. Click the triangle icon beside the Activity title to expand the activity log.

   The activity log shows information about past operations performed on the database deployment, with the most recent activity first.
3. Click the triangle icon beside an operation to see details about that operation.

   If an operation failed, the details include information about why it failed.

Rebooting a Compute Node

On occasion, you might find it necessary to reboot a compute node associated with Oracle Database Cloud Service. Follow these steps to perform the operation.

> **Note:**
>
> Rebooting a compute node is different from restarting a compute node. Restarting a compute node, as described in Restarting a Database Deployment, stops and removes the Compute Cloud Service instance on which the compute node is running and then creates and starts a new Compute Cloud Service instance for the compute node. Rebooting a compute node uses a Linux command to restart the Compute Cloud Service instance on which the compute node is running.

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Enter the command to reboot the compute node:

   ```
   # reboot
   ```

   > **Caution:**
   >
   > Do not use the `halt`, `shutdown` or `shutdown –h` commands to shut down the compute node. Doing so will stop the compute node indefinitely and will require manual intervention by Oracle Cloud system administrators to restart the compute node.

   Your connection to the compute node is closed and the compute node reboots.

Scaling a Database Deployment

If a database deployment on Oracle Database Cloud Service is performing poorly or is running out of storage, you can scale up the environment supporting the database deployment.

Usually, the need to scale arises as the result of analyzing database performance, as described in Tuning Oracle Database Performance on Database Cloud Service.

Occasionally, the need to scale arises from some change made to the database or backup configuration after it was created. For example, if the decision to use the In-Memory Database option was made after database creation, you might need to scale up the compute shape.

Scaling the Compute Shape for a Database Deployment

This topic does not apply to Oracle Cloud Infrastructure.

> **Note:**
>
> When you scale the compute shape of a database deployment on Database Cloud Service, the deployment is put into Maintenance status during the operation and it is restarted. As a result of the restarting, any resources you've manually added using the Compute Classic console become detached from the database deployment. For more information and for instructions on reattaching such resources, see About Stopping, Starting and Restarting a Database Deployment.

To scale the compute shape for a database deployment:

1. View the Overview page for the database deployment:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the name of the deployment you want to scale.

      The Oracle Database Cloud Service Overview page is displayed.
2. Choose the scaling command:
   - For database deployments that use Oracle RAC, click the next to the deployment name and choose **Scale Up/Down**.
   - For other deployments, click the menu in the box for the compute node and choose **Scale Up/Down**.

   The Scale Up/Down Service overlay is displayed. Note that the overlay includes information about the current compute shape.
3. Select a new compute shape.
4. Click **Yes, Scale Up/Down Service** to scale the database deployment.

   The scaling operation begins. The database deployment is in Maintenance status and unavailable while the scaling operation is in progress.

Scaling Up the Storage for a Database Deployment

When you scale up the storage for a database deployment, a storage volume is created and attached to the deployment.

This storage volume remains attached and available to the deployment even after it is restarted or is stopped and then started. Also, the storage volume exists until you delete the database deployment, at which time the storage volume is also deleted.

Scale-up limits and constraints differ depending on the infrastructure underlying the deployment:

- **Oracle Cloud Infrastructure** supports 32 block storage volumes attached to a compute node, of which 4 are used when the database deployment is created. Thus, you have 28 opportunities to scale up storage.

In each scale-up operation, you can create a storage volume of 50 GB to 16384 GB (16 TB) in 50 GB increments. The deployment is put into Maintenance status during the operation.

- **Oracle Cloud Infrastructure Classic** supports 10 block storage volumes attached to a compute node, of which 5 are used when the database deployment is created. Thus, you have only 5 opportunities to scale up storage. Consequently, each scale-up operation you perform can dramatically affect the maximum size your database can grow to.

  In each scale-up operation, you can create a storage volume of 1 GB to 2048 GB in 1 GB increments. The deployment is put into Maintenance status and restarted during the operation. As a result of the restarting, any resources you've manually added to the database deployment by using the Compute Classic console become detached from the deployment. For more information and for instructions on reattaching such resources, see About Stopping, Starting and Restarting a Database Deployment.

To scale up the storage for a database deployment:

1. View the Overview page for the database deployment:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the name of the database deployment you want to scale.

      The Oracle Database Cloud Service Overview page is displayed.
2. Choose the scaling command:
   - For database deployments in Oracle Cloud Infrastructure, click the  next to the deployment name and choose **Add Storage**.
   - For database deployments in Oracle Cloud Infrastructure Classic that use Oracle RAC, click the  next to the deployment name and choose **Scale Up/Down**.
   - For other deployments in Oracle Cloud Infrastructure Classic, click the  menu in the box for the compute node and choose **Scale Up/Down**.

   The Scale Up/Down Service overlay is displayed.
3. In the **Additional Storage (GB)** box, enter an amount raw storage to add to the database deployment.

   Note that up to 8% of this raw storage will be used for file system constructs and other overhead.

   Also note that, when adding storage to a database deployment that uses Oracle RAC, you should specify the same size as the other storage volume or volumes already in the Oracle ASM disk group you want to scale up: Data or Backup.
4. Specify how the additional storage should be allocated in the **Add Storage to** list:
   - **Create New Storage Volume**: adds a new storage volume to the database deployment and mounts it as the next available `/u0`*n* mount point. This option is not available for deployments that use Oracle RAC.

- **Extend Data Storage Volume**: adds the storage volume to the existing Linux LVM disk group (or Oracle ASM disk group on deployments that use Oracle RAC) for database data storage.
- **Extend Backup Storage Volume**: adds the storage volume to the existing Linux LVM disk group (or Oracle ASM disk group on deployments that use Oracle RAC) for backup and FRA storage.

5. Click **Yes, Scale Up/Down Service** to scale the database deployment.

   The scaling operation begins. The deployment is in Maintenance status while the scaling operation is in progress.

Creating and Managing IP Reservations

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

If your identity domain is enabled for regions, you can reserve IP addresses by using the IP Reservations tab on Oracle Database Cloud Service console. After you have created an IP reservation, you can use it when creating a Database Cloud Service database deployment, in which case Database Cloud Service does not create and assign an IP address for you.

> **Note:**
>
> Currently, IP reservations are not supported for database deployments that use Oracle Data Guard.

Here are the tasks for creating and managing IP reservations:

- Creating an IP Reservation
- Using an IP Reservation when Creating a Database Deployment
- Deleting an IP Reservation

Creating an IP Reservation

If your identity domain is enabled for regions, you can create IP reservations for later use when creating Database Cloud Service database deployments.

If there are no Database Cloud Service IP Reservations in your identity domain, the **IP Reservations** link does not appear in the Oracle Database Cloud Service console. In this case, see Creating the First IP Reservation to create an IP reservation. Otherwise, use the following instructions.

1. Go to the IP Reservations Page.
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click **IP Reservations**.

2. Click **Create**.

   The Create IP Reservation window displays.

3. Enter a name for the IP reservation and choose the region where the reservation will be available from.

4. If you intend to use this reservation for a database deployment that you attach to an IP network, select the **On IP Network** check box.

   If you leave this check box deselected, the IP reservation can be assigned to only a database deployment that you attach to the shared network.

5. Click **OK**.

Creating the First IP Reservation

To create the first IP Reservation, use the Create Instance wizard.

1. Start the Create Instance wizard.

   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click **Create Instance**.

2. Enter the text “temp” in the Instance Name field, select a region in the Region list, and click **Next**.

3. In the Database Configuration section, expand Advanced Settings.

4. Click the gear icon beside the IP Reservations field.

   The Create Instance wizard closes and the IP Reservations page displays.

5. Click **Create**.

   The Create IP Reservation window displays.

6. Enter a name for the IP reservation and choose the region where the reservation will be available from.

7. If you intend to use this reservation for a database deployment that you attach to an IP network, select the **On IP Network** check box.

   If you leave this check box deselected, the IP reservation can be assigned to only a database deployment that you attach to the shared network.

8. Click **OK**.

Using an IP Reservation when Creating a Database Deployment

Normally, Database Cloud Service automatically assigns new IP addresses to database deployments you create. If your identity domain is enabled for regions, you can direct it to use specific IP Reservations instead and so gain control over the IP addresses of your database deployments.

After creating an IP reservation, as described in Creating an IP Reservation, use that reservation when creating a database deployment by making the following choices in the Create Instance wizard:

1. On the Service page, choose a region other than **No Preference**.

2. On the Service Details page, expand Advanced Settings in the Database Configuration section. Then choose one IP reservation for a Single Instance configuration or two IP reservations for a RAC configuration from the IP Reservations list.

   If your deployment uses Data Guard, choose one IP reservation for a Single Instance configuration or two IP reservations for a RAC configuration from the Standby list.

Deleting an IP Reservation

When you no longer require an unused IP reservation, you can delete it.

1. Go to the IP Reservations Page.
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click **IP Reservations**.
2. Locate the unused IP reservation you want to delete and then click the **X** icon in the row for that reservation.

   You are prompted to confirm the deletion.
3. Click **OK** to confirm deletion of the IP reservation.

Creating and Managing Snapshots of a Database Deployment


This topic does not apply to Oracle Cloud Infrastructure.

On database deployments hosting a single-instance database, Oracle Database Cloud Service supports the creation of storage snapshots, which you can then use to create new database deployments called linked clones.

When you create a storage snapshot, the database deployment is put into maintenance status and a snapshot of all the storage volumes for the deployment is taken. Then, when you create a linked clone deployment, Database Cloud Service creates a new database deployment whose storage volumes are from the snapshot.

Using the “copy on write” technology that Oracle Compute Cloud Service supports for storage volume snapshots, the file data on the linked clone deployment can change without changing the snapshot itself. Thus, you can create several linked clones from the same snapshot to use for application testing or branched application development work.

Here are the tasks for creating and managing snapshots:

- Creating a Snapshot
- Creating a Clone Database Deployment from a Snapshot
- Listing Linked Clone Database Deployments Created from a Snapshot
- Deleting a Snapshot

Creating a Snapshot

1. Go to the Snapshots page of the deployment you want to create a snapshot of:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the name of the deployment you want to create a snapshot of.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Snapshots tab.

      The Oracle Database Cloud Service Snapshots page is displayed. Any snapshots already created for the deployment are shown in the Available Storage Snapshots list.
2. Click **Create Storage Snapshot**. In the Create Storage Snapshot window that is displayed, enter a name (and, optionally, a description) for the snapshot and then click **Create**.

   In the next window that is displayed, confirm that you want to put the database deployment into maintenance mode and create the snapshot by clicking **Create**.

Listing Linked Clone Database Deployments Created from a Snapshot

1. Go to the Snapshots page of the deployment whose snapshot you want to see linked clone deployments of:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the name of the deployment whose snapshot you want to see linked clone deployments of.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Snapshots tab.

      The Oracle Database Cloud Service Snapshots page is displayed.
2. In the Available Storage Snapshots list, locate the snapshot you're interested in and check the value displayed next to Linked Clones.

   If the value is zero, no linked clone deployments have been created from the snapshot. Otherwise, click

   before Linked Clones to view the list of linked clone deployments.

   When viewing the list of linked clone deployments, you can click the name of a deployment to go directly to that deployment's Overview page.

Deleting a Snapshot

> **Note:**
>
> You cannot delete a snapshot that has linked clone database deployments created from it. You must first delete the linked clone deployments, as described in Deleting a Database Deployment.

1. Go to the Snapshots page of the deployment whose snapshot you want to delete:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.
   b. Click the name of the deployment whose snapshot you want to delete.

      The Oracle Database Cloud Service Overview page is displayed.
   c. Click the Administration tile and then click the Snapshots tab.

      The Oracle Database Cloud Service Snapshots page is displayed.
2. In the Available Storage Snapshots list, locate the snapshot you want to delete and choose **Delete** from that snapshot's
   
   menu.
3. In the Delete Storage Snapshot window, confirm that you want to delete the snapshot by clicking **Delete**.

   If the window warns you that you cannot delete the snapshot because there are database deployments cloned from it, click **Close** and then delete the linked clone deployments before trying to delete the snapshot.

Deleting a Database Deployment

When you no longer require a database deployment on Oracle Database Cloud Service, you can delete it.

> **Note:**
>
> You cannot delete a database deployment that has any linked-clone deployments created from any of its snapshots. Before you can delete the deployment you must first delete all linked-clone deployments derived from it.

To delete a database deployment:

1. Open the Instances page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. Select **Delete** from the action menu ( ) corresponding to the database deployment that you want to delete.

   You are prompted to confirm the deletion.

3. Use the confirmation dialog to confirm that you want to delete the database deployment. Optionally, you can also select the option to delete the backups associated with the database deployment.

> **Note:**
>
> The option to delete the backups associated with the database deployment only exists for deployments that are created using Database Cloud Service release 17.1.5, or later.

Once deleted, the entry is removed from the list of database deployments displayed on the Oracle Database Cloud Service console.

Tracking the Number of Database Deployments in an Account

 This topic does not apply to Oracle Cloud at Customer.

You can track the number of database deployments on Oracle Database Cloud Service across all identity domains using the My Account Dashboard page. Note that the My Account Dashboard page is different from the My Services Dashboard page: My Account Dashboard shows information for your entire account, while My Services Dashboard shows information limited to one identity domain in your account.

To open the My Account Dashboard page, sign in to My Account. By default, the Dashboard page is in focus. You can also click **Dashboard** at any time to display the page.

The list of services is displayed using the following naming convention: *`service-name (service-type)`*. Click the service name to go to the details page, which displays status history, availability history, usage metrics, and additional information for the selected service.

For information about the details provided on the Dashboard page, see Exploring the My Account Dashboard Page in *Getting Started with Oracle Cloud*.

3

Managing Network Access to Database Cloud Service

When you create an Oracle Database Cloud Service database deployment in Oracle Cloud Infrastructure, network access to the deployment is provided and managed by the Oracle Cloud Infrastructure network components according to rules specified when you followed the instructions in Prerequisites for Oracle Platform Services on Oracle Cloud Infrastructure in the Oracle Cloud Infrastructure documentation.

When you create an Oracle Database Cloud Service database deployment in Oracle Cloud Infrastructure Classic, network access to the deployment is provided and managed using access rules in the Oracle Database Cloud Service console. With the exception of SSH access on port 22, the access rules created for a database deployment are disabled to ensure the deployment is secure by default. To open access to applications and management tools, you need to enable one of these access rules or create a custom access rule of your own.

Topics

About Network Access to Database Cloud Service

Network access to the compute nodes associated with Oracle Database Cloud Service is primarily provided by Secure Shell (SSH) connections on port 22. Other network protocols and services may also be used, but may require additional configuration.

SSH Access on Port 22

SSH is a cryptographic network protocol that uses two keys, one public and one private, to provide secure communication between two networked computers. Port 22 is the standard TCP/IP port that is assigned to SSH servers, and on Database Cloud Service instances this port is accessible to external clients by default.

When creating a database deployment, you specify the public key on the Service page of the Create Instance wizard by:

- Uploading a public key file.

  You can specify a file on your local system that contains the public key value.

- Pasting the public key value.

  You can paste the public key value into a box provided by the wizard. If you do so, make sure the value does not contain line breaks or end with a line break.

- Having the wizard generate a key pair for you.

  You can have the wizard create an SSH key pair and use the generated public key value. You download a zip file containing both the public key file and the private key file.

When you access a compute node using SSH, you must provide the private key that matches the public key specified when the database deployment was created.

To generate the SSH public/private key pairs needed to access Database Cloud Service, see Generating a Secure Shell (SSH) Public/Private Key Pair.

Access to Other Ports


This topic does not apply to Oracle Cloud Infrastructure.

Additional configuration may be required to access network protocols and services on a compute node by using a port other than port 22. You may:

- Enable network access to the port

  You can use the Oracle Database Cloud Service console to enable access to a port on a compute node. See Enabling Access to a Compute Node Port

- Create an SSH tunnel to the port

  Creating an SSH tunnel enables you to access a specific compute node port by using an SSH connection as the transport mechanism. To create the tunnel, you must have the SSH private key file that matches the public key specified during the database deployment creation process. See Creating an SSH Tunnel to a Compute Node Port.

Generating a Secure Shell (SSH) Public/Private Key Pair

Several tools exist to generate SSH public/private key pairs. The following sections show how to generate an SSH key pair on UNIX, UNIX-like and Windows platforms.

Generating an SSH Key Pair on UNIX and UNIX-Like Platforms Using the ssh-keygen Utility

UNIX and UNIX-like platforms (including Solaris and Linux) include the ssh-keygen utility to generate SSH key pairs.

To generate an SSH key pair on UNIX and UNIX-like platforms using the ssh-keygen utility:

1. Navigate to your home directory:

   ```
   $ cd $HOME
   ```

2. Run the ssh-keygen utility, providing as ***filename*** your choice of file name for the private key:

   ```
   $ ssh-keygen -b 2048 -t rsa -f filename
   ```

The ssh-keygen utility prompts you for a passphrase for the private key.

3. Enter a passphrase for the private key, or press Enter to create a private key without a passphrase:

```
Enter passphrase (empty for no passphrase): passphrase
```

> **Note:**
>
> While a passphrase is not required, you should specify one as a security measure to protect the private key from unauthorized use. When you specify a passphrase, a user must enter the passphrase every time the private key is used.

The ssh-keygen utility prompts you to enter the passphrase again.

4. Enter the passphrase again, or press Enter again to continue creating a private key without a passphrase:

```
Enter the same passphrase again: passphrase
```

5. The ssh-keygen utility displays a message indicating that the private key has been saved as *filename* and the public key has been saved as *filename*`.pub`. It also displays information about the key fingerprint and randomart image.

Generating an SSH Key Pair on Windows Using the PuTTYgen Program

The PuTTYgen program is part of PuTTY, an open source networking client for the Windows platform.

To generate an SSH key pair on Windows using the PuTTYgen program:

1. Download and install PuTTY or PuTTYgen.

   To download PuTTY or PuTTYgen, go to http://www.putty.org/ and click the **You can download PuTTY here** link.

2. Run the PuTTYgen program.

   The PuTTY Key Generator window is displayed.

3. Set the **Type of key to generate** option to **SSH-2 RSA**.
4. In the **Number of bits in a generated key** box, enter **2048**.
5. Click Generate to generate a public/private key pair.

   As the key is being generated, move the mouse around the blank area as directed.

6. (Optional) Enter a passphrase for the private key in the **Key passphrase** box and reenter it in the **Confirm passphrase** box.

> **Note:**
>
> While a passphrase is not required, you should specify one as a security measure to protect the private key from unauthorized use. When you specify a passphrase, a user must enter the passphrase every time the private key is used.

7. Click **Save private key** to save the private key to a file. To adhere to file-naming conventions, you should give the private key file an extension of `.ppk` (PuTTY private key).

> **Note:**
>
> The `.ppk` file extension indicates that the private key is in PuTTY's proprietary format. You must use a key of this format when using PuTTY as your SSH client. It cannot be used with other SSH client tools. Refer to the PuTTY documentation to convert a private key in this format to a different format.

8. Select all of the characters in the **Public key for pasting into OpenSSH authorized_keys file** box.

   Make sure you select all the characters, not just the ones you can see in the narrow window. If a scroll bar is next to the characters, you aren't seeing all the characters.

9. Right-click somewhere in the selected text and select **Copy** from the menu.
10. Open a text editor and paste the characters, just as you copied them. Start at the first character in the text editor, and do not insert any line breaks.
11. Save the text file in the same folder where you saved the private key, using the `.pub` extension to indicate that the file contains a public key.
12. If you or others are going to use an SSH client that requires the OpenSSH format for private keys (such as the `ssh` utility on Linux), export the private key:
    a. On the **Conversions** menu, choose **Export OpenSSH key**.
    b. Save the private key in OpenSSH format in the same folder where you saved the private key in `.ppk` format, using an extension such as `.openssh` to indicate the file's content.

Creating an SSH Tunnel to a Compute Node Port

To create an SSH tunnel to a port on a compute node associated with Oracle Database Cloud Service, you use Secure Shell (SSH) client software that supports tunneling.

Several SSH clients that support tunneling are freely available. The following sections show how to use SSH clients on the Linux and Windows platforms to connect to a compute node using an SSH tunnel.

Creating an SSH Tunnel Using the ssh Utility on Linux

The Linux platform includes the ssh utility, an SSH client that supports SSH tunneling.

Before you use the ssh utility to create an SSH tunnel, you need the following:

- The IP address of the target compute node.

  The IP addresses associated with a database deployment on Oracle Database Cloud Service are listed on the details page associated with the database deployment. See Viewing Detailed Information for a Database Deployment.
- The SSH private key file that pairs with the public key used during the database deployment creation process.
- The port number for which you want to create an SSH tunnel.

To create an SSH tunnel for a port using the ssh utility on Linux:

1. In a command shell, set the file permissions of the private key file so that only you have access to it:

   ```
   $ chmod 600 private-key-file
   ```

   private-key-file is the path to the SSH private key file that matches the public key used during the database deployment creation process.
2. Run the ssh utility:

   ```
   $ ssh -i private-key-file -L local-port:target-ip-address:target-port opc@target-ip-address
   ```

   where:

   - ***private-key-file*** is the path to the SSH private key file.
   - ***local-port*** is the number of an available port on your Linux system. Specify a port number greater than 1023 and less than 49152 to avoid conflicts with ports that are reserved for the system. As a good practice, and for the sake of simplicity, you should specify the same port number as the one to which you are creating a tunnel.
   - ***target-ip-address*** is the IP address of the target compute node in *x.x.x.x* format.
   - ***target-port*** is the port number to which you want to create a tunnel.
3. If this is the first time you are connecting to the target compute node, the ssh utility prompts you to confirm the public key. In response to the prompt, enter **yes**.

After the SSH tunnel is created, you can access the port on the target compute node by specifying `localhost:`*`local-port`* on your Linux system.

Creating an SSH Tunnel Using the PuTTY Program on Windows

PuTTY is a freely available SSH client program for Windows that supports SSH tunneling.

Before you use the ssh utility to create an SSH tunnel, you need the following:

- The IP address of the target compute node.

The IP addresses associated with a database deployment on Oracle Database Cloud Service are listed on the details page associated with the database deployment. See Viewing Detailed Information for a Database Deployment.

- The SSH private key file that pairs with the public key used during the database deployment creation process.
- The port number for which you want to create an SSH tunnel.

To create an SSH tunnel for a port using the PuTTY program on Windows:

1. Download and install PuTTY.

   To download PuTTY, go to http://www.putty.org/ and click the **You can download PuTTY here** link.

2. Run the PuTTY program.

   The PuTTY Configuration window is displayed, showing the Session panel.

3. Configure SSH connectivity:

   a. In **Host Name (or IP address)** box, enter the IP address of the target compute node.

   b. Confirm that the **Connection type** option is set to **SSH**.

   c. In the Category tree, expand **Connection** if necessary and then click **Data**.

      The Data panel is displayed.

   d. In **Auto-login username** box, enter **oracle**.

   e. Confirm that the **When username is not specified** option is set to **Prompt**.

   f. In the Category tree, expand **SSH** and then click **Auth**.

      The Auth panel is displayed.

   g. Click the **Browse** button next to the **Private key file for authentication** box. Then, in the Select private key file window, navigate to and open the private key file that matches the public key used during the database deployment creation process.

4. Add a forwarded port:

   a. In the Category tree, click **Tunnels**.

      The Tunnels panel is displayed.

   b. In the **Source Port** box, enter the number of an available port on your system. Specify a port number greater than 1023 and less than 49152 to avoid conflicts with ports that are reserved for the system. As a good practice, and for the sake of simplicity, you should specify the same port number as the one to which you are creating a tunnel.

   c. In the **Destination box**, enter the IP address of the target compute node, a colon, and the port number to which you want to create a tunnel; for example, 192.0.2.100:1521.

   d. Confirm that the **Local** and **Auto** options are set.

   e. Click Add to add the forwarded port.

      The new forwarded port appears in the **Forwarded ports** list.

5. In the Category tree, click **Session**.

   The Session panel is displayed.

6. In the **Saved Sessions** box, enter a name for this connection configuration. Then, click **Save**.
7. Click **Open** to open the connection.

   The PuTTY Configuration window is closed and the PuTTY window is displayed.
8. If this is the first time you are connecting to the target compute node, the PuTTY Security Alert window is displayed, prompting you to confirm the public key. Click **Yes** to continue connecting.

After the SSH tunnel is created, you can access the port on the target compute node by specifying `localhost:`*`local-port`* on your system, where *`local-port`* is the source port that you specified when creating the tunnel.

Enabling Access to a Compute Node Port


This topic does not apply to Oracle Cloud Infrastructure.

For database deployments created in Oracle Cloud Infrastructure Classic, Oracle Database Cloud Service uses access rules to provide secure network access to database deployments. You can use the Oracle Database Cloud Service console to perform network access operations such as enabling and disabling access rules and creating new access rules.

When a database deployment is created, the following access rules are created, but set to a disabled status.

- **ora_p2_dbconsole**, which controls access to port 1158, the port used by Enterprise Manager 11g Database Control.
- **ora_p2_dbexpress**, which controls access to port 5500, the port used by Enterprise Manager Database Express 12c.
- **ora_p2_dblistener**, which controls access to the port used by SQL*Net.
- **ora_p2_http**, which controls access to port 80, the port used for HTTP connections.
- **ora_p2_httpssl**, which controls access to port 443, the port used for HTTPS connections, including Oracle REST Data Services, Oracle Application Express, and Oracle SQL Developer Web.

To enable access to a compute node port, you enable the appropriate access rule. When you enable one of the predefined access rules, the given port on the compute node is opened to the public internet. To enable access to a different port, or restrict access to a port, you must create an access rule.

Topics

- Enabling Port Access by Enabling an Automatically Created Access Rule
- Enabling or Restricting Port Access by Creating an Access Rule

Enabling Port Access by Enabling an Automatically Created Access Rule


This topic does not apply to Oracle Cloud Infrastructure.

You can use the Oracle Database Cloud Service console to enable one of the automatically created access rules:

1. Open the Access Rules page for the database deployment for which you want to enable an access rule:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. From the menu for the database deployment, select **Access Rules**.

      The Access Rules page is displayed.
2. Locate the rule you want to enable.
3. From the menu for the located rule, select **Enable**.

   The Enable Access Rule window is displayed.
4. Select **Enable**.

   The Enable Access Rule window closes and the rule is displayed as enabled in the list of rules. The given port on the compute node is opened to the public internet.

Enabling or Restricting Port Access by Creating an Access Rule


This topic does not apply to Oracle Cloud Infrastructure.

You can create an access rule to enable ports not associated with a predefined rule, or to restrict access to ports to only permit connections from specific IP addresses:

1. Open the Access Rules page for the database deployment for which you want to create an access rule:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. From the menu for the database deployment, select **Access Rules**.

      The Access Rules page is displayed. For information about the details provided on this page, see Access Rules Page.
2. Click **Create Rule**. In the Create Access Rule dialog, enter the following information.
   - **Rule Name:** Any name to identify this rule. Must start with a letter, followed by letters, numbers, hyphens, or underscores. Cannot start with `ora_` or `sys_`.

- **Description:** Any description of your choice (optional).
- **Source:** The hosts from which traffic should be allowed. Choices are:
  - **DB_1:** The `ora_db` security list for the deployment.
  - **PUBLIC-INTERNET:** The `public-internet` Security IP List.
  - **custom:** A custom list of addresses from which traffic should be allowed. In the field that displays below when you select this option, enter a comma-separated list of the subnets (in CIDR format) or IPv4 addresses for which you want to permit access.
- **Destination:** The security list to which traffic should be allowed. The only option is **DB_1**.
- **Destination Port(s):** The port or range of ports you want to open. Specify a single port, such as 5001, or a range of ports separated by a hyphen, such as 5001-5010.

3. Click **Create**.

   The Create Access Rule dialog closes and the rule is displayed in the list of rules. New rules are enabled by default.

> **Tip:**
>
> If necessary, adjust the number of results displayed on the Access Rules page so you can see the newly created rule.

Enabling Access to Database Cloud Service Using FastConnect Classic

You can use Oracle Cloud Infrastructure FastConnect Classic to access Oracle Database Cloud Service using a reliable, private and direct connection from your corporate network. When you use FastConnect Classic, your network traffic is routed over a direct and deterministic path that is separate from the public Internet. Consequently, FastConnect Classic provides guaranteed bandwidth and delivers consistent performance and latency.

FastConnect Classic is an add-on networking service, which is available for an additional subscription fee. FastConnect Classic is offered in increments of 1 Gbps and 10 Gbps, and you can combine 1 Gbps and 10 Gbps ports to meet your required network bandwidth.

The configuration of FastConnect Classic depends on the network configuration of the Database Cloud Service instance:

- If the Database Cloud Service instance is configured to use IP networks, FastConnect private peering is used. In this case, your corporate network RFC1918 IP address space is advertised to the Oracle FastConnect router and Oracle advertises the RFC1918 IP address spaces of the IP networks.
- If the Database Cloud Service instance is not configured to use IP networks, FastConnect public peering is used. In this case, your corporate network public IP addresses are advertised to the Oracle FastConnect router. If your corporate network used private IP addresses internally, then you must use a Network

Address Translation (NAT) to define the required public addresses. Oracle advertises all of the public IP addresses assigned to your Database Cloud Service instance.

The FastConnect provisioning process is a collaborative effort between Oracle Cloud network engineers and your corporate network administrators, which starts when you make a service request to configure FastConnect Classic in conjunction with Database Cloud Service.

Defining a Custom Host Name or Domain Name for Database Cloud Service


This topic does not apply to Oracle Cloud Infrastructure.

You can associate a custom host name or domain name to the public IP address of a compute node associated with your Oracle Database Cloud Service environment.

To associate a custom host name to the public IP address of a compute node, contact the administrator of your DNS (Domain Name Service) and request a custom DNS record for the compute node's public IP address. For example, if your domain is `example.com` and you wanted to use `clouddb1` as the custom host name for a compute node, you would request a DNS record that associates `clouddb1.example.com` to your compute node's public IP address.

To associate a custom domain name to the public IP address of a compute node:

1. Register your domain name through a third-party domain registration vendor, such as `Register.com`, `Namecheap`, and so on. For example, `example.com`.
2. Resolve your domain name to the IP address of the Database Cloud Service compute node, using the third-party domain registration vendor console. For more information, refer to the third-party domain registration documentation.

You can obtain the public IP address of a compute node by viewing details as described in Viewing Detailed Information for a Database Deployment.

Using Network Encryption and Integrity

To secure connections to your Oracle Database Cloud Service databases, you can use native Oracle Net Services encryption and integrity capabilities.

Encryption of network data provides data privacy so that unauthorized parties are not able to view data as it passes over the network. In addition, integrity algorithms protect against data modification and illegitimate replay.

Oracle Database provides the Advanced Encryption Standard (AES), DES, 3DES, and RC4 symmetric cryptosystems for protecting the confidentiality of Oracle Net Services traffic. It also provides a keyed, sequenced implementation of the Message Digest 5 (MD5) algorithm or the Secure Hash Algorithm (SHA-1 and SHA-2) to protect against integrity attacks.

By default, database deployments on Database Cloud Service are configured to enable native Oracle Net Services encryption and integrity. Also, by default, Oracle Net Services clients are configured to enable native encryption and integrity when they connect to an appropriately configured server. If your Oracle Net Services client is

configured to explicitly reject the use of native encryption and integrity then connection attempts will fail.

You can check your configuration and verify the use of native Oracle Net Services encryption and integrity as follows. For more general information about configuring native Oracle Net Services encryption and integrity, see "Configuring Oracle Database Network Encryption and Data Integrity" in *Oracle Database Security Guide* for Release 18, 12.2 or 12.1 or "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients" in *Database Advanced Security Administrator's Guide* for Release 11.2.

Checking your Database Cloud Service environment

The following procedure outlines the basic steps required to confirm that native Oracle Net Services encryption and integrity are enabled in your Database Cloud Service environment.

1. In a command shell, connect to the compute node as the `oracle` user. See Connecting to a Compute Node Through Secure Shell (SSH).
2. Change directories to the location of the `sqlnet.ora` configuration file. For example:

   ```
   $ cd $ORACLE_HOME/network/admin
   $ ls sqlnet.ora
   sqlnet.ora
   ```

3. View the `sqlnet.ora` file and confirm that it contains the following parameter settings:

   ```
   SQLNET.ENCRYPTION_SERVER = required
   SQLNET.CRYPTO_CHECKSUM_SERVER = required
   ```

   The `required` setting enables the encryption or integrity service and disallows the connection if the client side is not enabled for the security service. This is the default setting for database deployments on Database Cloud Service.

Checking your Oracle Net Services Client Configuration

The following procedure outlines the basic steps required to confirm that native encryption and integrity are enabled in your Oracle Net Services client configuration.

1. In a command shell, connect to the Oracle Net Services client.
2. Change directories to the location of the `tnsnames.ora` and `sqlnet.ora` configuration files, for example:

   ```
   $ cd $ORACLE_HOME/network/admin
   $ ls *.ora
   sqlnet.ora tnsnames.ora
   ```

3. View the `sqlnet.ora` file and confirm that it *does not* contain the following parameter settings:

   ```
   SQLNET.ENCRYPTION_CLIENT = rejected
   SQLNET.CRYPTO_CHECKSUM_CLIENT = rejected
   ```

The `rejected` setting explicitly disables the encryption or integrity service, even if the server requires it. When a client with an encryption or integrity service setting of `rejected` connects to a server with the `required` setting, the connection fails with the following error: `ORA-12660: Encryption or crypto-checksumming parameters incompatible`.

Because native Oracle Net Services encryption and integrity are enabled in your Database Cloud Service environment by default, any parameter setting other than `rejected`, or no setting at all, would result in the use of native encryption and integrity.

Verifying the use of Native Encryption and Integrity

You can verify the use of native Oracle Net Services encryption and integrity by connecting to your Oracle database and examining the network service banner entries associated with each connection. This information is contained in the `NETWORK_SERVICE_BANNER` column of the `V$SESSION_CONNECT_INFO` view. The following example shows the SQL command used to display the network service banner entries associated with current connection:

```
SQL> select network_service_banner
     from v$session_connect_info
     where sid in (select distinct sid from v$mystat);
```

The following example output shows banner information for the available encryption service and the crypto-checksumming (integrity) service, including the algorithms in use:

```
NETWORK_SERVICE_BANNER
----------------------------------------------------------------------------
----------
TCP/IP NT Protocol Adapter for Linux: Version 12.1.0.2.0 - Production
Encryption service for Linux: Version 12.1.0.2.0 - Production
AES256 Encryption service adapter for Linux: Version 12.1.0.2.0 -
Production
Crypto-checksumming service for Linux: Version 12.1.0.2.0 - Production
SHA1 Crypto-checksumming service adapter for Linux: Version 12.1.0.2.0 -
Production
```

If native Oracle Net Services encryption and integrity was not in use, the banner entries would still include entries for the available security services; that is, the services linked into the Oracle Database software. However, there would be no entries indicating the specific algorithms in use for the connection. The following output shows an example:

```
NETWORK_SERVICE_BANNER
----------------------------------------------------------------------------
----------
TCP/IP NT Protocol Adapter for Linux: Version 12.1.0.2.0 - Production
Encryption service for Linux: Version 12.1.0.2.0 - Production
Crypto-checksumming service for Linux: Version 12.1.0.2.0 - Production
```

4

Administering Database Cloud Service

This section describes tasks for administering your Oracle Database Cloud Service environment and the Oracle databases contained therein.

Topics

Adding Temporary Storage to a Database Deployment


This topic does not apply to Oracle Cloud Infrastructure.

In general, when you add storage to a database deployment on Oracle Database Cloud Service, you want the storage to be permanent; that is, to remain attached and available until the deployment is deleted. To add this kind of permanent storage, scale up the storage as described in Scaling a Database Deployment.

However, you may sometimes want to add storage to a database deployment temporarily for a short period of time, after which you want to detach and delete the storage.

Topics

Adding Temporary Storage to a Database Deployment

To add temporary storage to a database deployment, you add a storage volume to a compute node. First, you create a Compute Cloud Service storage volume and attach it to the compute node. Then, while logged into the compute node you use Linux commands to partition, format and mount the storage volume.

The storage you add by following these steps is “temporary” in that you can later unmount it from the compute node and delete it. In all other ways it is “permanent”: it remains in existence, even if you delete the database deployment to which it is attached, until you delete it.

> **Note:**
>
> If the database deployment to which you attach this temporary storage is restarted or is stopped and then started, the storage volume becomes detached from the compute node and you must reattach it. For instructions, see Attaching a Storage Volume to an Instance in *Using Oracle Cloud Infrastructure Compute Classic*. After reattaching the storage volume, you must then connect to the compute node and remount it.
>
> If the compute node is rebooted, such as when following the instructions in Rebooting a Compute Node, the temporary storage becomes unmounted and you must remount it.

When adding a Compute Cloud Service storage volume as temporary storage, keep these points in mind:

- A compute node can have a maximum of ten storage volumes attached to it.
- You can create a storage volume from 1 GB to 2048 GB in size, in increments of 1 GB.

To add temporary storage to a database deployment:

1. Use the Create Storage Volume wizard in the Compute Cloud Service console to create a storage volume.

   Follow the instructions provided in Creating a Storage Volume in *Using Oracle Cloud Infrastructure Compute Classic*, choosing **storage/default** as the value for **Storage Property**.

2. Attach the storage volume to the Compute Cloud Service instance on which the compute node is running by following the instructions in Attaching a Storage Volume to an Instance in *Using Oracle Cloud Infrastructure Compute Classic*.

   When you attach the storage volume, it is assigned a disk number. Note down this disk number for later use.

3. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

4. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

5. Confirm the addition of the storage volume by using the `ls` command:

   ```
   # ls /dev/xvd*
   ```

   In the listing that is displayed, look for an entry of the form `/dev/xvd`*`LETTER`*, where *`LETTER`* is the letter that is alphabetically one higher than the storage volume’s disk number.

For example, if the storage volume's disk number is 6, you should look for the entry `/dev/xvdg` because the letter "g" is the seventh letter of the alphabet.

6. Create a single, primary partition that occupies the entire storage volume by using the `fdisk` command. For example:

```
# fdisk /dev/xvdg
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF
disklabel
Building a new DOS disklabel with disk identifier 0xaa660f6f.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

Command (m for help): c
DOS Compatibility flag is not set

Command (m for help): u
Changing display/entry units to sectors

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (2048-20971519, default 2048): [press Enter]
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519): [press
Enter]
Using default value 20971519

Command (m for help): p

Disk /dev/xvdg: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xaa660f6f

    Device Boot      Start         End      Blocks   Id  System
/dev/xvdg1            2048    20971519    10484736   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

In this example, note the use of the **p** command to print the partition table before writing the table and exiting. Use this command and note down the name of the new partition as displayed in the Device column.

7. Create a file system on the partition by using the `mkfs` command. For example:

```
# mkfs -t ext4 /dev/xvdg1
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621184 blocks
131059 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2684354560
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 32 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

8. Create a directory to use as the mount point for the partition by using the `mkdir` command. For example:

   ```
   # mkdir /u05
   ```

9. Mount the partition on the directory you just created by using the `mount` command. For example:

   ```
   # mount /dev/xvdg1 /u05
   ```

10. Set the ownership and permissions of the mount-point directory appropriately by using the `chown` and `chmod` commands. For example:

    ```
    # chown oracle:oinstall /u05
    # chmod 755 /u05
    ```

11. Exit the root-user command shell:

    ```
    # exit
    $
    ```

Deleting Temporary Storage from a Database Deployment

To delete temporary storage, you unmount the storage on the compute node and then detach the storage volume from the Compute Cloud Service instance and delete it.

To delete temporary storage from a database deployment:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Display mounted filesystems and confirm the mount point of your temporary storage volume:

```
# df -hT
```

4. Unmount your temporary storage volume; for example:

```
# umount /u05
```

> **Note:**
>
> The Linux command to unmount a volume is *u*`mount` (with no *n*).

5. Exit the root-user command shell and disconnect from the compute node:

```
# exit
exit
$ exit
```

6. Open the Oracle Compute Cloud Service console.

   For instructions, see Accessing Oracle Compute Cloud Service in *Using Oracle Cloud Infrastructure Compute Classic*.

   When you open the Oracle Compute Cloud Service console, the Instances page is displayed .

7. Click **Storage** to display a list of storage volumes.

8. In the list of storage volumes, click the menu in the row for your temporary storage volume and choose **Detach from Instance** and then confirm the action.

9. Refresh the page occasionally until the row for your temporary storage no longer shows that it is attached.

10. From the menu in the row for your temporary storage volume, choose **Delete** and then confirm the action.

    The Oracle Compute Cloud service sets the status of the storage volume to Deleting and begins deleting it.

Adding an SSH Public Key

Should the need arise, you can add an SSH public key to your Oracle Database Cloud Service environment. After you add the public key, you can provide the matching private key to connect to a compute node using SSH as either the `opc` or the `oracle` user.

To add an SSH public key:

1. Go to the SSH Access page of the deployment you want to add a public key to:

   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. From the menu for a database deployment, select **SSH Access**.

      The **Add New Key** overlay is displayed with its **Key value** field displaying the most recent SSH public key.

2. Click **Add New Key**.

You can also add SSH public keys to one or more deployments on the SSH Access Page.

Updating the Cloud Tooling on Database Cloud Service

How you update the cloud tooling on Oracle Database Cloud Service depends on the type of database deployment you are using:

- For database deployments hosting single-instance databases, you can manually update the cloud tooling by following the instructions in Updating the Cloud Tooling by Using the dbaascli Utility, or you can configure automatic cloud tooling updates by following the instructions in Configuring Automatic Cloud Tooling Updates.
- For database deployments hosting Oracle Real Application Clusters (RAC) databases, see Updating the Cloud Tooling by Using the raccli Utility.

Check the Cloud Tooling and Image Versions

You can use a function of the dbaasapi to determine the current tools version and the image version.

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Create a JSON file named `input.json` with the following content:

   ```
   {
     "object":  "os",
     "action":  "get",
     "operation": "info_version",
     "params": {
       "infofile":"/home/opc/info.json"
     },
     "outputfile":   "/home/opc/out.json",
     "FLAGS": "debug"
   }
   ```

4. Run the dbaasapi command:

   ```
   # dbaasapi -i /home/opc/input.json
   ```

5. Display the results:

   ```
   # cat /home/opc/info.json
   ```

   Example output:

   ```
   {
      "dbtools_dbaas_monitor" : null,
      "dbaas_image" :
   "OL_6.8_UEKR4_x86_64_PSM_MANAGED-18.3.2-20180627-131137.tar.gz",
      "dbtools_ords_sdw" :
   "dbtools_ords_standalone-18.2.0-1.r1831332.el7.x86_64",
      "dbaastools_version" : "dbaastools-1.0-1+18.4.3.0.0_181011.1252.x86_64"
   ...
   ```

6. Exit the session:

```
# exit
$ exit
```

Updating the Cloud Tooling by Using the dbaascli Utility

You use the `patch tools` subcommand of the `dbaascli` utility to update the cloud tooling on database deployments hosting a single-instance database or an Oracle Data Guard configuration of single-instance databases.

> **Note:**
>
> When updating the cloud tooling on database deployments hosting a Data Guard configuration, you must perform the following steps **on both nodes**; that is, on the one hosting the primary database and on the one hosting the standby database.

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Check whether any cloud tooling updates are available:

```
# dbaascli patch tools list
```

   If you receive an error message stating that this option is not supported, the cloud tooling on the compute node is too old to support this method of updating cloud tooling. In this case, follow the instructions in I can't use dbaascli to update my cloud tooling.

   The `patch tools list` subcommand is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm list_tools command.

4. In the command response, locate the patch ID of the cloud tooling update.

   The patch ID is listed as the "Patchid" value. If multiple patches are listed, choose the latest one.

5. Check your current version of cloud tooling:

```
# rpm -qa | grep -i dbaastools
dbaastools-version_number-release_number
```

6. After confirming that the latest update is newer than your current version, download and apply the patch containing the latest cloud tooling update:

```
# dbaascli patch tools apply --patchid LATEST
```

   Note: if you get a warning message indicating that this parameter is invalid, you can ignore the message.


The `patch tools apply` subcommand is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm toolsinst command.

7. Reset the backup configuration:

```
# /var/opt/oracle/ocde/assistants/bkup/bkup
```

8. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

9. If you are updating cloud tooling on a database deployment hosting a Data Guard configuration, repeat the preceding steps on the other compute node of the deployment.

Configuring Automatic Cloud Tooling Updates


This topic does not apply to Oracle Cloud at Customer.

You can configure automatic cloud tooling updates for compute nodes hosting Oracle Database Cloud Service database deployments that use a single-instance database or an Oracle Data Guard configuration of single-instance databases.

An entry is added to the `/etc/crontab` file to regularly check for cloud tooling updates and apply new updates to the compute node when they become available.

> **Note:**
>
> Currently, automatic cloud tooling updates are not supported for Database Cloud Service database deployments that use Oracle Real Application Clusters (RAC).

Here are the tasks for configuring automatic cloud tooling updates:

- Enabling Automatic Cloud Tooling Updates
- Disabling Automatic Cloud Tooling Updates
- Checking the Status of Automatic Cloud Tooling Updates
- Performing On-Demand Cloud Tooling Updates

Enabling Automatic Cloud Tooling Updates

To enable automatic cloud tooling updates for a database deployment:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Enter the following command:

```
# dbaascli patch tools auto enable
```

4. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

5. If you are enabling automatic cloud tooling updates for a database deployment hosting a Data Guard configuration, repeat the preceding steps on the other compute node of the deployment.

Disabling Automatic Cloud Tooling Updates

To disable automatic cloud tooling updates for a database deployment:

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Enter the following command:

```
# dbaascli patch tools auto disable
```

4. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

5. If you are disabling automatic cloud tooling updates for a database deployment hosting a Data Guard configuration, repeat the preceding steps on the other compute node of the deployment.

Checking the Status of Automatic Cloud Tooling Updates

To check whether automatic cloud tooling updates are enabled or disabled for a database deployment:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Enter the following command:

```
# dbaascli patch tools auto status
```

   - If the command response includes “INFO: auto rpm update is enabled”, then automatic updates are enabled.
   - If the command response includes “INFO: auto rpm update is disabled”, then automatic updates are disabled.

4. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

5. If you are checking whether automatic cloud tooling updates are enabled or disabled for a database deployment hosting a Data Guard configuration, repeat the preceding steps on the other compute node of the deployment.

Performing On-Demand Cloud Tooling Updates

If you cannot wait for the next automatic cloud tooling update, you can perform an on-demand cloud tooling check and update. A check is performed to determine whether the latest available cloud tooling update is newer than the current version. If so, the update is downloaded and applied to the compute node.

> **Note:**
>
> In order to perform an on-demand cloud tooling update, automatic updates must be enabled for the compute node. See Enabling Automatic Cloud Tooling Updates.

To perform an on-demand cloud tooling update for a database deployment:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Perform an on-demand cloud tooling check and update:

   ```
   # dbaascli patch tools auto execute
   ```

4. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

5. If you are performing an on-demand cloud tooling update for a database deployment hosting a Data Guard configuration, repeat the preceding steps on the other compute node of the deployment.

Updating the Cloud Tooling by Using the raccli Utility

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You use the `raccli` utility to update the cloud tooling on database deployments hosting an Oracle RAC database.

> **Note:**
>
> If you are updating the cloud tooling on a database deployment that also uses Oracle Data Guard, you must also execute the `update databasepassword` command to store the password in the keystore (wallet) if you are updating from release 17.2.1 or earlier.

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Execute the `raccli update rdk` subcommand:

   > **Caution:**
   >
   > After executing the `raccli update rdk` command, wait three minutes before executing commands using either raccli or the user interface to allow time for the server to restart.

   ```
   $ raccli update rdk -tag tag-number
   ```

   where *tag-number* is the version of tooling you want to update to, without the dots in the version number. For example, to update to 17.2.5 tooling you would enter **1725**.

   To find out the tag number for the latest available tooling update, see *What's New for Oracle Database Cloud Service*.

Administering a Data Guard Configuration

Oracle Database Cloud Service provides several commands and features to simplify the administration of database deployments that contain an Oracle Data Guard configuration.

> **Note:**
>
> Oracle Database Cloud Service does not currently include the fast-start failover (FSFO) feature of Oracle Data Guard. In Database Cloud Service, you perform failover operations manually, as described in Performing a Manual Failover Operation.

Topics

- Checking the Status of the Oracle Data Guard Configuration
- Performing a Switchover Operation
- Performing a Manual Failover Operation
- Reinstating a Failed Primary Database

- Changing the SYS Password
- Configuring Clients for Automatic Failover

Checking the Status of the Oracle Data Guard Configuration

> **Note:**
>
> This topic **does not** apply to database deployments using both Oracle RAC and Oracle Data Guard. For such deployments, use the `raccli status dataguard` command to check the status of the deployment.

You can use the `dataguard status` subcommand of the `dbaascli` utility to check the status of your Oracle Data Guard configuration.

Before performing certain operations, you may want to check the status of your Oracle Data Guard configuration.

To check the status of the Oracle Data Guard configuration by using the `dataguard status` subcommand:

1. Connect to either compute node in the Data Guard configuration as the **`oracle`** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Check the status of the configuration:

   ```
   $ dbaascli dataguard status [--details yes|no] [--password password]
   ```

   Use the `details` option to generate a detailed listing including data protection mode, redo transport services mode, maximum data loss potential, and approximate database role transition time.

   Use the `password` option to supply the `SYS` user password if you changed it since creating the Data Guard configuration.

Performing a Switchover Operation

You can perform a switchover to the standby database in your Oracle Data Guard configuration by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Perform a Switchover Operation at the end of this topic.

A switchover operation enables the primary database to switch roles with the standby database. There is no data loss during a switchover. After a switchover, each database continues to participate in the Oracle Data Guard configuration in its new role. A switchover is typically used to reduce primary database downtime during planned outages, such as operating system or hardware upgrades, or rolling upgrades of the Oracle Database software and patch sets. For more information, see "Switchovers" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2.

Performing a Switchover Operation by Using the Oracle Database Cloud Service Console

1. Go to the Overview page for the database deployment you want to perform a switchover on:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment you want to perform the switchover on.

      The Oracle Database Cloud Service Overview page is displayed.
2. To ensure the Overview page reflects the current role of each database, click the Refresh Configuration icon.
3. From the action menu ( ) located beside the deployment name or beside any of the compute nodes, select **Switchover**, and then confirm the action.

   The deployment shows a status of Maintenance in the Oracle Database Cloud Service console until the switchover is complete.
4. Refresh the page occasionally.

   Database Role will be updated to reflect the new role for each database.

> **Note:**
>
> For Data Guard configurations of two Oracle RAC databases, you must manually configure backups after the switchover operation completes. For instructions, see Manual backup configuration required after switchover or failover on Oracle RAC plus Data Guard deployments in *Known Issues for Oracle Database Cloud Service*.

Other Ways to Perform a Switchover Operation

- For Data Guard configurations of two single-instance databases, you can use the `dataguard switchover` subcommand of the `dbaascli` utility. See Performing a Switchover Operation by Using the dbaascli Utility.
- For Data Guard configurations of two Oracle RAC databases, you can use the `raccli switchover dataguard` command.

Performing a Switchover Operation by Using the dbaascli Utility

You can use the `dataguard switchover` subcommand of the `dbaascli` utility to perform a switchover to the standby database in your Oracle Data Guard configuration.

To perform a switchover by using the `dataguard switchover` subcommand:

1. Connect to the compute node in the Oracle Data Guard configuration that will host the new primary database as the **opc** user.

For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell and then switch to the **oracle** user:

```
$ sudo -s
# su - oracle
$
```

3. Initiate the switchover to the standby database:

```
$ dbaascli dataguard switchover [--password password]
```

Use the `password` option to supply the `SYS` user password if you changed it since creating the Data Guard configuration.

4. Return to being the root user:

```
$ exit
#
```

5. Restart the ORDS server:

```
# /etc/init.d/ords restart
```

6. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Performing a Manual Failover Operation

You can perform a manual failover to the standby database in your Oracle Data Guard configuration by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Perform a Manual Failover Operation at the end of this topic.

A failover operation changes the standby database to the primary role in response to a primary database failure. If the primary database was not operating in either maximum protection mode or maximum availability mode before the failure, some data loss may occur. If Flashback Database is enabled on the primary database, it can be reinstated as a standby for the new primary database once the reason for the failure is corrected. A failover is typically used only when the primary database becomes unavailable, and there is no possibility of restoring it to service within a reasonable period of time. For more information, see "Failovers" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2.

Performing a Manual Failover Operation by Using the Oracle Database Cloud Service Console

1. Go to the Overview page for the database deployment you want to perform the failover on:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment you want to perform the failover on.

      The Oracle Database Cloud Service Overview page is displayed.

2. To ensure the Overview page reflects the current role of each database, click the Refresh Configuration icon.
3. From the action menu ( ) located beside the deployment name or beside any of the compute nodes, select **Failover**, and then confirm the action.

   The deployment shows a status of Maintenance in the Oracle Database Cloud Service console until the operation is complete.
4. Refresh the page occasionally.

   Database Role will be updated to reflect the new role for each database.

> **Note:**
>
> For Data Guard configurations of two Oracle RAC databases, you must manually configure backups after the failover operation completes. For instructions, see Manual backup configuration required after switchover or failover on Oracle RAC plus Data Guard deployments in *Known Issues for Oracle Database Cloud Service*.

Other Ways to Perform a Manual Failover Operation

- For Data Guard configurations of two single-instance databases, you can use the `dataguard failover` subcommand of the `dbaascli` utility. See Performing a Manual Failover Operation by Using the dbaascli Utility.
- For Data Guard configurations of two Oracle RAC databases, you can use the `raccli failover dataguard` command.

Performing a Manual Failover Operation by Using the dbaascli Utility

You can use the `dataguard failover` subcommand of the `dbaascli` utility to perform a manual failover to the standby database in your Oracle Data Guard configuration.

To perform a failover by using the `dataguard failover` subcommand:

1. Connect to the compute node in the Oracle Data Guard configuration that will host the new primary database as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).
2. Start a root-user command shell and then switch to the **oracle** user:

   ```
   $ sudo -s
   # su - oracle
   $
   ```
3. Initiate the failover to the standby database:

   ```
   $ dbaascli dataguard failover [--force yes|no] [--password password]
   ```

   Use the `force` option if the `status` subcommand shows that the Data Guard configuration is in a warning or error state.

   Use the `password` option to supply the `SYS` user password if you changed it since creating the Data Guard configuration.

4. Return to being the root user:

```
$ exit
#
```

5. Restart the ORDS server:

```
# /etc/init.d/ords restart
```

6. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Reinstating a Failed Primary Database

You can reinstate a failed primary database after a failover by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Reinstate a Failed Primary Database at the end of this topic.

After performing a failover to the standby database, you may be able to restore your original disaster-recovery solution by reinstating the failed primary database. You can use the Data Guard broker's reinstate capability to make the failed primary database a viable standby database for the new primary. For more information, see "Reenabling Disabled Databases After a Role Change" in *Oracle Data Guard Broker* for Release 18, 12.2, 12.1 or 11.2.

Reinstating a Failed Primary Database by Using the Oracle Database Cloud Service Console

1. Go to the Overview page for the database deployment you want to perform the reinstate on:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. In the list of deployments, click the name of the database deployment you want to perform the reinstate on.

      The Oracle Database Cloud Service Overview page is displayed.
2. To ensure the Overview page reflects the current role of each database, click the Refresh Configuration icon.
3. From the action menu ( ) located beside the deployment name , select **Reinstate**, and then confirm the action.

   The deployment has a status of Maintenance in the Oracle Database Cloud Service console until the operation is complete.
4. Refresh the page occasionally.

   Database Role will be updated to reflect the new role for each database.

Other Ways to Reinstate a Failed Primary Database

- For Data Guard configurations of two single-instance databases, you can use the `dataguard reinstate` subcommand of the `dbaascli` utility. See Reinstating a Failed Primary Database by Using the dbaascli Utility.

- For Data Guard configurations of two Oracle RAC databases, you can use the `raccli reinstate dataguard` command.

Reinstating a Failed Primary Database by Using the dbaascli Utility

You can use the `dataguard reinstate` subcommand of the `dbaascli` utility to reinstate a failed primary database after a failover.

To determine whether the database can be reinstated, use the `dataguard status` subcommand as described in Checking the Status of the Oracle Data Guard Configuration. A status of `ORA-16661: the standby database needs to be reinstated` indicates the standby database can be reinstated.

To reinstate a failed primary database by using the `dataguard reinstate` subcommand:

1. Connect to one of the compute nodes in the Oracle Data Guard configuration as the **`oracle`** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Initiate the reinstatement of the failed primary database:

   ```
   $ dbaascli dataguard reinstate [--password password]
   ```

   Use the `password` option to supply the `SYS` user password if you changed it since creating the Data Guard configuration.

3. Disconnect from the compute node.

   ```
   $ exit
   ```

Changing the SYS Password

> **Note:**
>
> This topic **does not** apply to database deployments using both Oracle RAC and Oracle Data Guard. Such deployments do not include the `dbaascli` utility.

You can use the `database changepassword` subcommand of the `dbaascli` utility to change the password of the `SYS` user in your Oracle Data Guard configuration.

To use the `database changepassword` subcommand of the `dbaascli` utility:

1. Connect as the **`oracle`** user to the compute node hosting the primary database.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Execute the `dbaascli database changepassword` command.

   ```
   $ dbaascli database changepassword
   ```

   Enter the `SYS` user name and new password when prompted.

3. Disconnect from the compute node.

```
$ exit
```

Configuring Clients for Automatic Failover

> **Note:**
>
> This topic **does not** apply to database deployments using both Oracle RAC and Oracle Data Guard.

By using pre-defined network service names, application clients can automatically reconnect to a new primary database following a role transition.

Your Data Guard configuration on Oracle Database Cloud Service is pre-configured to provide automatic transition of application connections from a failed primary database to a new primary database after a Data Guard role transition has taken place.

The following network service names are pre-defined:

- *SID*_dg: This service is used to connect to the primary database. If the database uses Oracle Database 12c Release 1, or later, this service connects to the root container.
- *SID*_dg_ro: If the Data Guard configuration includes a standby database with real-time query (Active Data Guard), this service is defined and is used to connect to the standby database. If the database uses Oracle Database 12c Release 1, or later, this service connects to the root container.
- *PDBname*_dg: In an Oracle Data Guard configuration using Oracle Databases 12c Release 1, or later, this service is defined and is used to connect to the default PDB of the primary database.
- *PDBname*_dg_ro: In an Oracle Data Guard configuration using Oracle Databases 12c Release 1, or later, that includes a standby database with real-time query (Active Data Guard), this service is defined and is used to connect to the default PDB of the standby database.

The services are managed on each database through the use of pre-defined triggers. Following a role transition, the trigger is fired to start the services on the new primary database. By using the pre-defined network service names in your application connections, your application clients will be automatically directed to the new primary database following a role transition.

See Client Failover Best Practices for Highly Available Oracle Databases: Oracle Database 11g Release 2 or Client Failover Best Practices for Highly Available Oracle Databases Oracle Database 12c for detailed information.

Preparing 12.2 Database Deployments for Patching

Several enhancements and bug fixes have been made to Oracle Database Release 12.2 since its release on Oracle Cloud in November 2016. However, these improvements have not yet been gathered into a PSU. Database Cloud Service is making available a “sync-up” patch that includes these many improvements and prepares your database deployment for patching when the first PSU becomes

available for Release 12.2. You must apply this patch to bring Database Cloud Service database deployments running Release 12.2 up-to-date and to permit PSUs to be applied later.

The following sections describe how to apply the “sync-up” patch to your database deployment. The procedure for applying the patch depends on the type of database running in the deployment.

Topics

- Preparing 12.2 Database Deployments Hosting Single-Instance Databases for Patching
- Preparing 12.2 Database Deployments Hosting Oracle RAC Databases for Patching

Preparing 12.2 Database Deployments Hosting Single-Instance Databases for Patching

You can bring a 12.2 database deployment hosting a single-instance database up-to-date and prepare it for patching when the first PSU becomes available for Release 12.2.

Apply a “sync-up” patch (ID 24824889-EE) to your 12.2 database deployment. This patch is available only through command-line tools. To apply this patch to a database deployment hosting a single-instance database or an Oracle Data Guard configuration of single-instance databases, refer to Applying a Patch by Using the dbaascli Utility.

Preparing 12.2 Database Deployments Hosting Oracle RAC Databases for Patching

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can bring a 12.2 database deployment hosting an Oracle RAC database up-to-date and prepare it for patching when the first PSU becomes available for Release 12.2.

> **Note:**
>
> The following procedure is required only for 12.2 database deployments using Database Cloud Service release 17.2.1 or earlier.

Before you begin

- Make sure the database deployment has cloud tooling version 17.2.5.1 or later. For more information, see Updating the Cloud Tooling by Using the raccli Utility.
- Before starting the procedure, you should back up the deployment. For instructions, see Creating an On-Demand Backup.
- Plan for the following downtime while performing the procedure:

- **Database Clustering with RAC**—Both Oracle RAC nodes will be shut down simultaneously while you apply a patch to the database.
- **Database Clustering with RAC and Data Guard Standby**—First, both Oracle RAC nodes of the standby database will be shut down simultaneously while you apply a patch to the standby, and then both Oracle RAC nodes of the primary database will be shut down simultaneously while you apply a patch to the primary.

Procedure

Apply a “sync-up” patch to your 12.2 database deployment.

- For **Database Clustering with RAC** databases, perform the steps in this procedure once.
- For **Database Clustering with RAC and Data Guard Standby** databases, perform the steps in this procedure twice: first on your standby database and then on your primary database.

1. Connect as the **opc** user to compute node 1.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Apply patch 24976125 to your database:

   ```
   $ raccli apply patch -db -tag 24976125
   ```

   The Oracle Database home is updated on both compute nodes. Both nodes are shut down at the same time, patched, and brought back online.

3. Check your database for invalid objects.

   The patch application may have invalidated some database objects. You can check for invalid objects and validate them as follows:

   a. Invoke SQL*Plus and log in as the SYS user with SYSDBA privileges.

   b. Make sure you are connected to the CDB root and run the following query:

      ```
      SQL> select count(*) from cdb_objects where status='INVALID';
      ```

   c. Exit SQL*Plus:

      ```
      SQL> exit
      $
      ```

   d. If the preceding query returns a count value of 1 or greater, then the database contains invalid objects. You can validate them by running the following command. Enter the command on one line with no line breaks.

      ```
      $ $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
      utlrp_output $ORACLE_HOME/rdbms/admin/utlrp.sql
      ```

Applying Linux OS Security Patches

You can apply Linux OS security patches to compute nodes hosting Oracle Database Cloud Service database deployments. How you apply OS patches depends on the database type used to create the database deployment:

- Single Instance
- Database Clustering with RAC

- Single Instance with Data Guard Standby
- Database Clustering with RAC and Data Guard Standby
- Data Guard Standby for Hybrid DR

Single Instance

You can apply OS patches to the compute node for a deployment hosting a single-instance database by using the `dbaascli` utility, or you can manually apply OS patches.

- To apply OS patches by using the `dbaascli` utility, see Using the dbaascli Utility on Deployments Hosting a Single-Instance Database.
- To manually apply OS patches, follow the instructions in Manually Applying Linux OS Security Patches.

Database Clustering with RAC

You manually apply OS patches to the two compute nodes for a deployment hosting an Oracle RAC database. For each compute node, follow the instructions in Manually Applying Linux OS Security Patches.

Single Instance with Data Guard Standby

You can apply OS patches to the two compute nodes for a deployment hosting an Oracle Data Guard configuration of single-instance databases by using the `dbaascli` utility, or you can manually apply OS patches.

- To apply OS patches by using the `dbaascli` utility, see Using the dbaascli Utility on Deployments Hosting an Oracle Data Guard Configuration of Single-Instance Databases.
- To manually apply OS patches:
  1. For the compute node associated with the **standby database**, follow the instructions in Manually Applying Linux OS Security Patches.
  2. After the compute node and standby database have rebooted, allow a few minutes for redo data to be applied to the standby database.

     To monitor the progress, connect as the **opc** user to the standby compute node, and run the following command:

     ```
     $ dgmgrl show configuration verbose;
     ```

     The operation is complete when the preceding command returns no errors.
  3. Perform a switchover from the primary database to the standby database in your Oracle Data Guard configuration. See Performing a Switchover Operation.
  4. For the compute node associated with the **new standby database** (this was the primary database before you performed the switchover), follow the instructions in Manually Applying Linux OS Security Patches.
  5. After the compute node and new standby database have rebooted, allow a few minutes for redo data to be applied to the new standby database.

     To monitor the progress, connect as the **opc** user to the new standby compute node, and run the following command:

```
$ dgmgrl show configuration verbose;
```

The operation is complete when the preceding command returns no errors.

6. Perform a switchover back to the original primary database in your Oracle Data Guard configuration. See Performing a Switchover Operation.

Database Clustering with RAC and Data Guard Standby

You manually apply patches to the four compute nodes for a deployment hosting an Oracle Data Guard configuration of Oracle RAC databases:

1. For each of the two compute nodes associated with the **standby RAC database**, follow the instructions in Manually Applying Linux OS Security Patches.
2. After the compute nodes and standby RAC database have rebooted, allow a few minutes for redo data to be applied to the standby RAC database.

   To monitor the progress, connect as the **opc** user to one of the standby RAC database compute nodes, and run the following command:

   ```
   $ dgmgrl show configuration verbose;
   ```

   The operation is complete when the preceding command returns no errors.
3. Perform a switchover from the primary RAC database to the standby RAC database in your Oracle Data Guard configuration. See Performing a Switchover Operation.
4. For each of the two compute nodes associated with the **new standby RAC database** (this was the primary RAC database before you performed the switchover), follow the instructions in Manually Applying Linux OS Security Patches.
5. After the compute nodes and new standby RAC database have rebooted, allow a few minutes for redo data to be applied to the new standby RAC database.

   To monitor the progress, connect as the **opc** user to one of the new standby RAC database compute nodes, and run the following command:

   ```
   $ dgmgrl show configuration verbose;
   ```

   The operation is complete when the preceding command returns no errors.
6. Perform a switchover back to the original primary RAC database in your Oracle Data Guard configuration. See Performing a Switchover Operation.

Data Guard Standby for Hybrid DR

To manually apply OS patches to the compute node for the cloud standby database in a Hybrid DR deployment, follow the instructions in Manually Applying Linux OS Security Patches.

For information on applying OS patches to the compute node for the on-premises primary database in a Hybrid DR deployment, refer to the documentation for the appropriate Oracle Database version.

Applying Linux OS Security Patches by Using the dbaascli Utility

You can use the `dbaascli` utility to apply Linux OS security patches to compute nodes associated with database deployments hosting a single-instance database or an Oracle Data Guard configuration of single-instance databases.

Topics

- Using the dbaascli Utility on Deployments Hosting a Single-Instance Database
- Using the dbaascli Utility on Deployments Hosting an Oracle Data Guard Configuration of Single-Instance Databases

Using the dbaascli Utility on Deployments Hosting a Single-Instance Database

You can use the `dbaascli` utility to apply Linux OS security patches to a compute node associated with a database deployment hosting a single-instance database.

> **Note:**
>
> Before you begin, review all steps in the procedure to understand how the compute node is rebooted at the end of the patching process.

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Check whether any Linux OS security patches are available:

   ```
   # dbaascli patch os list
   ```

   
   This command is not yet available on Oracle Cloud at Customer. Instead, you must use the following command:

   ```
   # /var/opt/oracle/patch/dbpatchmsm -list_ospatches
   ```

   If the command response indicates that patches are available, continue at the next step.

   Otherwise, the latest patches are already installed on the compute node, and you can exit the root-user command shell and disconnect from the compute node.

4. Apply the latest Linux OS security patches:

   ```
   # dbaascli patch os apply
   ```

   
   This command is not yet available on Oracle Cloud at Customer. Instead, you must use the following command:

   ```
   # /var/opt/oracle/patch/dbpatchmsm -apply_ospatch_async
   ```

5. The `dbaascli` utility automatically reboots the compute node at the end of the patch installation, if necessary.

   If the utility does not reboot the compute node, then exit the root-user command shell and disconnect from the compute node.

Using the dbaascli Utility on Deployments Hosting an Oracle Data Guard Configuration of Single-Instance Databases

You can use the `dbaascli` utility to apply Linux OS security patches to the two compute nodes associated with a database deployment hosting an Oracle Data Guard configuration of single-instance databases. This procedure uses the Data Guard switchover operation to reduce database downtime.

> **Note:**
>
> Before you begin, review all steps in the procedure to understand how the compute nodes are rebooted during the patching process.

1. Perform the following steps on the compute node hosting the **standby database**:
   a. Connect as the **opc** user to the compute node.

      See Connecting to a Compute Node Through Secure Shell (SSH).

   b. Start a root-user command shell:

      ```
      $ sudo -s
      #
      ```

   c. Check whether any Linux OS security patches are available:

      ```
      # dbaascli patch os list
      ```

      
      This command is not yet available on Oracle Cloud at Customer. Instead, you must use the following command:

      ```
      # /var/opt/oracle/patch/dbpatchmsm -list_ospatches
      ```

      If the command response indicates that patches are available, continue following this procedure.

      Otherwise, the latest patches are already installed on the compute node. Exit the root-user command shell, disconnect from the compute node, and skip to Step 2 of this procedure.

   d. Apply the latest Linux OS security patches:

      ```
      # dbaascli patch os apply
      ```

      
      This command is not yet available on Oracle Cloud at Customer. Instead, you must use the following command:

      ```
      # /var/opt/oracle/patch/dbpatchmsm -apply_ospatch_async
      ```

   e. Switch to the **oracle** user and shut down the database:

```
# su - oracle
$ sqlplus '/ as sysdba'
...
SQL> shutdown
...
SQL> exit
```

f. Return to being the root user and reboot the compute node:

```
$ exit
# reboot
...
The system is going down for reboot NOW!
```

g. After the compute node and standby database have rebooted, allow a few minutes for redo data to be applied to the standby database.

To monitor the progress, connect as the **opc** user to the standby compute node, and run the following command:

```
$ dgmgrl show configuration verbose;
```

The operation is complete when the preceding command returns no errors.

2. Perform a switchover from the primary database to the standby database in your Oracle Data Guard configuration.

See Performing a Switchover Operation.

3. Perform step 1 on the compute node hosting the **new standby database** (this was the primary database before you performed the switchover).

4. Perform a switchover back to the original primary database in your Oracle Data Guard configuration.

See Performing a Switchover Operation.

Manually Applying Linux OS Security Patches

You can manually apply Linux OS security patches to compute nodes hosting Oracle Database Cloud Service database deployments.

> **Note:**
>
> Before you begin, review all steps in the procedure to understand that the compute node is rebooted at the end of the patching process.

1. Connect as the **opc** user to the compute node.

See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Shut down the database.

- If the compute node is associated with a database deployment hosting a Data Guard configuration, then enter the following command to shut down the database and the grid infrastructure:

  ```
  # crsctl stop crs -f
  ```

- Otherwise, shut down the database as follows:

  Switch to the **oracle** user:

  ```
  # su - oracle
  $
  ```

  Shut down the database:

  ```
  $ sqlplus '/ as sysdba'
  ...
  SQL> shutdown
  ...
  SQL> exit
  ```

  Return to being the root user:

  ```
  $ exit
  #
  ```

4. Install the `yum-plugin-security` package:

   ```
   # yum install yum-plugin-security
   ```

5. Update all packages to the latest versions that contain security patches:

   ```
   # yum --security update-minimal
   ```

6. Reboot the compute node:

   ```
   # reboot
   ...
   The system is going down for reboot NOW!
   ```

Administering Oracle REST Data Services (ORDS)

When a database deployment is created on Oracle Database Cloud Service, Oracle REST Data Services (formerly known as Oracle APEX Listener) is started.

> **Note:**
>
> This section does not apply to database deployments that use Oracle Real Application Clusters. Such deployments do not include Oracle REST Data Services.

Topics

Adding a Signed SSL Certificate to Oracle REST Data Services

You can add a signed SSL certificate to the ORDS environment on a Database Cloud Service database deployment running ORDS 3.0.5 or later.

Deployments created after early October 2016 (version 16.4.1) already have the necessary version of ORDS installed. To check your version of ORDS, connect to the deployment's compute node and enter this command:

```
$ java -jar /u01/app/oracle/product/ords/ords.war version
```

If you need to upgrade your version of ORDS, see Updating the Cloud Tooling by Using the dbaascli Utility.

Before You Begin

To add a signed SSL certificate, you must have the following:

- The SSL certificate file from the certificate provider. This is a `.crt` file.
- The private key file you gave to the certificate provider as part of your Certificate Signing Request (CSR). This is a `.der` or `.pem` file.

Procedure

1. Copy the certificate and private key files to the database deployment's compute node.

   Copy these files as the **`oracle`** user to the following locations:

   - Certificate file: `/u01/app/oracle/product/ords/certificate.crt`
   - Private key file: `/u01/app/oracle/product/ords/privkey.der` or `/u01/app/oracle/product/ords/privkey.pem`, depending on format.

   For instructions on copying files to the compute node, see Copying Files to or from a Database Cloud Service Database Deployment.

2. Connect as the **`oracle`** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

3. If your key file is in `.pem` format, convert it to `.der` format:

   ```
   $ cd /u01/app/oracle/product/ords
   $ /usr/bin/openssl pkcs8 -topk8 -inform PEM -outform DER -in privkey.pem -out
   privkey.der -nocrypt
   $ rm -f privkey.pem
   ```

4. Edit the `standalone.properties` file of your ORDS environment, adding these lines if missing:

   ```
   ssl.cert=/u01/app/oracle/product/ords/certificate.crt
   ssl.cert.key=/u01/app/oracle/product/ords/privkey.der
   ```

   This file is located at `/u01/app/oracle/product/ords/conf/ords/standalone/standalone.properties`.

5. Set the permissions on the certificate and private key files:

```
$ chown oracle:oinstall /u01/app/oracle/product/ords/certificate.crt
$ chown oracle:oinstall /u01/app/oracle/product/ords/privkey.der
$ chmod 400 /u01/app/oracle/product/ords/certificate.crt
$ chmod 400 /u01/app/oracle/product/ords/privkey.der
```

6. Restart ORDS:

```
$ /etc/init.d/ords restart
```

7. Close your connection to the compute node:

```
$ exit
```

Stopping Oracle REST Data Services

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Stop ORDS:

```
# /u01/app/oracle/product/ords/ords stop
INFO: Stopping Oracle REST Data Services...
INFO: Oracle REST Data Services stopped
```

4. Exit the root-user command shell and close your connection to the compute node:

```
# exit
$ exit
```

Starting Oracle REST Data Services

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Start ORDS:

```
# /u01/app/oracle/product/ords/ords start
INFO: Starting Oracle REST Data Services...
INFO: Oracle REST Data Services started with PID pid
```

4. Exit the root-user command shell and close your connection to the compute node:

```
# exit
$ exit
```

Loading Data into the Oracle Database on Database Cloud Service

You load data into an Oracle database on Oracle Database Cloud Service using the same tools you would use for an Oracle database on another system.

If you are using Database Cloud Service on Oracle Cloud, the location of the database in an Oracle data center does not place any special restrictions on data loading. However, transmission speeds across the Internet tend to be slower, sometimes much slower, than on internal networks, and you should factor this in when choosing any data loading approach.

The location of the database in an Oracle data center does not place any special restrictions on data loading. However, transmission speeds across the Internet tend to be slower, sometimes much slower, than on internal networks, and you should factor this in when choosing any data loading approach.

The following sections outline several common tools and techniques used to load data into an Oracle database. Also, see Migrating Oracle Databases to Database Cloud Service for additional techniques and more specific information about migrating existing Oracle databases to Database Cloud Service.

Using SQL*Loader to Load Data into the Database

SQL*Loader is a high-speed data loading utility that loads data from external files into tables in an Oracle database. SQL*Loader accepts input data in a variety of formats, can perform filtering, and can load data into multiple Oracle database tables during the same load session. SQL*Loader provides three methods for loading data: Conventional Path Load, Direct Path Load, and External Table Load.

For information, see "SQL Loader" in *Oracle Database Utilities* for Release 18, 12.2, 12.1 or 11.2.

Using Oracle Data Pump Import to Load Data into the Database

Oracle Data Pump is an Oracle Database feature that offers very fast bulk data and metadata movement between Oracle databases. Oracle Data Pump provides two high-speed, parallel utilities: Export (expdp) and Import (impdp). Data Pump automatically manages multiple, parallel streams for maximum throughput of unload and load operations. The degree of parallelism can be adjusted on-the-fly.

For information, see "Data Pump Import" in *Oracle Database Utilities* for Release 18, 12.2, 12.1 or 11.2.

Using Transportable Tablespaces to Load Data into the Database

Transportable Tablespaces is an Oracle Database feature that copies a set of tablespaces from one Oracle database to another. Moving data using transportable tablespaces can be much more efficient than performing either an export/import or unload/load of the same data. This is because the tablespace datafiles are copied to the destination location, which avoids the cost of formatting the data into Oracle blocks. Also, in some circumstances, your Transportable Tablespace can contain previously encrypted or compressed data, which avoids the cost of decrypting and re-encrypting, or expanding and re-compressing the data.

For information, see "Transporting Tablespaces Between Databases" in *Oracle Database Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2.

Using Pluggable Databases (PDBs) to Load Data into the Database

The multitenant architecture of Oracle Database 12c and later releases supports the moving of a pluggable database (PDB) from one container database (CDB) to another. This capability makes it easy to load data into Database Cloud Service, provided that the source data is already inside a PDB on Oracle Database 12c or a later release.

For information about PDBs and how to unplug, move, and plug them, see "Overview of Configuring and Managing a Multitenant Environment" in *Oracle Multitenant Administrator's Guide* for Release 18 or "Overview of Managing a Multitenant Environment" in *Oracle Database Administrator's Guide* for Release 12.2 or 12.1.

Tuning Oracle Database Performance on Database Cloud Service

You tune the performance of Oracle Database on Oracle Database Cloud Service using the same tools you would use for an Oracle database running on any system in your data center. Database Cloud Service does not place any special restrictions on performance tuning.

The *Oracle Database Performance Tuning Guide* for Release 18, 12.2, 12.1 or 11.2 provides extensive information about how to use Oracle Database performance tools to optimize database performance. It also describes performance best practices and includes performance-related reference information.

Additionally, the Enterprise Manager Tuning and Performance option packs are included in database deployments created using the High Performance and Extreme Performance software editions. These option packs provide several utilities to assist in maintaining performance and identifying and correcting performance issues.

If your performance tuning activities indicate that you need more computing power or more storage, you can scale Database Cloud Service to satisfy the need. See Scaling a Database Deployment.

Monitoring and Managing Oracle Database on Database Cloud Service

To monitor and manage the Oracle database deployed on Oracle Database Cloud Service, you can use the standard management tool provided with the version of the database:

- For Oracle Database 18c, use Enterprise Manager Database Express 18c. See Accessing Enterprise Manager Database Express 18c.
- For Oracle Database 12c, use Enterprise Manager Database Express 12c. See Accessing Enterprise Manager Database Express 12c.
- For Oracle Database 11g, use Enterprise Manager 11g Database Control. See Accessing Enterprise Manager 11g Database Control.

Beyond these tools provided with the database, Database Cloud Service Oracle SQL Developer Web in deployments hosting single-instance databases. SQL Developer Web supports monitoring and management both of Oracle Database and of computing resources. For more information, see Using Oracle SQL Developer Web in Database Cloud Service.

Managing the Log and Diagnostic Files on Database Cloud Service

The software components in Oracle Database Cloud Service generate a variety of log and diagnostic files, and not all these files are automatically archived and purged. Thus, managing the identification and removal of these files to avoid running out of file storage space is an important administrative task.

Database deployments that host single-instance databases include the `cleandblogs` script to simplify this administrative task. This script runs weekly as a `crontab` job to archive key files and remove old log and diagnostic files. It uses a configuration file named `cleandblogs.cfg` to determine how long to retain each kind of log or diagnostic file. You can edit this file to change the default retention periods. This file is located at `/var/opt/oracle/cleandb/cleandblogs.cfg`.

The following table lists the parameters that appear in the `cleandblogs.cfg` file, providing a description and the default retention period in days for each file type.

Parameter	Description and Default Value
`AlertRetention`	Alert log (`alert_`*`instance`*`.log`) retention value in days. Default value in file: 14
`ListenerRetention`	Listener log (`listener.log`) retention value in days. Default value in file: 14
`AuditRetentionDB`	Database audit (`*.aud`) retention value in days. Default value in file: 1
`CoreRetention`	Core dump/files (`*.cmdp*`) retention value in days. Default value in file: 7
`TraceRetention`	Trace file (`*.tr*` and `*.prf`) retention value in days. Default value in file: 7
`longpRetention`	Data designated in the Automatic Diagnostic Repository (ADR) as having a long life (the `LONGP_POLICY` attribute). For information about ADR, see "Automatic Diagnostic Repository (ADR)" in *Oracle Database Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2. Default value in file: 30
`shortpRetention`	Data designated in the Automatic Diagnostic Repository (ADR) as having a short life (the `SHORTP_POLICY` attribute). For information about ADR, see "Automatic Diagnostic Repository (ADR)" in *Oracle Database Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2. Default value in file: 7
`obkupLogRetention`	`obkup` log file retention in days. Default value in file: 30
`LogDirRetention`	cleandblogs log file retention in days. Default value in file: 14

Archiving Alert Logs and Listener Logs

When cleaning up alert and listener logs, `cleandblogs` first archives and compresses the logs, operating as follows:

1. The current log file is copied to an archive file that ends with a date stamp.
2. The current log file is emptied.
3. The archive file is compressed using `gzip`.
4. Any existing compressed archive files older than the retention period are deleted.

Running the cleandblogs Script Manually

The `cleandblogs` script automatically runs weekly, but you can also run the script manually if the need arises.

1. Connect as the **`oracle`** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Change to the directory containing the `cleandblogs` script:

   ```
   $ cd /var/opt/oracle/cleandb
   ```

3. Run the `cleandblogs` script:

   ```
   $ ./cleandblogs.pl
   ```

   When running the script manually, you can specify an alternate configuration file to use instead of `cleandblogs.cfg` by using the `--pfile` option:

   ```
   $ ./cleandblogs.pl --pfile config-file-name
   ```

4. Close your connection to the compute node:

   ```
   $ exit
   ```

5

Accessing Database Cloud Service

This section describes how to access tools, utilities and interfaces available in Oracle Database Cloud Service.

Topics

Connecting to a Compute Node Through Secure Shell (SSH)

To gain local access the tools, utilities and other resources on a compute node associated with Oracle Database Cloud Service, you use Secure Shell (SSH) client software to establish a secure connection and log in as the user `oracle` or the user `opc`.

Several SSH clients are freely available. The following sections show how to use SSH clients on UNIX, UNIX-like and Windows platforms to connect to a compute node associated with Database Cloud Service.

Connecting to a Compute Node Using the ssh Utility on UNIX and UNIX-Like Platforms

UNIX and UNIX-like platforms (including Solaris and Linux) include the ssh utility, an SSH client.

Before You Begin

Before you use the ssh utility to connect to a compute node, you need the following:

- The IP address of the compute node

  The IP address of a compute node associated with a database deployment on Oracle Database Cloud Service is listed on the Oracle Database Cloud Service Overview page. See Viewing Detailed Information for a Database Deployment.
- The SSH private key file that matches the public key associated with the deployment.

Procedure

To connect to a compute node using the ssh utility on UNIX and UNIX-like platforms:

1. In a command shell, set the file permissions of the private key file so that only you have access to it:

   ```
   $ chmod 600 private-key-file
   ```

   private-key-file is the path to the SSH private key file that matches the public key that is associated with the deployment.

2. Run the ssh utility:

   ```
   $ ssh -i private-key-file user-name@node-ip-address
   ```

   where:

   - ***private-key-file*** is the path to the SSH private key file.
   - ***user-name*** is the operating system user you want to connect as:
     - Connect as the user **oracle** to perform most operations; this user does not have root access to the compute node. On database deployments that use Oracle RAC, you cannot by default connect as the `oracle` user. You must add the public key to the `oracle` user's `$HOME/.ssh/authorized_keys` file to grant SSH access.
     - Connect as the user **opc** to perform operations that require root access to the compute node, such as backing up or patching; this user can use the `sudo` command to gain root access to the compute node.
   - ***node-ip-address*** is the IP address of the compute node in *x.x.x.x* format.

3. If this is the first time you are connecting to the compute node, the ssh utility prompts you to confirm the public key. In response to the prompt, enter **yes**.

Connecting to a Compute Node Using the PuTTY Program on Windows

PuTTY is a freely available SSH client program for Windows.

Before You Begin

Before you use the PuTTY program to connect to a compute node, you need the following:

- The IP address of the compute node

  The IP address of a compute node associated with a database deployment on Oracle Database Cloud Service is listed on the Oracle Database Cloud Service Overview page. See Viewing Detailed Information for a Database Deployment.

- The SSH private key file that matches the public key associated with the deployment. This private key file must be in the PuTTY .ppk format. If the private key file was originally created on the Linux platform, you can use the PuTTYgen program to convert it to the .ppk format.

Procedure

To connect to a compute node using the PuTTY program on Windows:

1. Download and install PuTTY.

   To download PuTTY, go to http://www.putty.org/ and click the **You can download PuTTY here** link.

2. Run the PuTTY program.

   The PuTTY Configuration window is displayed, showing the Session panel.

3. In **Host Name (or IP address)** box, enter the IP address of the compute node.

4. Confirm that the **Connection type** option is set to **SSH**.

5. In the Category tree, expand **Connection** if necessary and then click **Data**.

   The Data panel is displayed.

6. In **Auto-login username** box, enter the user you want to connect as:

   - Connect as the user **oracle** to perform most operations; this user does not have root access to the compute node. On database deployments that use Oracle RAC, you cannot by default connect as the **oracle** user. You must add the public key to the oracle user's `$HOME/.ssh/authorized_keys` file to grant SSH access.
   - Connect as the user **opc** to perform operations that require root access to the compute node, such as backing up or patching; this user can use the `sudo` command to gain root access to the compute node.

7. Confirm that the **When username is not specified** option is set to **Prompt**.

8. In the Category tree, expand **SSH** and then click **Auth**.

   The Auth panel is displayed.

9. Click the **Browse** button next to the **Private key file for authentication** box. Then, in the Select private key file window, navigate to and open the private key file that matches the public key that is associated with the deployment.

10. In the Category tree, click **Session**.

    The Session panel is displayed.

11. In the **Saved Sessions** box, enter a name for this connection configuration. Then, click **Save**.

12. Click **Open** to open the connection.

    The PuTTY Configuration window is closed and the PuTTY window is displayed.

13. If this is the first time you are connecting to the VM, the PuTTY Security Alert window is displayed, prompting you to confirm the public key. Click **Yes** to continue connecting.

Accessing Enterprise Manager Database Express 18c

Enterprise Manager Database Express (EM Express), a web-based tool for managing Oracle Database 18c, is available on Oracle Database Cloud Service database deployments created using Oracle Database 18c.

You can access EM Express in the following ways:

- Using the Open EM Console menu item
- Using a direct URL
- Using an SSH tunnel

Using the Open EM Console Menu Item to Access EM Express

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access EM Express is blocked by default. To use the **Open EM Console** menu item, you must unblock port 5500, either by enabling the deployment's **ora_p2_dbexpress** predefined access rule or by creating your own access rule that opens port 5500. For instructions, see Enabling Access to a Compute Node Port.

1. Open the Instances page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. From the action menu ( ) for the deployment, select **Open EM Console**.
3. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to ignore the warning and continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.
4. When prompted for a user name and password, enter the name of a user with the`DBA` privilege (such as `SYS` or `SYSTEM`) and the password.

   Enter a PDB name if you want to access a specific PDB or leave it blank to access the root container.

   If you want to connect with `SYSDBA` privileges, select **as SYSDBA**.

   After entering or selecting the required values, click **Login**.

   Note: if the database deployment was created using a QuickStart template, the password for `SYS` and `SYSTEM` is available in the zip file downloaded when the deployment was created.

The **Open EM Console** menu item is also available from the action menu ( ) on the Oracle Database Cloud Service Instance Overview page.

> **Note:**
>
> On database deployments hosting an Oracle RAC database, the **Open EM Console** menu item opens EM Express on node 1. To open EM Express on node 2, you must use a direct URL.

Using a Direct URL to Access EM Express

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access EM Express is blocked by default. To use a direct URL, you must unblock port 5500, either by enabling the deployment's **ora_p2_dbexpress** predefined access rule or by creating your own access rule that opens port 5500. For instructions, see Enabling Access to a Compute Node Port.

1. In your web browser, go to the following URL:

   `https://node-ip-address:5500/em`

   where *node-ip-address* is the IP address of the compute node as listed on the deployment's Overview page.
2. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to ignore the warning and continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.
3. When prompted for a user name and password, enter the name of a user with `DBA` privilege (such as `SYS` or `SYSTEM`) and the password.

   Enter a PDB name if you want to access a specific PDB or leave it blank to access the root container.

   If you want to connect with `SYSDBA` privileges, select **as SYSDBA**.

   After entering or selecting the required values, click **Login**.

   Note: if the database deployment was created using a QuickStart template, the password for `SYS` and `SYSTEM` is available in the zip file downloaded when the deployment was created.

Using an SSH Tunnel to Access EM Express

1. Create an SSH tunnel to port 5500 on the compute node hosting EM Express. For information about creating an SSH tunnel, see Creating an SSH Tunnel to a Compute Node Port.
2. After creating the SSH tunnel, direct your browser to the URL `https://localhost:5500/em`.

3. When prompted for a user name and password, enter the name of a user with DBA privilege (such as SYS or SYSTEM) and the password.

   Enter a PDB name if you want to access a specific PDB or leave it blank to access the root container.

   If you want to connect with SYSDBA privileges, select **as SYSDBA**.

   After entering or selecting the required values, click **Login**.

   Note: if the database deployment was created using a QuickStart template, the password for SYS and SYSTEM is available in the zip file downloaded when the deployment was created.

Accessing Enterprise Manager Database Express 12c

Enterprise Manager Database Express 12c (EM Express), a web-based tool for managing Oracle Database 12c, is available on Oracle Database Cloud Service database deployments created using Oracle Database 12c Release 1 (12.1) or Oracle Database 12c Release 2 (12.2).

Before you access EM Express to manage your database you must determine, and in some cases configure, the network port that is used to access EM Express as follows:

- **To manage the CDB.** When a database deployment is created, Database Cloud Service automatically sets port 5500 on the deployment's compute nodes for EM Express access to the CDB. You do not need to perform any manual configuration steps.
- **To manage a PDB with Oracle Database 12c Release 1 (version 12.1).** For a version 12.1 database deployment, you must manually set a port for each PDB you want to manage using EM Express. See Setting the Port for EM Express to Manage a PDB.
- **To manage a PDB with Oracle Database 12c Release 2 (version 12.2).** With Oracle Database 12c Release 2, EM Express can be configured to access the CDB and all PDBs on a single port, which is known as the global port. For version 12.2 database deployments created after early December 2016, the global port is set by default. For deployments created prior to December 2016, see Setting the Global Port for EM Express to Manage a CDB and the PDBs (Oracle Database 12.2 Only).

> **Note:**
>
> To confirm the port that is in use for a specific database, connect to the database as a database administrator and execute the query shown in the following example:
>
> ```
> SQL> select dbms_xdb_config.getHttpsPort() from dual;
>
> DBMS_XDB_CONFIG.GETHTTPSPORT()
> ------------------------------
>                             5502
> ```

After you determine the EM Express port for the CDB or PDB that you want to manage, you must choose one of the following options to access EM Express:

- **Unblock the port.** You can unblock the port by Enabling Access to a Compute Node Port.

  After unblocking the port, you can access EM Express on that port as described in Accessing EM Express Using the EM Express Port.

- **Leave the port blocked.** If your security requirements demand that you leave the port blocked, you can still access EM Express by connecting to it through an SSH tunnel, as described in Accessing EM Express Using an SSH Tunnel.

Setting the Port for EM Express to Manage a PDB

In Oracle Database 12c Release 1, a unique HTTPS port must be configured for the root container (CDB) and each PDB that you manage using EM Express.

To configure a HTTPS port so that you can manage a PDB with EM Express:

1. Invoke SQL*Plus and log in to the PDB as the `SYS` user with `SYSDBA` privileges.
2. Execute the `DBMS_XDB_CONFIG.SETHTTPSPORT` procedure.

   ```
   SQL> exec dbms_xdb_config.sethttpsport(port-number)
   ```

Setting the Global Port for EM Express to Manage a CDB and the PDBs (Oracle Database 12.2 Only)

In Oracle Database 12c Release 2, you can configure a single port (known as the global port), which enables you to use EM Express to connect to all of the PDBs in the CDB using the HTTPS port for the CDB.

In database deployments created after early December 2016, the global port is set by default.

To configure the global port in deployments created before December 2016:

1. Invoke SQL*Plus and log in to the root container (CDB) as the `SYS` user with `SYSDBA` privileges.
2. Execute the `DBMS_XDB_CONFIG.SETGLOBALPORTENABLED` procedure.

   ```
   SQL> exec dbms_xdb_config.SetGlobalPortEnabled(TRUE)
   ```

Accessing EM Express Using the EM Express Port

If the EM Express port is not blocked, you can access EM Express by directing your browser to the URL `https://node-ip-address:EM-Express-port/em`, where *`node-ip-address`* is the public IP address of the compute node hosting EM Express, and *`EM-Express-port`* is the EM Express port used by the database.

You can also access EM Express to manage the CDB in 12.1 or the root container and PDBs through the global port in 12.2 through the Oracle Database Cloud Service console:

1. Open the Instances page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. From the action menu ( ) for the deployment, select **Open EM Console**.

The EM Express login page is displayed.

3. Enter the name of a user with the `DBA` privilege (such as `SYS` or `SYSTEM`) and the password. To connect with `SYSDBA` privileges, select **as sysdba**. Then click **Login**.

This option is also available from the action menu ( ) on the Oracle Database Cloud Service Instance Overview page.

> **Note:**
>
> On a database deployment hosting an Oracle RAC database, the link **Open EM Console** opens EM Express on node 1. To open EM Express on node 2, direct your browser to the URL `https://`*`node2-ip-address`*`:`*`EM-Express-port`*`/em`, where *`node2-ip-address`* is the public IP address of node 2, and *`EM-Express-port`* is the EM Express port used by the database.

Accessing EM Express Using an SSH Tunnel

To access EM Express when its port is blocked, you must create an SSH tunnel to the EM Express port on the compute node hosting EM Express. See Creating an SSH Tunnel to a Compute Node Port.

After the SSH tunnel is created, you can access EM Express by directing your browser to the URL `https://localhost:`*`EM-Express-port`*`/em`.

After the EM Express login page is displayed, enter the name of a user with the `DBA` privilege (such as `SYS` or `SYSTEM`) and the password. To connect with `SYSDBA` privileges, select **as sysdba**. Then click **Login**.

Accessing Enterprise Manager 11g Database Control

Enterprise Manager 11g Database Control (Database Control), a web-based tool for managing Oracle Database 11g, is available on Oracle Database Cloud Service database deployments created using Oracle Database 11g Release 2.

You can access Database Control in the following ways:

- Using the Open EM Console menu item
- Using a direct URL
- Using an SSH tunnel

Using the Open EM Console Menu Item to Access Database Control

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access Database Control is blocked by default. To use the **Open EM Console** menu item, you must unblock port 1158, either by enabling the deployment's **ora_p2_dbconsole** predefined access rule or by creating your own access rule that opens port 1158. For instructions, see Enabling Access to a Compute Node Port.

1. Open the Instances page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. From the action menu ( ) for the deployment, select **Open EM Console**.
3. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to ignore the warning and continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.
4. When prompted for a user name and password, enter the name of a user with the`DBA` privilege (such as `SYS` or `SYSTEM`) and the password.

   If you want to connect with `SYSDBA` privileges, select **as SYSDBA**.

   After entering or selecting the required values, click **Login**.

   Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

The **Open EM Console** menu item is also available from the action menu ( ) on the Oracle Database Cloud Service Instance Overview page.

> **Note:**
>
> On database deployments hosting an Oracle RAC database, the **Open EM Console** menu item opens Database Control on node 1. To open Database Control on node 2, you must use a direct URL.

Using a Direct URL to Access Database Control

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access Database Control is blocked by default. To use a direct URL, you must unblock port 1158, either by enabling the deployment's **ora_p2_dbconsole** predefined access rule or by creating your own access rule that opens port 1158. For instructions, see Enabling Access to a Compute Node Port.

1. In your web browser, go to the following URL:

   `https://node-ip-address:1158/em`

   where *node-ip-address* is the IP address of the compute node as listed on the deployment's Overview page.

2. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to ignore the warning and continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.

3. When prompted for a user name and password, enter the name of a user with the`DBA` privilege (such as `SYS` or `SYSTEM`) and the password.

   If you want to connect with `SYSDBA` privileges, select **as SYSDBA**.

   After entering or selecting the required values, click **Login**.

   Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

Using an SSH Tunnel to Access Database Control

1. Create an SSH tunnel to port 1158 on the compute node hosting Database Control. For information about creating an SSH tunnel, see Creating an SSH Tunnel to a Compute Node Port.

2. After creating the SSH tunnel, direct your browser to the URL `https://localhost:1158/em`.

3. When prompted for a user name and password, enter the name of a user with the`DBA` privilege (such as `SYS` or `SYSTEM`) and the password.

   If you want to connect with `SYSDBA` privileges, select **as SYSDBA**.

   After entering or selecting the required values, click **Login**.

   Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

Connecting Remotely to the Database by Using Oracle SQL Developer

You can define an Oracle SQL Developer connection to your database in the cloud to perform operations as you would with an on-premises database.

How you make a SQL Developer connection to the Oracle Database depends on whether the Oracle Net Listener port has been unblocked. When a Database Cloud Service database deployment is created on Oracle Cloud Infrastructure Classic, the Oracle Net Listener port is blocked to ensure network security. For information about unblocking a port, see Enabling Access to a Compute Node Port.

Before You Begin

Before you use Oracle SQL Developer to connect to a database deployment, you need the following:

- The IP address of the compute node

  The IP address of a compute node associated with a database deployment on Oracle Database Cloud Service is listed on the Oracle Database Cloud Service Overview page. For instructions to display this page, see Viewing Detailed Information for a Database Deployment.
- The SSH private key file that matches the public key associated with the deployment.

Creating a SQL Developer Connection When the Listener Port Is Unblocked

To create a SQL Developer connection to a database deployment when the Oracle Net Listener port is unblocked:

1. Open SQL Developer. Right-click Connections and select **New Connection**.

   > **Note:**
   >
   > If you are using a version of SQL Developer in which the Connections panel shows both "Connections" and "Cloud Connections", right-click **Connections**. Do not right-click Cloud Connections, which is for Oracle Database Exadata Express Cloud Service.

   The New / Select Database Connection dialog appears.
2. Provide the following information and then click **Test**.
   - **Connection Name**: Create a name for this connection.
   - **Username**: Name of the database user for the connection. This user must have sufficient privileges to perform the tasks that you want to perform while connected to the database, such as creating, editing, and deleting tables, views, and other objects.
   - **Password**: Provide the "Administration" password that you specified when you created the database deployment.

- **Hostname**: Provide the Public IP address for the database deployment compute node you are connecting to.
- **Port**: Provide the listener port number that you specified when you created the database deployment.
- **SID** or **Service Name**: If you are connecting to Oracle Database 11g (non-CDB) or Oracle Database 12c or later (CDB), provide the SID. If you are connecting to an Oracle Database 12c or later pluggable database (PDB), provide the service name instead of the SID.

3. If your test results show success, click **Connect**. You have connected SQL Developer to your database deployment in Oracle Database Enterprise Cloud Service. Now you can use SQL Developer as you normally would with an on-premises database.

Creating a SQL Developer Connection When the Listener Port Is Blocked

If the listener port has not been unblocked by enabling the `ora_p2_dblistener` access rule, you can define an SSH connection in Oracle SQL Developer 4.0.3 or later, with functionality to connect to a database through port forwarding. In that case, you will not need to follow the instructions in Enabling Access to a Compute Node Port.

1. From the View menu, select **SSH**.
2. In the SSH Hosts navigation panel, right click SSH Hosts and select **New SSH Host**.
3. In the New SSH Host dialog:
   - Enter a name for the SSH Host.
   - In the Host field, enter the IP address of your database deployment.
   - In the Username field, enter `oracle` or `opc`.
   - Check **Use key file**, and browse for your private SSH key file.
   - Select **Add a Local Port Forward**.
   - Leave the Name field as `Default`. Leave the Host field as `localhost`. Set the Port field to the listener port number that you specified when you created the database deployment. Keep the default of `Automatically assign local port`.
   - Click **OK**.

Connecting Remotely to the Database by Using Oracle Net Services

> **Note:**
>
> How you connect to an Oracle RAC database on Database Cloud Service differs from how you connect to a non-clustered database. For information, see Creating an Oracle Net Connection to an Oracle RAC Database.

Oracle Database Cloud Service support access to Oracle Database on the standard Oracle Net Listener port.

How you make a Oracle Net connection to the Oracle Database depends on whether the Oracle Net Listener port has been unblocked. When a Database Cloud Service database deployment is created on Oracle Cloud Infrastructure Classic, the Oracle Net Listener port is blocked to ensure network security. For information about unblocking a port, see Enabling Access to a Compute Node Port.

Before You Can Connect

In order to make a remote database connection by using Oracle Net Services, you require pieces of information:

- The IP address for the compute node that you wish to connect to. You can obtain this information by viewing details as described in Viewing Detailed Information for a Database Deployment.
- The database identifier, either the database SID or service name. For database deployments running Oracle Database 11g, you can identify the database by using the SID. For deployments running Oracle Database 12c or later, connecting to the database by specifying the database SID connects you to the CDB (container database). To connect to a PDB (pluggable database), specify the service name of the pluggable database. In Database Cloud Service, the format of a database service name is:

  pdb.*identity-domain*.oraclecloud.internal

  where *pdb* is the name of the PDB and *identity-domain* is the name of the identity domain housing your Database Cloud Service subscription; for example:

  PDB1.usexample5822.oraclecloud.internal

  You can obtain the required information by viewing details as described in Viewing Detailed Information for a Database Deployment.

Creating an Oracle Net Connection When the Listener Port Is Unblocked

To create an Oracle Net connection when the listener port is unblocked, you can use the easy connect method to specify a connect identifier with the following format:

node-ip-address:*listener-port-number*/*sid-or-service-name*

For example:

198.51.100.101:1521/ORCL

or

198.51.100.102:1521/PDB1.usexample5822.oraclecloud.internal

Creating an Oracle Net Connection When the Listener Port Is Blocked

To create an Oracle Net connection when the listener port is blocked, you must create an SSH tunnel from your client (localhost) to the port of the compute node hosting the Oracle Net Listener. For information about creating an SSH tunnel, see Creating an SSH Tunnel to a Compute Node Port.

To create an Oracle Net connection that uses the SSH tunnel, you can use the easy connect method to specify a connect identifier with the following format:

```
localhost:listener-port-number/sid-or-service-name
```

For example:

```
localhost:1521/ORCL
```

or

```
localhost:1521/PDB1.usexample5822.oraclecloud.internal
```

> **Note:**
>
> Some database access products, such as Oracle SQL Developer 4.0.3 or later, include functionality to connect to a database through an SSH tunnel. When using these products, you create the SSH tunnel to the listener port within the product and do not need to follow the instructions in Creating an SSH Tunnel to a Compute Node Port.

Creating an Oracle Net Connection to an Oracle RAC Database

On an Oracle RAC database on Database Cloud Service, the SCAN listeners on each compute node listen on the Oracle Net Services port. By default, this port and the port for the DB listener are blocked on the compute nodes. Therefore, you must first open access to these ports by enabling the **ora_p2_scan_listener** and **ora_p2_db_listener** security rules using the instructions Enabling Port Access by Enabling an Automatically Created Access Rule. Then, on the client, specify a connect descriptor that references the SCAN listeners on both compute nodes; for example:

```
alias-name = (DESCRIPTION =
  (ENABLE = BROKEN)
  (ADDRESS_LIST =
    (ADDRESS = (PROTOCOL = TCP)(HOST = node1-ip-address)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = node2-ip-address)(PORT = 1521))
  )
  (CONNECT_DATA = (SERVICE_NAME = service-name) )
)
```

where:

- *alias-name* is the name you use to identify the alias.
- *node1-ip-address* and *node2-ip-address* are the public IP addresses of the two compute nodes associated with the database deployment.
- *service-name* is the service name in the form:

  ```
  pdb.identity-domain.oraclecloud.internal
  ```

  For example:

  ```
  PDB1.usexample5822.oraclecloud.internal
  ```

Copying Files to or from a Database Cloud Service Database Deployment

To copy files to or from a database deployment, connect to a compute node of the deployment using a secure copy utility that supports key-based, passwordless authentication. Examples of such utilities are `scp` and `sftp` on Linux and WinSCP and FileZilla on Windows.

When connecting, provide the private key file that matches the public key provided when the database deployment was created, and specify the user as `opc` or `oracle`. Do not attempt to connect as an Oracle Cloud SFTP user. The compute nodes of Database Cloud Service database deployments are not accessible to Oracle Cloud SFTP users.

6

Backing Up and Restoring Databases on Database Cloud Service

This section explains how to back up and restore Oracle databases on Oracle Database Cloud Service.

Topics

About Backing Up Database Deployments on Database Cloud Service

By backing up your Oracle Database Cloud Service database deployments, you can protect the software, configuration and database against loss if a failure occurs. By restoring from a backup, you can restore the deployment's software, configuration, and database to their state at the time of the backup.

Database Cloud Service provides a backup feature that backs up:

- The database
- Database configuration files
- Grid Infrastructure configuration files (on deployments hosting an Oracle RAC database)
- Important system and cloud tooling files

To provide this backup feature, Database Cloud Service relies on system utilities, Oracle Database utilities, and Oracle Database Backup Cloud Service, all of which are installed in the database deployment.

Default Backup Configuration

When you create a database deployment, you choose one of the following backup destinations:

- **Both Cloud Storage and Local Storage.** Backups are configured to be created automatically and stored both on local compute node storage and on an Oracle Storage Cloud Service container.
- **Cloud Storage Only.** Backups are configured to be created automatically and stored on an Oracle Storage Cloud Service container.

> **Note:**
>
> This choice is not currently available for database deployments that use Oracle Real Application Clusters (Oracle RAC).

- **None.** No backup configuration is created.

If you want backups of your database, Oracle recommends that you choose one of the automatic backup configurations instead of manually creating backups.

The backup configuration created when you choose a destination other than **None** follows a set of Oracle best-practice guidelines:

- Full (level 0) backup of the database followed by rolling incremental (level 1) backups on a seven-day cycle (a 30-day cycle for the **Cloud Storage Only** destination)
- Full backup of selected database configuration files
- Full backup of selected system files
- Automatic backups daily at a time between 11 PM (23:00) and 3 AM (03:00), with the specific time set during the database deployment creation process
- Retention period:
  - **Both Cloud Storage and Local Storage**: 30 days, with the 7 most recent days' backups available on local storage
  - **Cloud Storage Only**: 30 days
- Encryption:
  - **Both Cloud Storage and Local Storage**: All backups to cloud storage are encrypted; backups of Enterprise Edition databases to local storage are encrypted; backups of Standard Edition databases to local storage are not encrypted.
  - **Cloud Storage Only**: All backups to cloud storage are encrypted.

If the defaults do not suit your needs, you can customize the backup configuration for your database deployment. For information, see Customizing the Current Backup Configuration.

You can also change the entire backup configuration from the one used when your deployment was created. For information, see Enabling and Reconfiguring the Automatic Backups Feature.

About Local Storage for Backups

When a database deployment is created on Database Cloud Service, Oracle Compute Cloud Service storage volumes are created and associated with the compute nodes. The storage volume reserved for backups is named `fra`. On deployments hosting Oracle RAC databases, this storage volume is shared by the compute nodes.

You can see details about `fra` (as well as other storage volumes) in the Compute Cloud Service console. See Viewing Details of a Storage Volume in *Using Oracle Cloud Infrastructure Compute Classic*.

The space allocated for the local storage used for backups on `fra` is 1.7 times the space allocated for data storage. For example, if you chose 100 GB for Usable Database Storage when creating the database deployment, 170 GB are allocated for backups.

Viewing Backup Configuration Information

How you view information about the current backup configuration of a database deployment depends on the type of database running in the deployment.

- For database deployments hosting single-instance databases, you use the `bkup_api` utility. See Viewing Backup Configuration Information by Using the bkup_api Utility.
- For database deployments hosting Oracle Real Application Clusters (RAC) databases, you use the `raccli` utility. See Viewing Backup Configuration Information by Using the raccli Utility.

Viewing Backup Configuration Information by Using the bkup_api Utility


This topic does not apply to Oracle Cloud at Customer.

You use the `get_config_info` command of the `bkup_api` utility to view backup configuration settings for database deployments hosting a single-instance database. Optionally, the output can be used to create a file containing JSON syntax.

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Use the `get_config_info` subcommand to display information about the current backup configuration:

   ```
   # /var/opt/oracle/bkup_api/bkup_api get_config_info --all --dbname dbname [--
   json json_destination]
   ```

   where *dbname* is the database name and *json_destination* is the name of a file to be generated containing JSON syntax.

4. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Viewing Backup Configuration Information by Using the raccli Utility


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You use the `list backupconfig` command of the `raccli` utility to view backup configuration settings for database deployments hosting an Oracle Real Application Clusters (RAC) database.

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Use the `list backupconfig` subcommand to display information about the current backup configuration:

```
$ raccli list backupconfig
```

3. Disconnect from the compute node:

```
$ exit
```

Creating an On-Demand Backup

You can create an on-demand backup of an Oracle Database Cloud Service database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Create an On-Demand Backup at the end of this topic.

Creating an On-Demand Backup by Using the Oracle Database Cloud Service Console

1. Open the Instances page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. Click the database deployment for which you want to create a backup.

   The Oracle Database Cloud Service Overview page is displayed.

3. Click the Administration tile.

   The Oracle Database Cloud Service Backup page is displayed.

4. Click **Backup Now**.

   The Backup Now dialog is displayed.

5. Make a selection for the **Keep Forever** option and then click **Backup**.

   The **Keep Forever** option controls the backup retention policy, as follows:

   - **No** — specifies that the backup is produced and maintained in accordance with the automatic backup retention policy that is associated with the database deployment.

- **Yes** — specifies that the backup is a long-term backup, which is produced and maintained independently of the automatic backup retention policy that is associated with the database deployment. Long-term backups remain until you explicitly remove them from the system.

Other Ways to Create an On-Demand Backup

- For database deployments hosting single-instance databases, you can use the `bkup_api` utility. See Creating an On-Demand Backup by Using the bkup_api Utility.
- For database deployments hosting Oracle Real Application Clusters (RAC) databases, you can use the `raccli` utility. See Creating an On-Demand Backup by Using the raccli Utility.

Creating an On-Demand Backup by Using the bkup_api Utility

You can use the `bkup_api` utility to create an on-demand backup of a database deployment hosting a single-instance database or an Oracle Data Guard configuration of two single-instance databases.

1. Connect as the **opc** user to the compute node. In a Data Guard configuration, connect to the compute node hosting the primary database.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Enter the `bkup_api` command:

   - To create a backup that follows the current retention policy, enter the following `bkup_api` command:

     ```
     # /var/opt/oracle/bkup_api/bkup_api bkup_start
     ```

   - To create a long-term backup of the complete database that persists until you delete it, enter the following `bkup_api` command:

     ```
     # /var/opt/oracle/bkup_api/bkup_api bkup_start --keep
     ```

     > **Note:**
     >
     > By default, the backup is given a timestamp-based tag. To specify a custom backup tag, add the `--tag` option to the `bkup_api` command. For example, to create a long-term backup with the tag `monthly`, enter the following command:
     >
     > ```
     > # /var/opt/oracle/bkup_api/bkup_api bkup_start --keep --tag=monthly
     > ```

4. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

> **Note:**
>
> After you enter a `bkup_api bkup_start` command, the `bkup_api` utility starts the backup process, which runs in the background. To check the progress of the backup process, enter the following `bkup_api` command:
>
> ```
> # /var/opt/oracle/bkup_api/bkup_api bkup_status
> ```

Creating an On-Demand Backup by Using the raccli Utility


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can use the `raccli` utility to create an on-demand backup of a database deployment hosting an Oracle Real Application Clusters (RAC) database:

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Enter the `raccli create backup` subcommand:

   ```
   $ raccli create backup
   $
   ```

By default, the backup is given a timestamp-based tag. To specify a custom backup tag, add the `-tag` option to the `raccli` command; for example, to create a long-term backup with the tag "monthly", enter the following command:

```
$ raccli create backup -tag monthly
```

After you enter a `raccli create backup` command, the `raccli` utility starts the backup process, which runs in the background. To check the progress of the backup process, enter the following `raccli` command:

```
$ raccli describe job
```

Deleting a Backup

You can delete long-term backups created using the `bkup_api` utility with the `--keep` option.

You cannot delete backups that are associated with the automatic backup configuration, whether they were created using the `bkup_api` utility, the `raccli` utility, or the Oracle Database Cloud Service console. These backups are deleted automatically based on the retention period that is associated with the automatic backup configuration.

To delete a long-term backup of a database deployment on Oracle Database Cloud Service:

1. Connect to the compute node as the **opc** user.

For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. List the available long-term backups:

```
# /var/opt/oracle/bkup_api/bkup_api list --keep
```

A list of available backups is displayed. Note the tag of the backup that you want to delete.

4. Delete the backup:

```
# /var/opt/oracle/bkup_api/bkup_api bkup_delete --bkup=backup-tag
```

where ***backup-tag*** is the tag of the backup you want to delete.

5. Exit the root-user command shell:

```
# exit
$
```

Updating the Password for Backing Up to the Storage Cloud

Whenever the password is changed for an Oracle Cloud user whose credentials are used for backing up to an Oracle Storage Cloud container, you need to update the user's password in the Oracle Wallet file that maintains the credentials.

Because Oracle Cloud requires users to change their passwords on a regular basis, you need to perform this task regularly.

You can update the password by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Update the Password at the end of this topic.

Updating the Password by Using the Oracle Database Cloud Service Console

> **Note:**
>
> Currently, you cannot use the Oracle Database Cloud Service console to update the password on database deployments hosting an Oracle Real Application Clusters (RAC) database. Instead you must use the `raccli` utility. See Updating the Password by Using the raccli Utility.

1. Go to the Backup page of the deployment whose backup credentials you want to update:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the name of the database deployment whose backup credentials you want to update.

The Overview page for the deployment is displayed.

c. Click the Administration tile.

The Backup page for the deployment is displayed.

2. Click **Configure Backups**.

The Configure Backups window is displayed.

3. Enter the Cloud user name and new password.
4. Click **Save** and then confirm the operation.

Other Ways to Update the Password

- For database deployments hosting single-instance databases, you can use the `bkup_api` utility. See Updating the Password by Using the bkup_api Utility.
- For database deployments hosting Oracle Real Application Clusters (RAC) databases, you can use the `raccli` utility. See Updating the Password by Using the raccli Utility.

Updating the Password by Using the bkup_api Utility

You use the `bkup_api` utility to update the Oracle Wallet file containing the backup user's password:

1. Connect as the **opc** user to the compute node.

For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Enter this `bkup_api` command to generate a file containing the current backup settings:

```
# /var/opt/oracle/bkup_api/bkup_api get config [--file=filename] --dbname=dbname
```

where ***filename*** is an optional parameter used to specify a name for the file that will be generated and ***dbname*** is the database name for the database that you want to act on.

4. Edit the generated file, setting the value of the `bkup_oss_passwd` parameter to the new password.
5. Enter this `bkup_api` command to update the backup settings using the file you generated:

```
# /var/opt/oracle/bkup_api/bkup_api set config --file=filename --dbname=dbname
```

where ***filename*** is used to specify a name for the file that will be used to update the backup settings and ***dbname*** is the database name for the database that you want to act on.

6. You can use this `bkup_api` command to check the status of the update:

```
# /var/opt/oracle/bkup_api/bkup_api configure_status
```

7. Exit the root-user command shell:

```
# exit
$
```

Any changes you make by using the `bkup_api` command are not reflected in the Oracle Database Cloud Service console.

Updating the Password by Using the raccli Utility

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

On database deployments hosting an Oracle Real Application Clusters (RAC) database you use the `raccli` utility to update the Oracle Wallet file containing the backup user's password:

> **Note:**
>
> If you have used the `update rdk` subcommand of the `raccli` utility to update the cloud tooling to 16.4.5 or later, you must manually update the opc installer for the Oracle Database Cloud Backup Module before you use the `update backupconfig` subcommand. For instructions, see in Updating the cloud tooling on a deployment hosting Oracle RAC requires manual update of the Oracle Database Cloud Backup Module in *Known Issues for Oracle Database Cloud Service*.

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Update the password:

   ```
   $ raccli update backupconfig -params '{"cloudStorageUser":"user-
   name","cloudStoragePwd":"new-password"}'
   ```

   where ***user-name*** is the user name of the Oracle Cloud user whose credentials are used to access the Storage Cloud Service container and ***new-password*** is this user's new password.

   If the update succeeds, output from the command indicates that the wallet was successfully created and the credentials are valid. If you could not authenticate, you entered the wrong password and need to try again with the correct credentials.

Customizing the Current Backup Configuration

How you customize the current backup configuration depends on the type of database running in the database deployment.

Topics

- Customizing the Current Backup Configuration on Database Deployments Hosting Single-Instance Databases

- Customizing the Current Backup Configuration on Database Deployments Hosting Oracle RAC Databases

Customizing the Current Backup Configuration on Database Deployments Hosting Single-Instance Databases

You can customize many of the characteristics of the automatic backup configuration.

Topics

- Customizing Backup Settings by Using a Generated Configuration File
- Customizing Which System Files Are Backed Up
- Customizing Which Database Configuration Files Are Backed Up

Customizing Backup Settings by Using a Generated Configuration File

You can customize backup settings by generating a file containing the current settings, editing the file, and then using the file to update the backup settings. To generate a configuration file with the current backup settings and use it to update the settings:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Enter this `bkup_api` command to generate a file containing the current backup settings:

   ```
   # /var/opt/oracle/bkup_api/bkup_api get config [--file=filename] --dbname=dbname
   ```

   where ***filename*** is an optional parameter used to specify a name for the file that will be generated and ***dbname*** is the database name for the database that you want to act on.

4. Edit the parameter values in the generated file to change any settings you want to update in the current backup configuration. The following parameters are used to customize the current backup configuration:

Parameter	Description
`bkup_archlog_frequency`	Frequency of archivelog file backups expressed in hours.
`bkup_cfg_files`	Enable backup of configuration files. Valid values are `yes` and `no`.
`bkup_daily_time`	Time of the daily incremental backup expressed as `hh:mm`.
`bkup_disk_recovery_window`	Recovery window expressed in number of days between 1 and 14.
`bkup_oss_l0_day`	Day of Oracle Storage Cloud level 0 backup. Valid values are `mon`, `tue`, `wed`, `thu`, `fri`, `sat`, `sun`. Only applicable when `bkup_oss` is set to `yes`.

Parameter	Description
`bkup_oss_recovery_window`	Recovery window for backups to an Oracle Storage Cloud container, expressed in number of days between 1 and 30. Only applicable when `bkup_oss` is set to `yes`. Only applicable when `bkup_oss` is set to `yes`.

5. Enter this `bkup_api` command to update the backup settings using the file you generated:

   ```
   # /var/opt/oracle/bkup_api/bkup_api set config --file=filename --dbname=dbname
   ```

   where ***filename*** is used to specify a name for the file that will be used to update the backup settings and ***dbname*** is the database name for the database that you want to act on.

6. You can use this `bkup_api` command to check the status of the update:

   ```
   # /var/opt/oracle/bkup_api/bkup_api configure_status
   ```

7. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Any changes you make by using the `bkup_api` command are not reflected in the Oracle Database Cloud Service console.

Customizing Which System Files Are Backed Up

To change which system files and directories are backed up:

1. Connect as the **`oracle`** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Edit the contents of the `/home/oracle/bkup/oscfg.spec` file.

   The backup feature provided by Oracle Database Cloud Service backs up the files and folders listed in this specification file.

   An example of an `oscfg.spec` file with a default configuration is as follows:

   ```
   ## OS Configuration Files
   #
   # Doc Spec
   oscfg.spec
   #
   # Directories
   /etc/rc.d
   /home/oracle/bkup
   #
   # Single files
   /home/oracle/.bashrc
   /etc/crontab
   /etc/sysctl.conf
   /etc/passwd
   /etc/group
   /etc/oraInst.loc
   ```

```
/etc/oratab
/etc/fstab
```

Customizing Which Database Configuration Files Are Backed Up

To change which database configuration files are backed up:

1. Connect as the **`oracle`** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Edit the contents of the `/home/oracle/bkup/dbcfg.spec` file:

   The backup feature provided by Oracle Database Cloud Service backs up the files and folders listed in this specification file.

   An example of a `dbcfg.spec` file with a default configuration is as follows:

```
### Oracle_Home configuration files.
#
# Doc Spec
dbcfg.spec
# DB id
dbid
#
# Directories
/u01/app/oracle/product/12.1.0/dbhome_1/admin/ORCL/xdb_wallet
/u01/app/oracle/admin/ORCL/xdb_wallet
/u01/app/oracle/admin/ORCL/opc_wallet
# Note: tde_wallet must be backed up in a different location than DATA
bkup.
/u01/app/oracle/admin/ORCL/tde_wallet
/u01/app/oracle/admin/ORCL/cat_wallet
#/u01/app/oracle/product/12.1.0/dbhome_1/dbs
#/u01/app/oracle/product/12.1.0/dbhome_1/network/admin
#/u01/app/oraInventory
#
# Single files
/u01/app/oracle/product/12.1.0/dbhome_1/dbs/opcORCL.ora
/u01/app/oracle/product/12.1.0/dbhome_1/dbs/orapworcl
/u01/app/oracle/product/12.1.0/dbhome_1/network/admin/listener.ora
/u01/app/oracle/product/12.1.0/dbhome_1/network/admin/sqlnet.ora
/u01/app/oracle/product/12.1.0/dbhome_1/network/admin/tnsnames.ora
/u01/app/oracle/product/12.1.0/dbhome_1/rdbms/lib/env_rdbms.mk
/u01/app/oracle/product/12.1.0/dbhome_1/rdbms/lib/ins_rdbms.mk
```

Customizing the Current Backup Configuration on Database Deployments Hosting Oracle RAC Databases


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can customize many of the characteristics of the backup configuration.

Topics

Customizing Which System Files Are Backed Up

To change which system files and directories are backed up:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Edit the contents of the `/opt/oracle/dcs/rdbaas/config/oscfg.spec` file.

   The backup feature provided by Oracle Database Cloud Service backs up the files and folders listed in this specification file.

Customizing Which Database Configuration Files Are Backed Up

To change which database configuration files are backed up:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Edit the contents of the `/opt/oracle/dcs/rdbaas/config/dbcfg.spec` file.

   The backup feature provided by Oracle Database Cloud Service backs up the files and folders listed in this specification file.

Customizing Which Grid Infrastructure Configuration Files Are Backed Up

To change which grid infrastructure configuration files are backed up:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Edit the contents of the `/opt/oracle/dcs/rdbaas/config/gicfg.spec` file.

   The backup feature provided by Oracle Database Cloud Service backs up the files and folders listed in this specification file.

Customizing the Recovery Window for Backups to Local Storage

> **Note:**
>
> If you have used the `update rdk` subcommand of the `raccli` utility to update the cloud tooling to 16.4.5 or later, you must manually update the opc installer for the Oracle Database Cloud Backup Module before you use the `update backupconfig` subcommand. For instructions, see in Updating the cloud tooling on a deployment hosting Oracle RAC requires manual update of the Oracle Database Cloud Backup Module in *Known Issues for Oracle Database Cloud Service*.

To change the recovery window for backups to local storage:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Enter this `raccli` command:

   ```
   $ raccli update backupconfig -params '{"diskRecoveryWindow" : days}'
   ```

   where ***days*** is the number of days for which you want to retain backups.

Customizing the Recovery Window for Backups to Cloud Storage

> **Note:**
>
> If you have used the `update rdk` subcommand of the `raccli` utility to update the cloud tooling to 16.4.5 or later, you must manually update the opc installer for the Oracle Database Cloud Backup Module before you use the `update backupconfig` subcommand. For instructions, see in Updating the cloud tooling on a deployment hosting Oracle RAC requires manual update of the Oracle Database Cloud Backup Module in *Known Issues for Oracle Database Cloud Service*.

To change the recovery window for backups to the Oracle Storage Cloud container:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Enter this `raccli` command:

   ```
   $ raccli update backupconfig -params '{"ossRecoveryWindow" : days}'
   ```

   where ***days*** is the number of days for which you want to retain backups.

Customizing the Time of Automatic Daily Backups

> **Note:**
>
> If you have used the `update rdk` subcommand of the `raccli` utility to update the cloud tooling to 16.4.5 or later, you must manually update the opc installer for the Oracle Database Cloud Backup Module before you use the `update backupconfig` subcommand. For instructions, see in Updating the cloud tooling on a deployment hosting Oracle RAC requires manual update of the Oracle Database Cloud Backup Module in *Known Issues for Oracle Database Cloud Service*.

To change the time of day when daily automatic backups are performed:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Enter this `raccli` command:

   ```
   $ raccli update backupconfig -params '{"cronDate" : "time"}'
   ```

   where ***time*** is the time (using 24-hour, HH:MM format) when daily backups are to occur. For example, `"02:45"` is 2:45 AM, and `"14:45"` is 2:45 PM.

Enabling and Reconfiguring the Automatic Backups Feature

When you create an Oracle Database Cloud Service database deployment, you specify whether automatic backups are configured by choosing a backup destination. After the deployment is created you can change the configuration by changing the deployment's backup destination.

The instructions in this topic describe how to switch backup destinations for an existing database deployment. Specifically, the following changes are possible using the instructions in this topic:

- From **None** to **Both Cloud Storage and Local Storage**
- From **None** to **Cloud Storage Only** (not currently available for database deployments that use Oracle RAC)
- From **Both Cloud Storage and Local Storage** to **Cloud Storage Only**
- From **Cloud Storage Only** to **Both Cloud Storage and Local Storage**

For background information on the destinations, see About Backing Up Database Deployments on Database Cloud Service.

> **Note:**
>
> The Oracle Database Cloud Service console does not currently recognize changes to the backup destination made by using the `bkup.cfg` file with the backup assistant. Therefore, the console will not reflect the new backup destination and may not display any backups taken, depending on what backup destination change you have made. If the backups are not displayed, you will not be able to use the Oracle Database Cloud Service console to perform recovery.

Prerequisites

- If you are switching to the backup destination Both Cloud Storage and Local Storage, you must increase the size of the local storage used for backups. Use the **Extend Backup Storage Volume** option of the Oracle Database Cloud Service console's scaling feature to add storage such that the backup storage is 1.7 times the size of your database storage. For instructions, see Scaling a Database Deployment.

  > **Note:**
  >
  > Older deployments do not support the **Extend Backup Storage Volume** option, and an error is displayed when you try to use it. In this case, you must manually add backup storage. For instructions, see Increasing Local Storage for Backups on Older Database Deployments.

- If you are switching to the backup destination Both Cloud Storage and Local Storage or Cloud Storage Only, you must have an Oracle Storage Cloud Service container in your account that is reserved for backups. If you don't have one, you must create one. See Creating Containers in *Using Oracle Storage Cloud Service*, or see the tutorial Oracle Storage Cloud Service: Creating Containers Using the REST API.

The commands you use to change the backup destination depend on the type of database running in the database deployment.

Topics

- Changing the Backup Configuration on Database Deployments Hosting Single-Instance Databases
- Changing the Backup Configuration on Database Deployments Hosting Oracle RAC Databases

Changing the Backup Configuration on Database Deployments Hosting Single-Instance Databases

Oracle Database Cloud Service allows you to change the backup destination for your database deployments after creating them.

Before changing the backup destination, make sure you have performed applicable Prerequisites.

1. Connect as the **oracle** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start an RMAN session:

   ```
   $ rman target=/
   ...
   RMAN>
   ```

3. Delete any existing backups.

   ```
   RMAN> delete backup;
   ```

   All backups for this database recorded in the RMAN repository are deleted. (This process may take several minutes.)

4. Exit the RMAN session:

   ```
   RMAN> exit;
   ```

5. If you are switching from the **None** destination, and this is the first time you have done so, you must configure Transparent Data Encryption (TDE). Run the following command:

   ```
   $ /var/opt/oracle/dbaascli/dbaascli tde config --ks_login auto
   ```

   TDE provides encryption of database files at the file level. For information about TDE, including auto login, see *Oracle Database Advanced Security Guide* for Release 18, 12.2, 12.1 or 11.2.

6. Close your connection to the compute node as the **oracle** user.

7. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

8. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

9. Enter this `bkup_api` command to generate a file containing the current backup settings:

   ```
   # /var/opt/oracle/bkup_api/bkup_api get config [--file=filename] --dbname=dbname
   ```

   where ***filename*** is an optional parameter used to specify a name for the file that will be generated and ***dbname*** is the database name for the database that you want to act on.

10. Edit the parameter values in the generated file to change any settings you want to update in the current backup configuration. The following parameters are used to customize the current backup configuration:

Parameter	Description
`bkup_disk`	Enable backup to disk. Valid values are `yes` and `no`.
`bkup_disk_recovery_window`	Recovery window expressed in number of days between 1 and 14.
`bkup_oss`	Enable backup to an Oracle Storage Cloud container. Valid values are `yes` and `no`.
`bkup_oss_passwd`	Oracle Storage Cloud user password. Only applicable when `bkup_oss` is set to `yes`.
`bkup_oss_recovery_window`	Recovery window for backups to an Oracle Storage Cloud container, expressed in number of days between 1 and 30. Only applicable when `bkup_oss` is set to `yes`. Only applicable when `bkup_oss` is set to `yes`.
`bkup_oss_url`	URL for an Oracle Storage Cloud container such as `https://storage.oraclecorp.com/v1/Storage-test/test`.
`bkup_oss_user`	Oracle Storage Cloud username. Only applicable when `bkup_oss` is set to `yes`.

11. Enter this `bkup_api` command to update the backup settings using the file you generated:

```
# /var/opt/oracle/bkup_api/bkup_api set config --file=filename --dbname=dbname
```

where ***filename*** is used to specify a name for the file that will be used to update the backup settings and ***dbname*** is the database name for the database that you want to act on.

12. You can use this `bkup_api` command to check the status of the update:

```
# /var/opt/oracle/bkup_api/bkup_api configure_status
```

13. Exit the root-user command shell:

```
# exit
$
```

Changing the Backup Configuration on Database Deployments Hosting Oracle RAC Databases


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

Oracle Database Cloud Service allows you to change the backup destination for your database deployments after creating them.

Before changing the backup destination, make sure you have performed applicable Prerequisites.

> **Note:**
>
> If you have used the `update rdk` subcommand of the `raccli` utility to update the cloud tooling to 16.4.5 or later, you must manually update the opc installer for the Oracle Database Cloud Backup Module before you use the `update backupconfig` subcommand. For instructions, see in Updating the cloud tooling on a deployment hosting Oracle RAC requires manual update of the Oracle Database Cloud Backup Module in *Known Issues for Oracle Database Cloud Service*.

1. Connect as the **opc** user to compute node 1.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Use the `raccli update backupconfig` command to update the configuration.

   To change the backup destination to **Both Cloud Storage and Local Storage**, enter the following command. Line breaks have been added for clarity; you must enter the command on a single line.

   ```
   $ raccli update backupconfig -params '{"diskEnabled" : true, "ossEnabled" :
   true,
    "cloudStorageUser" : "username", "cloudStoragePwd" : "password",
    "cloudStorageContainerUrl" : "container-URL"}'
   ```

   where:

   - ***username*** is the user name of an Oracle Cloud user who has read/write access to the container.
   - ***password*** is the password of the user specified in `cloudStorageUser`.
   - ***container-URL*** is the URL of the Oracle Storage Cloud container.

Increasing Local Storage for Backups on Older Database Deployments

This topic does not apply to Oracle Cloud Infrastructure.

Older database deployments on Oracle Database Cloud Service do not support the Extend Backup Storage Volume feature, so you must manually add backup storage.

To create a larger storage volume and use it for backups:

1. Create a storage volume for backups. Its size should be 1.7 times the space allocated for data storage.

   Use the **Create New Storage Volume** option of the Oracle Database Cloud Service console's scaling feature to add a storage volume of the appropriate size. For instructions, see Scaling a Database Deployment.

2. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

3. Confirm the addition of the storage volume by listing the devices:

```
$ lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvdb    202:16   0   21G  0 disk
xvdb1 202:17   0  200M  0 part /boot
xvdb2 202:18   0 15.8G  0 part /
xvdb3 202:19   0    4G  0 part [SWAP]
xvdc    202:32   0   10G  0 disk
xvdc1 202:33   0   10G  0 part /u04
xvdd    202:48   0    7G  0 disk
xvdd1 202:49   0    7G  0 part /u03
xvde    202:64   0   30G  0 disk
xvde1 202:65   0   30G  0 part /u01
xvdf    202:80   0   11G  0 disk
xvdf1 202:81   0   11G  0 part /u02
xvdg    202:96   0   17G  0 disk
xvdg1 202:97   0   17G  0 part /u05
```

4. Change from the `opc` user to the `oracle` user:

```
$ sudo su - oracle
```

5. As the `oracle` user, use RMAN to shut down the database instance:

```
$ rman target=/
...
RMAN> shutdown immediate;

using target database control file instead of recovery catalog
database closed
database dismounted
Oracle instance shut down
```

Quit RMAN and exit the `oracle` user session:

```
RMAN> quit;

Recovery Manager complete.
$ exit
```

6. As the `root` user, copy the content from `/u03` (which you have been using for backup storage) to `/u05`:

```
# cp -R /u03/* /u05/
```

7. Unmount the `/u05` and `/u03` mount points:

```
# umount /u05/
# umount /u03/
```

> **Note:**
>
> The Linux *unmount* command is *u*`mount` (with no *n*).

8. Mount the partition of new storage volume you just created to `/u03`, which Oracle Database Cloud Service uses for backups. For example:

```
# mount /dev/xvdg1 /u03
```

9. Look at the results:

```
# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvdb    202:16   0   21G  0 disk
xvdb1 202:17   0  200M  0 part /boot
xvdb2 202:18   0 15.8G  0 part /
xvdb3 202:19   0    4G  0 part [SWAP]
xvdc    202:32   0   10G  0 disk
xvdc1 202:33   0   10G  0 part /u04
xvdd    202:48   0    7G  0 disk
xvdd1 202:49   0    7G  0 part
xvde    202:64   0   30G  0 disk
xvde1 202:65   0   30G  0 part /u01
xvdf    202:80   0   11G  0 disk
xvdf1 202:81   0   11G  0 part /u02
xvdg    202:96   0   17G  0 disk
xvdg1 202:97   0   17G  0 part /u03
```

The old 7 GB partition `xvdd1`, which was previously mounted to `/u03`, is no longer mounted at all. The new 17 GB partition `xvdg1` is now mounted to `/u03` and will therefore be used for backups. Note that the space used for `xvdd1` is now no longer available for any use.

10. Set the ownership and permissions of the mount-point directory `/u03`. For example:

```
# chown -R oracle:oinstall /u03
# chmod 755 /u03
```

11. As the `oracle` user, start SQL*Plus, connect to the database as SYSDBA, and start it. For example:

```
$ sqlplus /nolog

SQL*Plus: Release 12.1.0.2.0 Production on Sat Feb 21 13:19:51 2015

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> connect sys/password as sysdba
Connected to an idle instance.
SQL> startup
ORACLE instance started.

Total System Global Area 3170893824 bytes
Fixed Size                  2929400 bytes
Variable Size            1845497096 bytes
Database Buffers         1308622848 bytes
Redo Buffers               13844480 bytes
Database mounted.
Database opened.
```

12. If you want to see the current setting for the limit on the total space to be used by target database recovery files, show the values of the parameter `DB_RECOVERY_FILE_DEST`, for example:

```
SQL> show parameter
db_recovery_file_dest
NAME                          TYPE        VALUE
----------------------------- ----------- ------------------------------
db_recovery_file_dest         string      /u03/app/oracle/fast_recovery_
                                                 area
db_recovery_file_dest_size    big integer 7G
```

13. Increase the parameter `DB_RECOVERY_FILE_DEST_SIZE` to use the new extra space in `/u03`, for example:

```
SQL> alter system set db_recovery_file_dest_size=17G;

System altered.
```

14. Show the new value:

```
SQL> show parameter db_recovery_file_dest;
NAME                          TYPE        VALUE
----------------------------- ----------- ------------------------------
db_recovery_file_dest         string      /u03/app/oracle/fast_recovery_
                                                 area
db_recovery_file_dest_size    big integer 17G
```

15. Quit SQL*Plus and log out:

```
SQL> quit;
Disconnected from Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 -
64bit Production
$ exit
```

Disabling and Re-enabling Scheduled Backups

You can disable and re-enable regularly scheduled backups of a database deployment by manipulating the scheduling information in the system-wide `/etc/crontab` file.

> **Note:**
>
> Currently, disabling and re-enabling scheduled backups is not supported for Database Cloud Service database deployments that use Oracle Real Application Clusters (RAC).

Disabling Scheduled Backups

To disable scheduled backups:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Enter this `bkup_api` command.

```
# /var/opt/oracle/bkup_api/bkup_api disable backup
```

4. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Re-Enabling Scheduled Backups

To re-enable scheduled backups:

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Enter this `bkup_api` command.

```
# /var/opt/oracle/bkup_api/bkup_api enable backup
```

4. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Recover Backups Using the dbaasapi Utility

> **Note:**
>
> Starting with version 18.4.6, the default behavior of the recover operation changed. The command now recovers only the data and not the configuration files. Specifiy the -cfgfiles option to also recover configuration files.

You can use the `dbaasapi` utility to restore backup files and perform complete recovery on a database deployment hosting a single-instance database:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

```
$ sudo -s
#
```

3. Create a JSON input file with the command parameters. For example:

```
{
                    "action": "start",
                    "object": "db",
                    "operation": "recover",
                    "outputfile": "recovery.out",
                    "params": {
                        "dbname": "<dbname>",
                        "cfgfiles": "yes",
                        "bkup_tag": "TAG20190219T005048"
                    }
                }
```

Where valid values in the params tag are:

- dbname is required for Exadata systems
- cfgfiles optionally specifies whether you want to recover the configuration files. By default, configuration files are not recovered.
- Valid recovery types include:
  - latest specifies the latest available backup
  - bkup_tag specifies the tag of the target backup
  - scn specifies the System Change Number of the target backup
  - timestamp specifies the target timestamp of the target backup. The timestamp must be UTC and of the format dd-MMM-yyyy HH:mm:ss

4. Restore the backup and perform complete recovery using the `orec` subcommand of the `dbaascli` utility:

   ```
   # /var/opt/oracle/dbaasapi/dbaasapi -i <inputFileName>.json
   ```

   The restore and recover process performs these steps:

   - Shuts down the database
   - Extracts and restores configuration files, if "-cfgfiles":"yes" was specified
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database instance after recovery

5. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Restoring from the Most Recent Backup

You can restore the most recent backup and perform complete recovery on an Oracle Database Cloud Service database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Restore from the Most Recent Backup at the end of this topic.

Restoring from the Most Recent Backup by Using the Oracle Database Cloud Service Console

> **Note:**
>
> Currently, using the console to restore from the most recent backup is not supported for database deployments hosting an Oracle Data Guard configuration.

1. Go to the Backup page of the deployment you want to restore and recover:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.
   b. Click the database deployment you want to restore and recover.

      The Oracle Database Cloud Service Overview page is displayed.
   c. Click the Administration tile.

      The Oracle Database Cloud Service Backup page is displayed.
2. Click **Recover**.

   The Database Recovery overlay is displayed.
3. In the list of recovery options, select **Latest**. Then, click **Recover**.

   The restore and recover process performs these steps:
   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database after recovery

Other Ways to Restore from the Most Recent Backup

- You can use the `dbaascli` utility on database deployments hosting a single-instance database. See Restoring from the Most Recent Backup by Using the dbaascli Utility.
- You can use the `raccli` utility on database deployments hosting an Oracle Real Application Clusters (RAC) database. See Restoring from the Most Recent Backup by Using the raccli Utility.

Restoring from the Most Recent Backup by Using the dbaascli Utility

You can use the `dbaascli` utility to restore from the most recent backup and perform complete recovery on a database deployment hosting a single-instance database:

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Restore the most recent backup and perform complete recovery using the `orec` subcommand of the `dbaascli` utility:

   ```
   # dbaascli orec --args -latest
   ```

   The restore and recover process performs these steps:

   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database instance after recovery

4. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Restoring from the Most Recent Backup by Using the raccli Utility


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can use the `raccli` utility to restore from the most recent backup and perform complete recovery on a database deployment hosting an Oracle Real Application Clusters (RAC) database:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Restore the most recent backup and perform complete recovery using the `create recovery` subcommand of the `raccli` utility:

   ```
   $ raccli create recovery -latest
   ```

   The restore and recover process performs these steps:

   - Shuts down the database

- Extracts and restores configuration files
- Prepares for recovery
- Performs the recovery
- Restarts the database instances after recovery

Restoring from a Specific Backup

> **Note:**
>
> Currently, restoring from a specific backup is not supported for database deployments that use Oracle Real Application Clusters (RAC).

You can restore a specific backup and perform recovery to that backup on an Oracle Database Cloud Service database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Restore from a Specific Backup at the end of this topic.

Restoring from a Specific Backup by Using the Oracle Database Cloud Service Console

> **Note:**
>
> Currently, using the console to restore from a specific backup is not supported for database deployments hosting an Oracle Data Guard configuration.

1. Go to the Backup page of the deployment you want to restore and recover:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.
   b. Click the database deployment you want to restore and recover.

      The Oracle Database Cloud Service Overview page is displayed.
   c. Click the Administration tile.

      The Oracle Database Cloud Service Backup page is displayed.
2. In the list of backups, locate the backup you want to restore from.
3. Click the action menu ( ) that is associated with the backup you want to restore from. Choose **Recover** and then confirm the action.

   The restore and recover process performs these steps:
   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery

- Performs the recovery
- Restarts the database after recovery

Other Ways to Restore from a Specific Backup

- For database deployments hosting single-instance databases, you can use the `dbaascli` utility. See Restoring from a Specific Backup by Using the dbaascli Utility.

Restoring from a Specific Backup by Using the dbaascli Utility

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

You can use the `dbaascli` utility to restore from a specific backup and perform recovery to that backup on a database deployment hosting a single-instance database or a Data Guard configuration of single-instance databases.

1. Connect as the **opc** user to the compute node. In a Data Guard configuration, connect to the compute node hosting the primary database.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. List the available backups using the `orec` subcommand of the `dbaascli` utility.

   - For normal backups:

     ```
     # dbaascli orec --args -list
     ```

     A list of available backups displays; for example:

     ```
     Backup Tag          Completion Tag       Config File Location
     ------------------------------------------------------------------
     TAG20140626T191645  06/26/2014 19:16:45

     TAG20140626T195118  06/26/2014 19:51:18  .../ORCL/oscfgfiles/2014_06_26/
     oscfgfiles_20140626_1951.tar.gz
                                              .../ORCL/ohcfgfiles/2014_06_26/
     ohcfgfiles_20140626_1951.tar.gz

     Config files relative path to: /u03/app/oracle/fast_recovery_area
     ```

   - For long-term backups:

     ```
     # dbaascli orec --args -keep -list
     ```

     A list of available long-term backups displays; for example:

```
TAG20120117T065489
TAG20110117T077324
TAG20100117T023955
```

4. Restore the specific backup you want using the `orec` subcommand:

   - For normal backups:

     ```
     # dbaascli orec --args -pitr backup-tag
     ```

   - For long-term backups:

     ```
     # dbaascli orec --args -keep -tag backup-tag
     ```

   where ***backup-tag*** is the tag of the backup you want to restore.

   The restore and recover process performs these steps:

   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database instance after recovery

   In a Data Guard configuration, after the restore and recovery operation is complete, a message indicating that you need to run the `duplicate` command on the standby instance is displayed. Perform all of the following steps to complete the operation on the standby database. In a single-instance database you only need to perform the next step.

5. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

6. In the Data Guard configuration, connect to the standby instance's compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

7. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

8. Run the `duplicate` option of `orec`.

   ```
   # dbaascli orec --args -duplicate
   ```

9. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

Restoring to a Specific Point in Time

You can restore from a backup and perform recovery to a specific point in time on an Oracle Database Cloud Service database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Restore to a Specific Point in Time at the end of this topic.

Restoring to a Specific Point in Time by Using the Oracle Database Cloud Service Console

> **Note:**
>
> Currently, using the console to restore to a specific point in time is not supported for database deployments hosting an Oracle Data Guard configuration.

1. Go to the Backup page of the deployment you want to restore and recover:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.
   b. Click the database deployment you want to restore and recover.

      The Oracle Database Cloud Service Overview page is displayed.
   c. Click the Administration tile.

      The Oracle Database Cloud Service Backup page is displayed.
2. Click **Recover**.

   The Database Recovery overlay is displayed.
3. In the list of recovery options, select **Date and Time** or **System Change Number** (SCN) to indicate how you want to specify the end point of the recovery operation. Then, enter the appropriate value.

   > **Note:**
   >
   > If specified, the recovery date and time values are subject to the UTC time zone.
4. Click **Recover**.

   The restore and recover process performs these steps:
   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database after recovery

Other Ways to Restore to a Specific Point in Time

- For database deployments hosting single-instance databases, you can use the `dbaascli` utility. See Restoring to a Specific Point in Time by Using the dbaascli Utility.

- For database deployments hosting Oracle Real Application Clusters (RAC) databases, you can use the `raccli` utility. See Restoring to a Specific Point in Time by Using the raccli Utility.

Restoring to a Specific Point in Time by Using the dbaascli Utility

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

You can use the `dbaascli` utility to restore from a backup and perform recovery to a specific point in time on a database deployment hosting a single-instance database or a Data Guard configuration.

1. Connect as the **opc** user to the compute node. In a Data Guard configuration, connect to the compute node hosting the primary database.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Restore the most recent backup and perform complete recovery using the `orec` subcommand of the `dbaascli` utility:

   ```
   # dbaascli orec --args -scn SCN
   ```

   where ***SCN*** is the system change number (SCN) for the end point of the recovery.

   The restore and recover process performs these steps:

   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database instance after recovery

   In a Data Guard configuration, after the restore and recovery operation is complete, a message indicating that you need to run the `duplicate` command on the standby instance is displayed. Perform all of the following steps to complete the operation on the standby database. In a single-instance database you only need to perform the next step.

4. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

5. In the Data Guard configuration, connect to the standby instance's compute node as the **opc** user.

For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

6. Start a root-user command shell:

```
$ sudo -s
#
```

7. Run the `duplicate` option of `orec`.

```
# dbaascli orec --args -duplicate
```

8. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Restoring to a Specific Point in Time by Using the raccli Utility


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can use the `raccli` utility to restore from a backup and perform recovery to a specific point in time on a database deployment hosting an Oracle Real Application Clusters (RAC) database:

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Perform recovery to a specific point in time using the `create recovery` subcommand of the `raccli` utility:

   ```
   $ raccli create recovery -pitr -scn SCN
   ```

   or

   ```
   $ raccli create recovery -pitr -timestamp time
   ```

   where ***SCN*** is the system change number (SCN) for the end point of the recovery and ***time*** is time (in the format `MM/DD/YYYY HH24:MI:SS`) for the end point of the recovery.

   The restore and recover process performs these steps:

   - Shuts down the database
   - Extracts and restores configuration files
   - Prepares for recovery
   - Performs the recovery
   - Restarts the database instances after recovery

Retrieve the History of Scheduled Backup Results with the bkup_api Utility

Retrieve details for each backup instance in the history of a scheduled backup.

You can use the `bkup_api` utility to list each backup job for a scheduled backup. The command can also display CDB jobs and `bkup_archlogs` as well as details of a specific job.

1. Connect as the **opc** user to the compute node. In a Data Guard configuration, connect to the compute node hosting the primary database. For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH)
2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Enter the following `bkup_api` command:

   ```
   # /var/opt/oracle/bkup_api/bkup_api list jobs --all
   ```

   where:

   - `--all` specifies that the job list contains CDB jobs and `bkup_archlogs`.
   - `--dbname=`*`dbname`* is required for ExaData systems and is the database name for the database that you want to recover.
   - `--pdb` specifies that the list only contains PDB jobs from the specified database.
   - `--uuid=`*`uuid`* specifies that the command returns detailed information about the job with the specified UUID.

4. Choose a job UUID from the list displayed and enter the following command:

   ```
   # /var/opt/oracle/bkup_api/bkup_api list jobs  --uuid=uuid
   ```

5. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

```
[opc]# /var/opt/oracle/bkup_api/bkup_api list jobs --dbname=sample --all
DBaaS Backup API V1.5 @2016 Multi-Oracle home
-> This node is not master. Running on Elastic configuration.
-> Action : list_jobs
-> logfile: /var/opt/oracle/bkup_api/log/bkup_api.log
UUID                                | DATE                | STATUS  |
TAG                 | TYPE
c3d11e6c250211e9af3100163e6d8d75 | 2019-01-31 02:49:11 | success |
None                | archivelog
25c15436250b11e98a2000163e6d8d75 | 2019-01-31 03:49:12 | success |
None                | archivelog
87775fa6251311e9aca300163e6d8d75 | 2019-01-31 04:49:11 | success |
None                | archivelog
e9ce9da6251b11e985b400163e6d8d75 | 2019-01-31 05:49:12 | success |
None                | archivelog
4b375fbc252411e9be4e00163e6d8d75 | 2019-01-31 06:49:11 | success |
None                | archivelog
```

```
acb0ebe8252c11e9a12a00163e6d8d75 | 2019-01-31 07:49:11 | success |
None               | archivelog
0eaf0fd4253511e9a4bd00163e6d8d75 | 2019-01-31 08:49:12 | success |
None               | archivelog
7056d17e253d11e99e9b00163e6d8d75 | 2019-01-31 09:49:12 | success |
None               | archivelog
d2897916254511e9bfd200163e6d8d75 | 2019-01-31 10:49:12 | success |
None               | archivelog
34328510254e11e988cc00163e6d8d75 | 2019-01-31 11:49:12 | success |
None               | archivelog
95d6b522255611e9a59300163e6d8d75 | 2019-01-31 12:49:11 | success |
None               | archivelog
f79613a4255e11e9b1ea00163e6d8d75 | 2019-01-31 13:49:12 | success |
None               | archivelog
593d0844256711e9b75f00163e6d8d75 | 2019-01-31 14:49:11 | success |
None               | archivelog
bb4339de256f11e9a72c00163e6d8d75 | 2019-01-31 15:49:13 | success |
None               | archivelog
1d7dcf94257811e99c2100163e6d8d75 | 2019-01-31 16:49:13 | success |
None               | archivelog
80213598258011e9978700163e6d8d75 | 2019-01-31 17:49:30 | success |
None               | archivelog
e04ec392258811e9a96400163e6d8d75 | 2019-01-31 18:49:13 | success |
None               | archivelog
4264c9ca259111e990a900163e6d8d75 | 2019-01-31 19:49:13 | success |
None               | archivelog
a3e8f344259911e9bf4500163e6d8d75 | 2019-01-31 20:49:11 | success |
None               | archivelog
05cec57225a211e9aff500163e6d8d75 | 2019-01-31 21:49:12 | success |
None               | archivelog
678ba60625aa11e9993000163e6d8d75 | 2019-01-31 22:49:12 | success |
None               | archivelog
...

[opc]# /var/opt/oracle/bkup_api/bkup_api --
uuid=sampled829e711e993d700163e6d8d75
DBaaS Backup API V1.5 @2016 Multi-Oracle home
DBaaS Backup API V1.5 @2015 Multi-Oracle home
-> This node is not master. Running on Elastic configuration.
@ STARTING CHECK STATUS sampled829e711e993d700163e6d8d75
[ REQUEST TICKET ]
[UUID    ->  sampled829e711e993d700163e6d8d75
[DBNAME  ->  FSPOD1
[STATE   ->  failed
[ACTION  ->  create-backup-incremental
[STARTED ->  2019-02-06 08:19:34
[ENDED   ->  2019-02-06 00:46:51.469039
[PID     ->  258279
[TAG     ->  None
[ERROR   ->  API::ERROR Cannot complete the Incremental backup to cloud storage.
[ERROR   ->  API::ERROR Please check the obkup /var/opt/oracle/log/FSPOD1/obkup/
obkup_2019-02-06_00:19:52.log for more information.
[ERROR   ->  API::ERROR RMAN errors found during backup
[ERROR   ->  API::ERROR-Detail
[ERROR   ->  OBKUP:: ..... FAIL
[ERROR   ->  RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============
[ERROR   ->  RMAN-03002: failure of backup command at 02/06/2019 00:46:05
[ERROR   ->  RMAN-03002: failure of backup command at 02/06/2019 00:46:22
[LOG     ->  API:: Oracle database state is up and running
[LOG     ->  API:: DB instance: FSPOD1
[LOG     ->  API:: Validating the backup repository ......
```

```
[LOG     ->  API::      All backup pieces are ok
[LOG     ->  API:: Performing Incremental Backup to Cloud Storage
[LOG     ->  API:: Executing rman instructions
[LOG     ->  API:: ..... FAIL
[LOG     ->  API::ERROR Cannot complete the Incremental backup to cloud storage.
[LOG     ->  API::ERROR-Detail
[LOG     ->  API::ERROR RMAN errors found during backup
[LOG     ->  API::ERROR Please check the obkup /var/opt/oracle/log/FSPOD1/obkup/
obkup_2019-02-06_00:19:52.log for more information.
[LOG     ->  API:: Message sent to DB alertlog.
[LOG     ->  API:: Message sent to System log.
[LOG     ->  API:: Clean MOTD.
[LOG     ->  API:: Message sent to MOTD.
[ END TICKET ]
[opc]#
```

Recreating an Unrecoverable Database Deployment From a Backup to Cloud Storage

For a database deployment on Oracle Database Cloud Service that hosts a single-instance database or a Data Guard configuration of single-instance databases, you can use the `mrec` media recovery utility to recreate the deployment if it cannot be restored and recovered using the `orec` command. This situation could occur if you've deleted critical configuration files or data files (such as the database redo log files, for example), or if something else has happened that caused the deployment to be lost.

The `mrec` utility restores the database deployment to the point of the last backup. This is equivalent to restoring the database and configuration from an external tape device. The `mrec` utility should be considered a last resource and used only when a database deployment cannot be restored in any other way.

Before You Begin

To recreate a deployment using the `mrec` utility, the following condition must be met:

- A backup of the original deployment, including configuration files, must exist in cloud storage. The recreated deployment will be restored up to the latest backup available in cloud storage (data files and configuration files). If such a backup doesn't exist, you can't use `mrec` to restore the deployment.

You must also know the following:

- The settings used to create the original deployment.
- The credentials and URL of the Oracle Storage Cloud Service instance to which the original deployment is backed up.
- The database system identifier (SID) associated with the original deployment.

Procedure

Perform the following steps to recreate a database deployment using `mrec`. If you are recreating a deployment hosting a Data Guard configuration, perform steps on the compute node hosting the primary database unless you are directed otherwise.

1. Use the Create Instance wizard to create a new database deployment of the same type as the original deployment. See Creating a Customized Database

Deployment. The backup of the original deployment will be restored to this new deployment.

Make sure you specify the following:

- The same Oracle Database version and software edition as the original deployment.
- On the Instance Details page, the same settings as the original deployment. At a minimum, the compute shape, storage capacity, database SID, Data Guard configuration, and backup and recovery configuration settings must be identical to those of the original deployment. If the compute node associated with the original deployment still exists, a different deployment name can be used.

2. Reduce the newly created database deployment to a minimal configuration:
   a. Connect as the **opc** user to the compute node associated with the newly created database deployment.

      For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

   b. Start a root-user command shell:

      ```
      $ sudo -s
      #
      ```

   c. Run the `deinstall` script:

      ```
      # /var/opt/oracle/misc/deinstall.pl -dbname=dbname
      ...
      INFO : PLEASE REBOOT the VM before re-running ocde/dbsetup.sh
      ```

   d. Reboot the compute node:

      ```
      # reboot
      ...
      The system is going down for reboot NOW!
      ```

   e. Close your connection to compute node if it is not automatically closed due to the reboot.
3. If you are recreating a Data Guard configuration, repeat the preceding step on the compute node hosting the standby database.
4. Prepare the newly created deployment so `mrec` can be used:
   a. After giving the compute node associated with the newly created deployment time to reboot, connect to it as the **opc** user.
   b. Start a root-user command shell:

      ```
      $ sudo -s
      #
      ```

   c. Edit (or create, if necessary) the file `/var/opt/oracle/ocde/sm_params.cfg`, giving it the following content:

      ```
      adminPassword=password
      cloudStorageContainer=backup-container
      cloudStorageUser=storage-username
      cloudStoragePwd=storage-password
      ```

      Where:

- *password* is the administrator password specified when the new database deployment was created.
- *backup-container* is the fully qualified name of the storage container where backups are stored. For example, `https://exampledomain.storage.oraclecloud.com/v1/Storage-exampledomain/dbcsbackups`. Do not enclose this value in quotation marks.
- *storage-username* is the user name of the Oracle Cloud user to use when accessing the storage container.
- *storage-password* is the password of the user specified in `cloudStorageUser`.

> **Note:**
>
> As a security measure, this file will be deleted when you run the `ocde` script later.

d. Update the timestamp on the `sm_params.lk` file:

```
# touch /var/opt/oracle/ocde/sm_params.lk
```

e. Run the `ocde` script to execute the `prep` and `sda` assistants:

```
# /var/opt/oracle/ocde/ocde -alist='prep sda' -firstrun
Starting OCDE
...
#### Completed OCDE Successfully ####
```

This configuration will take some time to complete.

f. Keep this connection open for later use.

5. If you are recreating a Data Guard configuration, repeat the preceding step on the compute node hosting the standby database. Close the connection to the compute node that hosts the standby database after the last step completes.
6. Create a new `oss.cfg` configuration file with the exact same parameters and values as those in the `oss.cfg` file of the original deployment:

a. Connect to the original deployment as `opc` user and start a root-user command shell. If the original deployment is a Data Guard configuration, connect to the compute node hosting the primary database.

b. View the contents of the original deployment's `oss.cfg` file:

```
# cat /home/oracle/bkup/SID/oss.cfg
```

(On older database deployments the file is located at `/home/oracle/bkup/oss.cfg`.)

Copy the parameters in the file. The same parameters and values must be used in the `oss.cfg` file for the new deployment.

If you can't access the `oss.cfg` file for the original deployment, these are the parameters you'll need to specify:

```
oss_tid=storage-service-identity-domain
oss_sname=storage-service-name
oss_user=storage-service-admin-user-name
```

```
oss_passwd=storage-service-admin-user-password
oss_url=storage-service-container-URL
oss_auth_url=storage-service-authentication-URL
```

For example:

```
oss_tid=usoracle04791
oss_sname=dbaasoss
oss_user=admin-user-name
oss_passwd=admin-user-password
oss_url="https://storage.us2.oraclecloud.com/v1/dbaasoss-usoracle04791/
mycontainer"
oss_auth_url="https://storage.us2.oraclecloud.com/auth/v1.0"
```

c. Switch back to your connection to the new deployment. If the deployment hosts a Data Guard configuration, switch back to the connection to the compute node hosting the primary database.

d. Create an `oss.cfg` file that contains the parameters used in the `oss.cfg` file of the original deployment:

```
# cd /var/opt/oracle/mrec
# vim oss.cfg
```

e. Change ownership of the new `oss.cfg` file from the `root` user to the `oracle` user. Also change permissions:

```
# chown oracle:oinstall oss.cfg
# chmod 0600 oss.cfg
```

7. While still in the root-user command shell connection to the new deployment, run the `mrec` utility:

```
# cd /var/opt/oracle/mrec
# ./mrec -oss_cfgfile ./oss.cfg
-old_hostname hostname-of-node-to-restore -sid SID-of-instance-to-restore
```

where:

- ***hostname-of-node-to-restore*** is the simple name of the compute node to restore (doesn't need to be fully qualified). For example, `prod01`.
- ***SID-of-instance-to-restore*** is the database system identifier (SID) of the database instance to restore. For example, `orcl`.

The `mrec` utility pulls files over from the original deployment, installs the module used for cloud backups, and attempts recovery. Information about progress and status is displayed in the terminal window as the utility runs. If recovery is successful, you'll see a message indicating that the deployment has been recovered and is in an open state. The amount of time this takes depends on the size of the deployment that you're recovering.

8. If you are recreating a database deployment hosting a Data Guard configuration, you need to close and open each PDB.

a. Switch to the oracle user:

```
# su - oracle
$
```

b. Invoke SQL*Plus.

```
$ sqlplus '/ as sysdba'
```

c. Alter your session to connect to each PDB, and then close and reopen each PDB.

```
SQL> ALTER SESSION SET CONTAINER = pdb-name;
SQL> ALTER PLUGGABLE DATABASE CLOSE IMMEDIATE;
SQL> ALTER PLUGGABLE DATABASE OPEN;
```

d. Log out of SQL*Plus.

```
SQL> exit;
```

e. Return to being the root user.

```
$ exit
#
```

9. If you are recreating a database deployment hosting a Data Guard configuration, perform this step to recreate the standby database.

```
$ export HOSTNAME=`hostname -s`
$ /var/opt/oracle/ocde/assistants/dg/dgcc -out /var/opt/oracle/ocde/res/
dg_mrec.out
```

10. Close your connection to the compute node.

7

Patching Database Cloud Service

This section explains how to apply a patch to Oracle Database Cloud Service, and roll back the patch as necessary.

Topics

For general information about patching Oracle Database, see "Patch Set Updates and Requirements for Upgrading Oracle Database" in the *Oracle Database Upgrade Guide* for Release 18, 12.2, 12.1 or 11.2.

Viewing Available Patches

You can view a list of patches that are associated with a Database Cloud Service database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to View Available Patches at the end of this topic.

Viewing Available Patches by Using the Oracle Database Cloud Service Console

1. Go to the Patching page for the database deployment on which you want to check patching:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the database deployment on which you want to check patching.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Patching tab.

      The Oracle Database Cloud Service Patching page is displayed.
2. A list of patches you can apply appears in the Available Patches section.

Other Ways to View Available Patches

- For database deployments hosting single-instance databases, you can use the `patch db list` subcommand of the `dbaascli` utility. See Viewing Available Patches by Using the dbaascli Utility.

Viewing Available Patches by Using the dbaascli Utility

You can use the `patch db list` subcommand of the `dbaascli` utility to check whether any patches are available for a database deployment hosting a single-instance database or an Oracle Data Guard configuration of single-instance databases.

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. View available patches:

   ```
   # dbaascli patch db list
   ```

   A patch update is available if the command response includes the "INFO: images available for patching" message. The patch ID will be displayed as part of the command response. This patch ID can be used to download and apply the patch.

   
   The `patch db list` subcommand is not yet available on Oracle Cloud at Customer. Instead, for a database deployment hosting a single-instance database you must use the dbaascli dbpatchm list_patches command, and for a database deployment hosting an Oracle Data Guard configuration of single-instance databases you must use the `dbpatchmdg -list_patches` command.

4. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Checking Prerequisites Before Applying a Patch

Before you apply a patch, you can check its prerequisites to make sure that it can be successfully applied by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Check Prerequisites Before Applying a Patch at the end of this topic.

The prerequisites-checking operation:

- Confirms that the patch is available for download.
- Verifies that there is enough space in the `/u01` directory to apply the patch.
- Compares the patch's prerequisites to the database deployment by running `opatch prereq` commands.

Checking Prerequisites Before Applying a Patch by Using the Oracle Database Cloud Service Console

Before You Begin

Before checking patch prerequisites, make sure the database deployment has the latest cloud tooling because some patches require a certain minimum level of cloud

tooling. For more information, see Updating the Cloud Tooling on Database Cloud Service.

Procedure

1. Go to the Patching page for the database deployment on which you want to check patching:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the database deployment on which you want to check patching.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Patching tab.

      The Oracle Database Cloud Service Patching page is displayed. A list of patches you can apply appears in the Available Patches section.

2. Click the action menu () that is associated with the patch whose prerequisites you want to check, and then select **Precheck**.

   If further input is required, specify the required details in the Patch Precheck Service window and click **Precheck** to continue. The Patch Precheck Service window displays in the following circumstances:

   - If you have previously checked prerequisites for the selected patch, the Patch Precheck Service window shows the results of the previous check and asks if you want to perform another set of prerequisite checks.

   The Patching page redisplays, showing a status message indicating that prerequisite checks are in progress.

3. Refresh the Patching page occasionally to update the status message.

   Note that prerequisite checking can take several minutes to complete.

4. When the prerequisite checks are completed, the Precheck results link is displayed.

   Click Precheck results to display the results of the prerequisite checks.

Other Ways to Check Prerequisites Before Applying a Patch

- For database deployments hosting single-instance databases, you can use the `patch db prereq` subcommand of the `dbaascli` utility. See Checking Patch Prerequisites by Using the dbaascli Utility.
- For database deployments hosting Oracle Real Application Clusters (RAC) databases, you can use the `prechecks` option of the `raccli apply patch` command. See Checking Patch Prerequisites by Using the raccli Utility.

Checking Patch Prerequisites by Using the dbaascli Utility

You can use the `patch db prereq` subcommand of the `dbaascli` utility to the check the prerequisites of a patch before you apply it to a database deployment hosting a

single-instance database or an Oracle Data Guard configuration of single-instance databases.

Before You Begin

Before checking patch prerequisites, make sure the database deployment has the latest cloud tooling because some patches require a certain minimum level of cloud tooling. For more information, see Updating the Cloud Tooling on Database Cloud Service.

Procedure

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Check the the prerequisites of a patch:

   ```
   # dbaascli patch db prereq
   ```

   The prerequisites report displays.

   
   The `patch db prereq` subcommand is not yet available on Oracle Cloud at Customer. Instead, for a database deployment hosting a single-instance database you must use the dbaascli dbpatchm prereq command, and for a database deployment hosting an Oracle Data Guard configuration of single-instance databases you must use the dbpatchmdg precheck_async command.

4. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Checking Patch Prerequisites by Using the raccli Utility


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can use the `prechecks` option of the `raccli apply patch` subcommand to check the prerequisites of a patch before you apply it to a database deployment hosting an Oracle Real Application Clusters (RAC) database.

Before you begin

Before you precheck a patch, make sure the database deployment has the latest cloud tooling. For more information, see Updating the Cloud Tooling by Using the raccli Utility.

Procedure

1. Connect to the compute node as the **opc** user.

For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Precheck the patch using the `apply patch -prechecks` subcommand for the database type you are checking:
   - **Database Clustering with RAC**:

     ```
     $ raccli apply patch -db -tag tag-name -prechecks
     ```

   - **Database Clustering with RAC and Data Guard Standby**:

     ```
     $ raccli apply patch -db -tag tag-name -prechecks -dg
     ```

   Where *tag-name* is the name of the patch. To find out the tag name for the latest available patch, see *What's New for Oracle Database Cloud Service*.
3. Track the progress of the precheck job to its completion by using the raccli describe job command.

Applying a Patch

You can apply a patch that is associated with a database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Apply a Patch at the end of this topic.

> **Note:**
>
> The Oracle Database Cloud Service console does not currently reflect patching information if you apply a patch by using a command-line utility on a database deployment's compute nodes.

Applying a Patch by Using the Oracle Database Cloud Service Console

Before You Begin

- Before you apply a patch, make sure the database deployment has the latest cloud tooling because some patches require a certain minimum level of cloud tooling. For more information, see Updating the Cloud Tooling on Database Cloud Service.
- Before you apply a patch, you should back up the deployment. For instructions, see Creating an On-Demand Backup.

Procedure

1. Go to the Patching page of the database deployment to which you want to apply a patch:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the database deployment to which you want to apply a patch.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Patching tab.

The Oracle Database Cloud Service Patching page is displayed. A list of patches you can apply appears in the Available Patches section.

2. Click the action menu ( ) that is associated with the patch you want to apply, and then select **Patch**.

   The Patch Service window displays.

3. If you want errors to be ignored during the patching operation, select the **Force apply patch** option. Then, click **Patch**.

   If the **Force apply patch** option is selected, patch conflicts or errors discovered during the precheck stage of the patching operation are ignored and the patch will be applied (space permitting). If the option is not selected and conflicts or errors are discovered, the patch will not be applied.

   The Patch Service window closes and the patching operation begins.

   The Administration tile shows the starting time of the patching operation and a **Patching...** message replaces the **Patch** button.

   When the patching operation completes, the Patching page shows the completion time of the patching operation, and a log of the operation's activities appears in the Details of Last Patching Activity section. If the operation was successful, the patch is removed from the list of patches in the Available Patches. If the operation failed, the patch remains in the list. In this case, check the Details of Last Patching Activity section for information about the failure.

> **Note:**
>
> Patching operations are performed with a minimum of impact on the functioning of the database. For database deployments that include multiple compute nodes, patching operations are performed in a rolling manner, one compute node at a time, in order to minimize impact on the database.

Other Ways to Apply a Patch

- For database deployments hosting single-instance databases, you can use the `patch db apply` subcommand of the `dbaascli` utility. See Applying a Patch by Using the dbaascli Utility.
- For database deployments hosting an Oracle Real Application Clusters (RAC) database, you can use the `raccli` utility. See Applying a Patch by Using the raccli Utility.

Applying a Patch by Using the dbaascli Utility

You can use the `patch db apply` subcommand of the `dbaascli` utility to apply a patch to a database deployment hosting a single-instance database or an Oracle Data Guard configuration of single-instance databases.

Before you begin

- Before you apply a patch, make sure the database deployment has the latest cloud tooling because some patches require a certain minimum level of cloud

tooling. For more information, see Updating the Cloud Tooling by Using the dbaascli Utility.

- Before you apply a patch, you should back up the deployment. For instructions, see Creating an On-Demand Backup.

Procedure

To apply a patch to a database deployment by using the `patch db apply` subcommand:

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Apply the patch to the deployment using the `patch db apply` subcommand of `dbaascli`:

   ```
   # dbaascli patch db apply
   ```

   The `patch db apply` subcommand displays progress as the patch is applied.

   
   The `patch db apply` subcommand is not yet available on Oracle Cloud at Customer. Instead, for a database deployment hosting a single-instance database you must use the dbaascli dbpatchm apply command, and for a database deployment hosting an Oracle Data Guard configuration of single-instance databases you must use the dbpatchmdg apply_async command.

   > **Note:**
   >
   > Patching operations are performed with a minimum of impact on the functioning of the database. However, during part of the operation the database is shut down for a period of time, thus making it inaccessible.

4. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Applying a Patch by Using the raccli Utility


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can use the `apply patch` subcommand of the `raccli` utility to apply a patch to a database deployment hosting an Oracle Real Application Clusters (RAC) database.

Before you begin

- Before you apply a patch, make sure the database deployment has the latest cloud tooling because some patches require a certain minimum level of cloud tooling. For more information, see Updating the Cloud Tooling by Using the raccli Utility.
- Before you apply a patch, you should back up the deployment. For instructions, see Creating an On-Demand Backup.

Procedure

To apply a patch to a database deployment by using the `apply patch` subcommand:

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Apply a patch using the `apply patch` subcommand for the database type you are patching:

   - **Database Clustering with RAC**:

     ```
     $ raccli apply patch -db -tag tag-name
     ```

   - **Database Clustering with RAC and Data Guard Standby**:

     ```
     $ raccli apply patch -db -tag tag-name -dg
     ```

   Where *tag-name* is the name of the patch. To find out the tag name for the latest available patch, see *What's New for Oracle Database Cloud Service*.

   The Oracle Database home and Grid Infrastructure home are updated on both compute nodes of the deployment. The node from which you run the command is taken offline, patched, and then brought back online. Then the second node is taken offline, patched, and brought back online. For Data Guard configurations, the standby database nodes are patched first, followed by the primary database nodes.

3. Track the progress of the patching job to its completion by using the raccli describe job command.

Rolling Back a Patch or Failed Patch

> **Note:**
>
> Currently, rolling back a patch is not supported for database deployments hosting an Oracle Real Application Clusters (RAC) database.

You can roll back a patch or failed patch attempt on a database deployment by using the Oracle Database Cloud Service console or, if desired, by using one of the ways listed in Other Ways to Roll Back a Patch or Failed Patch at the end of this topic.

> **Note:**
>
> Oracle strongly recommends against performing the rollback operation on a database deployment that has never had patches applied to it. You should only use the rollback operation on patches that you have applied to a database deployment.

> **Note:**
>
> Beginning with the April 2015 Patch Set Update (Apr 2015 PSU), Oracle adopted a “composite” approach to patch set updates. With this composite approach, a rollback operation restores the software release level to the previous patch set level instead of to the base software release level. For example, if you roll back the April 2015 Patch Set Update, the software is restored to the January 2015 Patch Set Update release level, not the base release level.

Rolling Back a Patch or Failed Patch by Using the Oracle Database Cloud Service Console

To roll back the last patch or failed patch attempt by using the Oracle Database Cloud Service console:

1. Go to the Patching page of the database deployment on which you want to roll back a patch:
   a. Open the Oracle Database Cloud Service console.

      For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

   b. Click the database deployment on which you want to roll back a patch.

      The Oracle Database Cloud Service Overview page is displayed.

   c. Click the Administration tile and then click the Patching tab.

      The Oracle Database Cloud Service Patching page is displayed.
2. Click **Rollback**.

   The Patching page redisplays, showing a status message that your request has been submitted, the Administration tile shows the starting time of the rollback operation, and a **Rolling back...** message replaces the **Rollback** button.

   > **Note:**
   >
   > Rollback operations are performed with a minimum of impact on the functioning of the database. However, during a patch rollback operation the database may be shut down for a short period of time, thus making it inaccessible.

Other Ways to Roll Back a Patch or Failed Patch

- For database deployments hosting single-instance databases, you can use the `patch db switchback` subcommand of the `dbaascli` utility. See Rolling Back a Patch or Failed Patch by Using the dbaascli Utility.

Rolling Back a Patch or Failed Patch by Using the dbaascli Utility

You can use the `patch db switchback` subcommand of the `dbaascli` utility to roll back the last patch or failed patch attempt to a database deployment hosting a single-instance database or an Oracle Data Guard configuration of single-instance databases.

1. Connect as the **opc** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Roll back the last patch in the deployment using the `patch db switchback` subcommand of `dbaascli`:

   ```
   # dbaascli patch db switchback
   ```

   The patch is removed and the database instance reverts to its previous state.

   The `patch db switchback` subcommand is not yet available on Oracle Cloud at Customer. Instead, for a database deployment hosting a single-instance database you must use the dbaascli dbpatchm rollback command, and for a database deployment hosting an Oracle Data Guard configuration of single-instance databases you must use the dbpatchmdg rollback_async command.

   > **Note:**
   >
   > Rollback operations are performed with a minimum of impact on the functioning of the database. However, during part of the operation the database instance is shut down, thus making it inaccessible.

4. Exit the root-user command shell:

   ```
   # exit
   $
   ```

Applying a Patch to a Test Deployment

You can use the `clonedb` option of the `dbpatchm` subcommand to apply a patch to a test deployment of Oracle Database Cloud Service before you apply it to a live, production database deployment.

> **Note:**
>
> Currently, applying a patch to a test deployment is not supported for database deployments that use Oracle Real Application Clusters (RAC).

To apply a patch to a test deployment:

1. Using the Create Instance wizard, create a test deployment on Database Cloud Service, providing the same information as you did when creating the live deployment.
2. Connect as the `opc` user to the test deployment and then perform these steps:
   a. Start a root-user command shell:

      ```
      $ sudo -s
      #
      ```

   b. Run the following commands:

      ```
      # cd /home/oracle
      # ./deinstall.pl -dbname=dbname
      ...
      # /var/opt/oracle/ocde/ocde -dump -alist=prep sda
      ...
      #
      ```

   c. Exit the root-user command shell and disconnect from the test deployment.
3. Copy the SSH private key file for the SSH key used when creating the test deployment to the live deployment.
4. Connect as the `opc` user to the live deployment and then start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

5. Edit the `/var/opt/oracle/patch/dbpatchm.cfg` patching configuration file, setting the normal keys to perform a patching operation. In addition, set the following keys:
   - `cloning`: set this key to **`yes`**
   - `remotenode`: set this key to the IP address of the test deployment
   - `sshkey_fl`: set this key to value of the SSH private key file you copied to the live deployment
   - `upg`: set this key to **`upg`**
6. Apply the patch to the test deployment using the `dbpatchm` subcommand of `dbaascli`:

   ```
   # dbaascli dbpatchm --run -clonedb
   ```

   The `dbpatchm` subcommand displays progress as it copies information from the live deployment to the test deployment and then applies the patch to the test deployment.
7. Exit the root-user command shell and disconnect from the test deployment.

After applying the patch to the test deployment, confirm appropriate application of the patch on the test deployment. When satisfied, you can delete the test deployment and apply the patch to the live deployment.

Patching a Hybrid DR Deployment


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You cannot use the Oracle Database Cloud Service console or the `patch` subcommand of the `dbaascli` utility to patch the on-premises database in your Hybrid DR deployment. However, you can use the tools to patch the database on the cloud compute node in your Hybrid DR deployment.

To apply quarterly patches to the databases in your Hybrid DR deployment:

1. Ensure that the on-premises database is up-to-date with patches:
   a. For the April PSU, download one of the following files:
      - From Chicago: https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/dbaas_patch/dbcs_hdg_onprem/apr2017/hdg_patches.zip
      - From Slough: https://a88717.storage.oraclecloud.com/v1/Storage-a88717/dbaas_patch/dbcs_hdg_onprem/apr2017/hdg_patches.zip
   b. For the July PSU, download one of the following files:
      - From Chicago: https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/dbaas_patch/dbcs_hdg_onprem/jul2017/hdg_patches.zip
      - From Slough: https://a88717.storage.oraclecloud.com/v1/Storage-a88717/dbaas_patch/dbcs_hdg_onprem/jul2017/hdg_patches.zip
   c. Unzip the downloaded PSU file and follow the instructions in the README file to install the patch and execute any required scripts.
2. Patch the database on the cloud compute node and then run the post-install script:
   a. Use the Oracle Database Cloud Service console or the `patch` subcommand of the `dbaascli` utility. See Applying a Patch for details.
   b. Connect as the **`oracle`** user to the compute node.
   c. Run the post-install script:
      - For Oracle Database 12*c*:

        ```
        $ datapatch -verbose
        ```

      - For Oracle Database 11*g*:

        ```
        $ cd $OH/rdbms/admin
        $ sqlplus / as sysdba
        SQL> @@catbundle.sql psu apply
        SQL> exit;
        ```

   d. Disconnect from the compute node.

After applying the patch to the databases in the Hybrid DR deployment, confirm appropriate application of the patch .

The dbpatchm.cfg Configuration File

To perform a patching operation, the `dbpatchm` subcommand reads and acts on the content of the `/var/opt/oracle/patch/dbpatchm.cfg` patching configuration file. This file, which is created when the database deployment is created, provides information about the locations of various files that may be used in patching operations.

Before using `dbpatchm` to perform a patching operation, set the value of the `psunum` key in the `dbpatchm.cfg` file to the patch ID of the patch to apply. To find out what patches are available, see Viewing Available Patches.

The `dbpatchm.cfg` file contains additional keys you can edit to customize the patching operation, and comment lines describing the purpose of each key and how to set its value. Here is a sample `dbpatchm.cfg` file showing these keys and comments.

```
# sample config file

# oss storage container url with public access, normally should not be changed
# there is a default for this now, that is set to production container
oss_container_url=""

# change this following golden image zip/psu zip file to be used in patching
# this will be used only if you run dbaascli/dbpatchm directly
# keep these files in different location from temporary_space given below & /u01/psu
gold_img_loc="</tmp/db11.2.0.4.0_EE_PSU.tar.gz>"
psu_zip_loc="</tmp/p19121551_112040_Linux-x86-64.zip>"

# location where the temporary files will be kept - should have 15GB space minimum
# change this location if needed but do not use /u01/psu - used for conflict check
temporary_space="/u01/download";

# turn this ignore_patch_conflict to 1 to let patching ignore conflicts
ignore_patch_conflict=0

# turn this ignore_space_less_than_15g to 1 to let patching proceed if discspace <
15g
ignore_space_less_than_15g=0

# create /var/opt/oracle/patch/files_to_save.ora with full path of directory or
# files to preserve any special files you may have in your /u01/app directory.
# set this to yes, if you have files_to_save.ora
special_files="no"

# type could be "psu", "upg"
type="psu"
# psunum is the bugid for PSU's - needed to rollback the psu patches
psunum=

# patching method could be cloning (for temporary instance validation) or psu way
# when cloning is yes, remote node IP needs to be provided and also sys password
cloning="no"
remotenode=""
syspasswd=""

# ssh private key needed for cloning that needs a 2nd VM connectivity
# "/root/.ssh/pat.key"
sshkey_fl=""
```

```
# oracle recovery manager catalog connect string - if configured
rcatconnect=""

# data guard patching - need to provide primary and standby ip's and
# private sshkeys to connect to the same
# if dg_inst="yes", then all dg_ parameters need to be provided
dg_inst="no"
dg_primary=""
dg_primary_sshkey_fl=""
dg_standby=""
dg_standby_sshkey_fl=""
```

8

Configuring Database Features, Database Options, and Companion Products

Oracle Database Cloud Service provides special capabilities for certain Oracle Database features and options and for certain companion products.

Topics

Using Oracle Real Application Clusters (RAC) in Database Cloud Service


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

You can create an Oracle RAC database when creating a database deployment on Oracle Database Cloud Service.

Oracle RAC enhances Oracle Database capabilities so that you can concurrently use multiple database instances on different compute nodes. This allows you to scale workload across multiple database instances in order to efficiently store, update, and retrieve data.

Oracle RAC provides the software that manages multiple servers and database instances as a single set of servers, called a cluster. The data files that comprise the database reside on shared storage that is accessible from all servers that are part of the cluster. Each server in the cluster runs the Oracle RAC software.

Unlike a single-instance Oracle database, which has a one-to-one relationship between data files and the database instance, Oracle RAC databases have a one-to-many relationship between data files and database instances. This means that in an

Oracle RAC database multiple database instances access a single set of database files concurrently, allowing you to access the data from any database instance in the database cluster.

This allows you to use horizontal scalability beyond the scope of one compute node, in case this compute node is insufficient to run the desired workload. It also increases availability of the database and the data in case a database instance or compute node fails. The remaining database instance can be used to continue operations while the failed database instance or compute node is being restarted. Having more than one database instance also allows you to perform rolling patch upgrades.

> **Note:**
>
> To learn about using Oracle Real Application Clusters and Oracle Data Guard together, see Using Oracle Real Application Clusters (RAC) and Oracle Data Guard Together in Database Cloud Service.

To create an Oracle RAC database in Database Cloud Service, make the following choices in the Create Instance wizard:

- For Software Edition, choose **Enterprise Edition - Extreme Performance**.
- For Database Type, choose **Database Clustering with RAC**.
- For Compute Shape (on the Service Details page), choose a shape with two or more OCPUs.

When you make these choices, Database Cloud Service creates a two-node Oracle RAC database, hosting the database on two independent compute nodes that share data, fast recovery area, and redo log storage. It creates these compute nodes using computing, storage and networking resources provided by Oracle Cloud Infrastructure Compute Classic.

Managing a Deployment that Uses Oracle RAC Compared to a Deployment that Doesn't

Because a Database Cloud Service deployment that uses Oracle RAC comprises two compute nodes that each host a RAC database instance, you manage the deployment in slightly different ways:

- Cloud tooling: you use `raccli` instead of `bkup_api` or `dbaascli`. For more information, see The raccli Utility.
- You can stop and start the database instances and even the compute nodes independently of each other. Thus, the database can remain available even when you need to perform maintenance that requires you to stop a database instance or compute node.

More About the Oracle RAC Configuration on Database Cloud Service

- Cluster size: currently, the Oracle RAC database on a Database Cloud Service deployment is limited to a two-node cluster.
- Cloud tooling: Oracle Cloud tooling is provided for the common administrative tasks of scaling, backing up and recovering, and patching. For more information, see Scaling a Database Deployment and The raccli Utility.

- Network access: on the compute nodes, access to all ports except port 22 is disabled. Port 22 is open for passwordless, key-based SSH access by the `opc` user. To enable access to other ports, see Enabling Access to a Compute Node Port.
- Networking for client access: to make client connections to the Oracle RAC database, you include particular options in the connection's entry in the client's `tnsnames.ora` file. For more information, see Connecting Remotely to the Database by Using Oracle Net Services.
- Database file storage: storage for database data files, the fast recovery area, and the redo logs is created and managed using Oracle Automatic Storage Management (ASM) and Oracle Automatic Storage Management Cluster File System (ACFS) instead of Linux LVM.
- Included software: Oracle Grid Infrastructure, Oracle ASM and Oracle ACFS are included; Oracle Application Express, Oracle REST Data Services, and Oracle SQL Developer Web are not currently included.

Using Oracle Data Guard in Database Cloud Service

When creating an Oracle Database Cloud Service database deployment, you can create an Oracle Data Guard configuration.
Oracle Data Guard enables Oracle databases to survive disasters and data corruptions by providing a comprehensive set of services that create, maintain, manage, and monitor a standby database. Oracle Data Guard maintains the standby database as a copy of the primary database. If the primary database becomes unavailable because of a planned or an unplanned outage, you can switch the standby database to the primary role, minimizing the downtime associated with the outage.

> **Note:**
>
> To learn about using Oracle Data Guard and Oracle Real Application Clusters together, see Using Oracle Real Application Clusters (RAC) and Oracle Data Guard Together in Database Cloud Service.

Creating an Oracle Data Guard Configuration

> **Note:**
>
> To learn about creating an Oracle Data Guard configuration with a primary database on your own system and a standby database in the cloud, see Creating a Hybrid DR Deployment.

To create an Oracle Data Guard configuration in Database Cloud Service, make the following choices in the Create Instance wizard:

- For Software Edition, choose **Enterprise Edition**, **Enterprise Edition - High Performance**, or **Enterprise Edition - Extreme Performance**.
- For Database Type, choose **Single Instance with Data Guard Standby**.

- For Standby Database Configuration (on the Instance Details page), choose where you want the standby database placed in relation to the primary database:
  - **High Availability**—The standby database is placed in a different availability domain from the primary database, thus providing isolation at the infrastructure level.
  - **Disaster Recovery**—The standby database is placed in a different data center from the primary database, thus providing isolation at the infrastructure level and geographical separation to support availability despite catastrophic events.

For further details, see Creating a Customized Database Deployment.

When you make these choices, Database Cloud Service creates an Oracle Data Guard configuration with a primary database and a single standby database, hosting the databases on two independent compute nodes. It creates these compute nodes using computing, storage and networking resources provided by Oracle Cloud Infrastructure Compute Classic or Oracle Cloud Infrastructure.

If you selected Enterprise Edition - Extreme Performance, the configuration includes Oracle Active Data Guard. See "Opening a Physical Standby Database" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1, or 11.2 for more information on the real-time query and the automatic block media recovery features of Oracle Active Data Guard.

> **Note:**
>
> Oracle Database Cloud Service does not currently include the fast-start failover (FSFO) feature of Oracle Data Guard. In Database Cloud Service, you perform failover operations manually, as described in Performing a Manual Failover Operation.

In a Database Cloud Service database deployment, you can use the Oracle Database Cloud Service console or the `dataguard` subcommand of the `dbaascli` utility to perform many Data Guard operations. For more information, see Administering a Data Guard Configuration.

You can also manage primary and standby databases by using the SQL command-line interface or the Oracle Data Guard broker interfaces. The broker provides a command-line interface (DGMGRL) and a graphical user interface through Oracle Enterprise Manager Cloud Control.

> **Note:**
>
> You must use the `database changepassword` subcommand of the `dbaascli` utility to change the password of the `SYS` user in your Oracle Data Guard configuration to ensure that the password file on the standby database node is updated and cloud tooling works correctly. See Changing the SYS Password for details.

More About Oracle Data Guard

Oracle Data Guard provides a comprehensive set of services that create, maintain, manage, and monitor a standby database to enable Oracle databases to survive disasters and data corruptions.

An Oracle Data Guard configuration contains one primary database, which is the database that is accessed by most of your applications. An Oracle Data Guard configuration also contains up to thirty standby destinations, connected by Oracle Net Services. However, the Oracle Data Guard configuration in Database Cloud Service includes one primary database and one standby database.

A standby database is a transactionally consistent copy of the primary database. Once created, Oracle Data Guard automatically maintains each standby database by transmitting redo data from the primary database and then applying the redo to the standby database. In an Oracle Data Guard configuration on Database Cloud Service, the standby database is a physical standby database. A physical standby database provides a physically identical copy of the primary database, with on-disk database structures that are identical to the primary database on a block-for-block basis. A physical standby database is kept synchronized with the primary database, through Redo Apply, which recovers the redo data received from the primary database and applies the redo to the physical standby database.

See "Oracle Data Guard Configurations" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2 for additional information.

Enterprise Edition - Extreme Performance includes Oracle Active Data Guard. Oracle Active Data Guard provides read-only access to the physical standby database while it is synchronized with the primary database, enabling minimal latency between reporting and transactional data. With the Oracle Active Data Guard feature known as real-time query, Redo Apply can be active while the physical standby database is open, thus allowing queries to return results that are identical to what would be returned from the primary database. See "Opening a Physical Standby Database" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2 for additional information about real-time query.

Using Oracle Real Application Clusters (RAC) and Oracle Data Guard Together in Database Cloud Service


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

When creating a database deployment on Oracle Database Cloud Service, you can create a pair of Oracle RAC databases linked together as the primary and standby databases of an Oracle Data Guard configuration.

> **Note:**
>
> Oracle Database Cloud Service does not currently include the fast-start failover (FSFO) feature of Oracle Data Guard. In Database Cloud Service, you perform failover operations manually, as described in Performing a Manual Failover Operation.

To create a Database Cloud Service database deployment that uses Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard, make the following choices in the Create Instance wizard:

- For Software Edition, choose **Enterprise Edition - Extreme Performance**.
- For Database Type, choose **Database Clustering with RAC and Data Guard Standby**.
- For Compute Shape (on the Instance Details page), choose a shape with two or more OCPUs.
- For Standby Database Configuration (on the Instance Details page), choose where you want the standby database placed in relation to the primary database:
  - **High Availability**—The standby database is placed in a different availability domain from the primary database, thus providing isolation at the infrastructure level.
  - **Disaster Recovery**—The standby database is placed in a different data center from the primary database, thus providing isolation at the infrastructure level and geographical separation to support availability despite catastrophic events.

When you make these choices, Database Cloud Service creates two two-node cluster databases using Oracle RAC, one acting as the primary database and one acting as a physical standby database in an Oracle Data Guard configuration. In each cluster database, the two cluster nodes share data, fast recovery area, and redo log storage. Database Cloud Service creates these compute nodes using computing, storage and networking resources provided by Oracle Cloud Infrastructure Compute Classic.

Because the Software Edition is Enterprise Edition - Extreme Performance, the deployment includes Oracle Active Data Guard. For information on the real-time query and the automatic block media recovery features of Oracle Active Data Guard, see "Opening a Physical Standby Database" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2.

More About a Deployment that Uses Oracle RAC and Oracle Data Guard Together

- Cluster size: currently, the Oracle RAC database on a Database Cloud Service deployment is limited to a two-node cluster.
- Cloud tooling: Oracle Cloud tooling is provided for the common administrative tasks of scaling, backing up and recovering, and patching. For more information, see The raccli Utility.
- Data Guard operations: to perform operations such as switchover, failover and reinstate, you can use the Oracle Database Cloud Service console or you can use subcommands of the `raccli` utility. For information, see Administering a Data Guard Configuration and The raccli Utility.

- Updating passwords: use the raccli update databasepassword command to update the password in the keystore (wallet) and to update the password for database users.
- Network access: on the compute nodes, access to all ports except port 22 is disabled. Port 22 is open for passwordless, key-based SSH access by the `opc` user. To enable access to other ports, see Enabling Access to a Compute Node Port.
- Networking for client access: to make client connections to one of the Oracle RAC databases, you include particular options in the connection's entry in the client's `tnsnames.ora` file. For more information, see Connecting Remotely to the Database by Using Oracle Net Services.
- Database file storage: for each Oracle RAC database, storage for database data files, the fast recovery area, and the redo logs is created and managed using Oracle Automatic Storage Management (ASM) and Oracle Automatic Storage Management Cluster File System (ACFS) instead of Linux LVM.
- Included software: Oracle Application Express, Oracle REST Data Services, and Oracle SQL Developer Web are not currently included.

For more detailed information, see Characteristics of a Database Clustering with RAC and Data Guard Standby Database Deployment.

Using Oracle Multitenant in Database Cloud Service

When you create an Oracle Database Cloud Service database deployment that uses Oracle Database 12c or later, an Oracle Multitenant environment is created.

The multitenant architecture enables an Oracle database to function as a multitenant container database (CDB) that includes zero, one, or many pluggable databases (PDBs). A PDB is a portable collection of schemas, schema objects, and non-schema objects that appears to an Oracle Net Services client as a non-CDB. All Oracle databases before Oracle Database 12c were non-CDBs.

Topics

- Creating and Activating a Master Encryption Key for a PDB
- Exporting and Importing a Master Encryption Key for a PDB

Creating and Activating a Master Encryption Key for a PDB

To use Oracle Transparent Data Encryption (TDE) in a pluggable database (PDB), you must create and activate a master encryption key for the PDB.

In a multitenant environment, each PDB has its own master encryption key which is stored in a single keystore used by all containers.

To determine whether you need to create and activate an encryption key for the PDB, perform the following steps:

1. Invoke SQL*Plus and log in to the database as the `SYS` user with `SYSDBA` privileges.
2. Set the container to the PDB:

```
SQL> ALTER SESSION SET CONTAINER = pdb;
```

3. Query V$ENCRYPTION_WALLET as follows:

```
SQL> SELECT wrl_parameter, status, wallet_type FROM v$encryption_wallet;
```

If the STATUS column contains a value of OPEN_NO_MASTER_KEY you need to create and activate the master encryption key.

To create and activate the master encryption key in a PDB, perform the following steps:

1. Set the container to the PDB:

```
SQL> ALTER SESSION SET CONTAINER = pdb;
```

2. Create and activate a master encryption key in the PDB by executing the following command:

```
SQL> ADMINISTER KEY MANAGEMENT SET KEY USING TAG 'tag' FORCE KEYSTORE IDENTIFIED
BY keystore-password WITH BACKUP USING 'backup_identifier';
```

In the above command:

- *keystore-password* is the keystore password. By default, the keystore password is set to the value of the administration password that is specified when the database deployment is created.
- The optional USING TAG '*tag*' clause can be used to associate a tag with the new master encryption key.
- The WITH BACKUP clause, and the optional USING '*backup_identifier*' clause, can be used to create a backup of the keystore before the new master encryption key is created.

See also ADMINISTER KEY MANAGEMENT in *Oracle Database SQL Language Reference* for Release 18 or 12.2.

> **Note:**
>
> To enable key management operations while the keystore is in use, Oracle Database 12c Release 2, and later, includes the FORCE KEYSTORE option to the ADMINISTER KEY MANAGEMENT command. This option is also available for Oracle Database 12c Release 1 with the October 2017, or later, bundle patch.
>
> If your Oracle Database 12c Release 1 deployment does not have the October 2017, or later, bundle patch installed, you can perform the following alternative steps:
>
> - Close the keystore.
> - Open the password-based keystore.
> - Create and activate a master encryption key in the PDB by using ADMINISTER KEY MANAGEMENT without the FORCE KEYSTORE option.
> - Update the auto-login keystore by using ADMINISTER KEY MANAGEMENT with the CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE option.

3. Query V$ENCRYPTION_WALLET again to verify that the STATUS column is set to OPEN:

```
SQL> SELECT wrl_parameter, status, wallet_type FROM v$encryption_wallet;
```

Exporting and Importing a Master Encryption Key for a PDB

You must export and import the master encryption key for any encrypted PDBs you plug in to your database deployment.

If your source PDB is encrypted, you must export the master encryption key and then import it. In a multitenant environment, each PDB has its own master encryption key which is stored in a single keystore used by all containers.

You can export and import all of the TDE master encryption keys that belong to the PDB by exporting and importing the TDE master encryption keys from within a PDB. Export and import of TDE master encryption keys support the PDB unplug and plug operations. During a PDB unplug and plug, all of the TDE master encryption keys that belong to a PDB, as well as the metadata, are involved.

See "Exporting and Importing TDE Master Encryption Keys for a PDB" in *Oracle Database Advanced Security Guide* for Release 18, 12.2 or 12.1.

See "ADMINISTER KEY MANAGEMENT" in *Oracle Database SQL Language Reference* for Release 18, 12.2 or 12.1.

To export the master encryption keys, perform the following steps:

1. Invoke SQL*Plus and log in to the PDB.
2. Export the master encryption key by executing the following command:

```
SQL> ADMINISTER KEY MANAGEMENT EXPORT ENCRYPTION KEYS WITH SECRET "secret" TO
'filename' IDENTIFIED BY keystore-password;
```

To import the master encryption key perform the following steps:

1. Invoke SQL*Plus and log in to the PDB.
2. Export the master encryption key by executing the following command:

```
SQL> ADMINISTER KEY MANAGEMENT IMPORT ENCRYPTION KEYS WITH SECRET "secret" FROM
'filename' IDENTIFIED BY keystore-password;
```

Using Oracle Database Vault in Database Cloud Service

You can use Oracle Database Vault in an Oracle Database Cloud Service database deployment.

Oracle Database Vault provides powerful security controls to help protect application data from unauthorized access, and comply with privacy and regulatory requirements.

You can deploy controls to block privileged account access to application data and control sensitive operations inside the database. Trusted paths can be used to add additional security controls to authorized data access and database changes. Through the runtime analysis of privileges and roles, you can increase the security of existing applications by implementing least privileges and reducing the attack profile of your database accounts. Oracle Database Vault secures existing database environments transparently, eliminating costly and time consuming application changes.

The information in this document tells you about enabling and disabling Oracle Database Vault in an Oracle Database Cloud Service database deployment, but does not provide detail on using the features of Oracle Database Vault. Be sure to refer to

Oracle Database Vault Administrator's Guide for Release 18, 12.2, 12.1 or 11.2 for detailed information on implementing Oracle Database Vault features.

Topics

- Configuring and Enabling Oracle Database Vault
- Disabling Oracle Database Vault

Configuring and Enabling Oracle Database Vault

You can use the `dv on` subcommand of the `dbaascli` utility to configure and enable Database Vault with your database.

Oracle Database includes Database Vault, but you must configure and enable it before you can use it.

The `dbaascli` utility provides an easy-to-use interface for configuring and enabling Database Vault. As an alternative to using `dbaascli`, you can follow the steps in "Getting Started with Oracle Database Vault" in *Oracle Database Vault Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2.

Be sure to review "What to Expect After You Enable Oracle Database Vault" in *Oracle Database Vault Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2 to gain an understanding of the impact of enabling and configuring Database Vault.

As part of the configuration process, Database Vault administrative accounts are created. Oracle strongly recommends that you create two accounts for each role. One account, the primary account, will be used on a day-to-day basis and the other account will be used as a backup account in case the password of the primary account is lost and must be reset.

Refer to dbaascli dv on for additional information about the `dv on` subcommand, including options that can be used to enable Database Vault only for the root container (CDB) or a specified pluggable database (PDB) in a database deployment using Oracle Database 12c or later.

To enable and configure Database Vault by using the `dv on` subcommand:

1. Connect to the compute node as the **`oracle`** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Enable and configure Database Vault:

```
$ dbaascli dv on
...
Enter DV owner username: DVownerusername
Enter DV owner password: DVownerpassword
Re-enter DV owner password: DVownerpassword
Enter DV manager username: DVmanagerusername
Enter DV manager password: DVmanagerpassword
Re-enter DV manager password: DVmanagerpassword
...
Successfully configured DV
$
```

   Enter a user name and password for the Database Vault Owner and Database Vault Account Manager when prompted. In a database deployment using Oracle

Database 12c or later, the Database Vault Owner and Account Manager user names must begin with `c##`.

3. Disconnect from the compute node.

Disabling Oracle Database Vault

You can use the `dv off` subcommand of the `dbaascli` utility to disable Database Vault in your database.

The `dbaascli` utility provides an easy-to-use interface for disabling Database Vault. As an alternative to using `dbaascli`, you can follow the steps in "Disabling and Enabling Oracle Database Vault" in *Oracle Database Vault Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2.

When you install Oracle Database Vault, it revokes a set of privileges from several Oracle Database-supplied users and roles. Be aware that if you disable Oracle Database Vault, these privileges remain revoked. See "Privileges That Are Revoked from Existing Users and Roles" in *Oracle Database Vault Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2 for additional information.

Refer to dbaascli dv off for additional information about the `dv off` subcommand, including options to disable Database Vault for only the root container (CDB) or a specific pluggable database (PDB) in a database deployment using Oracle Database 12c or later.

To enable and configure Database Vault by using the `dv off` subcommand:

1. Connect to the compute node as the **`oracle`** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Disable Database Vault:

```
$ dbaascli dv off
...
Enter DV owner username: DVownerusername
Enter DV owner password: DVownerpassword
...
Successfully configured DV
$
```

   Enter the user name and password for the Database Vault Owner when prompted.

3. Disconnect from the compute node.

Using Oracle Application Express in Database Cloud Service

Database deployments on Oracle Database Cloud Service include Oracle Application Express.

> **Note:**
>
> This section does not apply to database deployments that use Oracle Real Application Clusters. Such deployments do not currently include Oracle Application Express.

Oracle Application Express enables you to design, develop and deploy responsive, database-driven applications using only your web browser. If you are new to Oracle Application Express, see its Overview and Getting Started pages on Oracle Technology Network to learn about its features and get started using it.

This section provides guidance on how to navigate to Oracle Application Express from Oracle Database Cloud Service and how to upgrade Oracle Application Express releases residing in Database Cloud Service.

Topics

Accessing the Oracle Application Express Console

Database deployments of single-instance databases on Oracle Database Cloud Service include Oracle Application Express, which you manage using the Oracle Application Express Instance Administration console.

If you are new to Oracle Application Express, see its Overview and Getting Started pages on Oracle Technology Network to learn about its features and get started using it.

You can access the Oracle Application Express Instance Administration console in the following ways:

- Using the “Open Application Express Console” menu item
- Using a direct URL
- Using an SSH tunnel

Using the “Open Application Express Console” Menu Item to Access the Console

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access the Oracle Application Express Instance Administration console is blocked by default. To use the **Open Application Express Console** menu item, you must unblock port 443, either by enabling the deployment’s **ora_p2_httpssl** predefined access rule or by creating your own access rule that opens port 443. For instructions, see Enabling Access to a Compute Node Port.

1. Open the Services page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. From the menu for the deployment, select **Open Application Express Console**.
3. If your browser displays a warning that your connection is not secure or not private, use the browser’s advanced option to continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.
4. When prompted for a workspace, username and password, enter the following information. Then click **Sign In**.
   - In the workspace box, enter **INTERNAL**.
   - In the username box, enter **ADMIN**.
   - In the password box, enter the password specified during the database deployment creation process.

     Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

The **Open Application Express Console** menu item is also available on the Overview page in the menu next to the deployment’s name.

Using a Direct URL to Access the Console

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access the Oracle Application Express Instance Administration console is blocked by default. To use a direct URL, you must unblock port 443, either by enabling the deployment's **ora_p2_httpssl** predefined access rule or by creating your own access rule that opens port 443. For instructions, see Enabling Access to a Compute Node Port.

1. In your web browser, go to the URL appropriate to the Oracle Database version on the database deployment:
   - For a PDB in an Oracle Database 12*c* database: `https://node-ip-address/ords/lowercase-pdb-name/`
   - For an Oracle Database 11*g* Release 2 database: `https://node-ip-address/apex/`

   where *compute-node-ip-address* is the IP address of the deployment's compute node as listed on the deployment's Overview page, and *lowercase-pdb-name* is the name of the PDB, with all letters in lowercase.
2. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.
3. When prompted for a workspace, username and password, enter the following information. Then click **Sign In**.
   - In the workspace box, enter **INTERNAL**.
   - In the username box, enter **ADMIN**.
   - In the password box, enter the password specified during the database deployment creation process.

     Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

Using an SSH Tunnel to Access the Console

1. Create an SSH tunnel to port 443 on the compute node hosting Oracle Application Express. For information about creating an SSH tunnel, see Creating an SSH Tunnel to a Compute Node Port.
2. After creating the SSH tunnel, direct your browser to the URL appropriate to the Oracle Database version on the database deployment:
   - For a PDB in an Oracle Database 12*c* database: `https://localhost/ords/lowercase-pdb-name/`
   - For an Oracle Database 11*g* Release 2 database: `https://localhost/apex/`

where *lowercase-pdb-name* is the name of the PDB, with all letters in lowercase.

3. When prompted for a workspace, username and password, enter the following information. Then click **Sign In**.
   - In the workspace box, enter **INTERNAL**.
   - In the username box, enter **ADMIN**.
   - In the password box, enter the password specified during the database deployment creation process.

     Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

Upgrading from Oracle Application Express 4.2 or 5.0 to 5.1 for Oracle Database 11*g*

These instructions are applicable if you have an Oracle Database 11*g* database deployment and want to upgrade Oracle Application Express 4.2 or Oracle Application Express 5.0 to Oracle Application Express 5.1.

To upgrade from Oracle Application Express 4.2 or Oracle Application Express 5.0 to 5.1:

1. Determine the version of your current Oracle Application Express installation:
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user and query `DBA_REGISTRY`:

      ```
      $ sqlplus / as sysdba

      SQL> SELECT VERSION FROM DBA_REGISTRY WHERE COMP_ID = 'APEX';
      ```

   c. Log out of SQL*Plus.
   d. Log out of the compute node.
2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the `opc` user.
   b. Check the version of the cloud tooling:

      ```
      rpm -q dbaastools
      ```

      You should see something similar to this: `dbaastools-1.0-1+17.3.1.0.0_170605.2102.x86_64`. Check the value between the + and _ for the version number. The tooling version must be `17.2.5.0.0` or higher. If your cloud tooling is at a lower version, see Updating the Cloud Tooling on Database Cloud Service for the steps to update to a later version.
3. Upload Oracle Application Express 5.1 to the compute node:
   a. Download Oracle Application Express 5.1.0.00.45 from Oracle Technology Network.
   b. Log in to the compute node as the `oracle` user.
   c. Use `scp` or `sftp` to upload the Oracle Application Express 5.1 zip file to the `/tmp` directory on the compute node.

4. Unzip the Oracle Application Express 5.1 zip file:
   a. Create the directory required for Oracle Application Express:

      ```
      mkdir -p /u01/app/oracle/product/apex/
      ```

   b. Change to the `/tmp` directory where you uploaded the Oracle Application Express zip file.
   c. Unzip the uploaded Oracle Application Express zip file into the `apex` directory:

      ```
      unzip apex_5.1.zip -d /u01/app/oracle/product/apex/
      ```

5. Install Oracle Application Express 5.1:
   a. Move the Oracle Application Express files to the `/u01/app/oracle/product/apex/5.1.0.00.45/` directory:

      ```
      mv /u01/app/oracle/product/apex/apex/ /u01/app/oracle/product/apex/
      5.1.0.00.45/
      ```

   b. Change to the new `5.1.0.00.45` directory:

      ```
      cd /u01/app/oracle/product/apex/5.1.0.00.45
      ```

   c. Log in to SQL*Plus as the `SYS` user and execute the installation script:

      ```
      sqlplus / as SYSDBA @apexins.sql SYSAUX SYSAUX TEMP /i/
      ```

   d. Log out of the compute node.
6. Configure Oracle Application Express by executing the Oracle REST Data Services (ORDS) assistant:
   a. Log in to the compute node as the `opc` user.
   b. Start a root-user shell:

      ```
      $ sudo -s
      ```

   c. Change to the `ords` directory:

      ```
      cd /var/opt/oracle/ocde/assistants/ords
      ```

   d. Execute the ORDS assistant:

      ```
      ./ords -out="/var/opt/oracle/ocde/res/ords.out" -
      ords_action="configure_apex"
      ```

   e. Restart ORDS:

      ```
      /etc/init.d/ords restart
      ```

   f. Exit the root-user shell and log out of the compute node.
7. If you upgraded from Oracle Application Express version 4.2, perform the following steps:
   a. Open the Oracle Database Cloud Service console.
   b. From the  menu for the deployment, select **Open Application Express Console**.

      The Oracle Application Express login page is displayed.
   c. Enter the following information to log in. Then click **Sign In**.
      - In the Workspace box, enter **INTERNAL**.

- In the Username box, enter **ADMIN**.
- In the Password box, enter the password specified during the database deployment creation process.

The Oracle Application Express Instance Administration page is displayed.

d. Locate Instance Settings and click on the pencil icon to edit the settings.

e. Select Report Printing.

f. In the Print Server field, select Oracle REST Data Services from the menu.

g. Click Apply Changes.

h. Log out of the Oracle Application Express Administration Services application.

Upgrading from Oracle Application Express 4.2 to 5.1 for Oracle Database 12*c*

These instructions are applicable if you have an Oracle Database 12*c* Release 1 or Release 2 database deployment and want to upgrade Oracle Application Express 4.2 in the root container (CDB$ROOT) to Oracle Application Express 5.1 in the pluggable databases (PDBs).

To upgrade from Oracle Application Express 4.2 in the root container (CDB$ROOT) to Oracle Application Express 5.1 in the PDBs:

1. Determine the version of Oracle Application Express installed in the root container (CDB$ROOT):
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user and query `DBA_REGISTRY`:

      ```
      $ sqlplus / as sysdba

      SQL> SELECT VERSION FROM DBA_REGISTRY WHERE COMP_ID = 'APEX';
      ```

      The version should be 4.2.5.00.08. If the version is 5.0.0.00.31 or 5.0.4.00.12, see Upgrading from Oracle Application Express 5.0 to 5.1 for Oracle Database 12*c*.
   c. Log out of SQL*Plus.
   d. Log out of the compute node.
2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the `opc` user.
   b. Check the version of the cloud tooling:

      ```
      rpm -q dbaastools
      ```

      You should see something similar to this: `dbaastools-1.0-1+17.3.1.0.0_170605.2102.x86_64`. Check the value between the + and _ for the version number. The tooling version must be `17.2.5.0.0` or higher. If your cloud tooling is at a lower version, see Updating the Cloud Tooling on Database Cloud Service for the steps to update to a later version.
3. Upload Oracle Application Express 5.1 to the compute node:
   a. Download Oracle Application Express 5.1.0.00.45 from Oracle Technology Network.

b. Log in to the compute node as the `oracle` user.

c. Use `scp` or `sftp` to upload the Oracle Application Express 5.1 zip file to the `/tmp` directory on the compute node.

4. Unzip the Oracle Application Express 5.1 zip file:

a. Create the directory required for Oracle Application Express:

```
mkdir -p /u01/app/oracle/product/apex/
```

b. Change to the `/tmp` directory where you uploaded the Oracle Application Express zip file.

c. Unzip the uploaded Oracle Application Express zip file into the `apex` directory:

```
unzip apex_5.1.zip -d /u01/app/oracle/product/apex/
```

d. Move the Oracle Application Express files to the `/u01/app/oracle/product/apex/5.1.0.00.45/` directory:

```
mv /u01/app/oracle/product/apex/apex/ /u01/app/oracle/product/apex/
5.1.0.00.45/
```

5. Determine which PDBs have an APEX instance:

a. Log in to the compute node as the `oracle` user.

b. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

c. Execute the query to determine which PDBs have an APEX instance:

```
SQL> SELECT P.PDB_NAME, R.VERSION, R.STATUS

2 FROM SYS.DBA_PDBS P, SYS.CDB_REGISTRY R

3 WHERE P.PDB_ID = R.CON_ID AND R.COMP_ID = 'APEX'

4 ORDER BY 1;
```

6. In each PDB, identify the APEX workspaces. Be sure to make note of which workspaces belong to which PDBs. You will need this information in a later step.

a. While still logged in to SQL*Plus, connect to each PDB by using the `ALTER SESSION` command:

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

b. Query `APEX_WORKSPACES` to determine the APEX workspace names:

```
SQL> SELECT WORKSPACE_ID FROM APEX_WORKSPACES WHERE WORKSPACE_ID >
100;
```

c. After connecting to each PDB and compiling a list of workspace IDs, log out of SQL*Plus.

7. Download a script that will be used to export the artifacts of the workspaces identified in the previous step into zip files:

a. Create a directory for the workspace zip files:

```
$ mkdir -p /home/oracle/workspaces
```

b. Using `wget`, download the `workspace_export_4.2.sh` file from Oracle Storage Cloud Service.

```
wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-
usoracle29538/dbaas_patch/apex_upg5_1/workspace_export_4.2.sh or
```

```
wget https://a88717.storage.oraclecloud.com/v1/Storage-a88717/
dbaas_patch/apex_upg5_1/workspace_export_4.2.sh
```

c. Move the script to the `/home/oracle/workspaces` directory.

d. Change to the `workspaces` directory:

```
$ cd /home/oracle/workspaces
```

e. Assign permissions to the `workspace_export_4.2.sh` file:

```
$ chmod 755 /home/oracle/workspaces/workspace_export_4.2.sh
```

```
$ chown oracle:oinstall /home/oracle/workspaces/
workspace_export_4.2.sh
```

8. Temporarily unlock the `APEX_040200` user in the CDB and temporarily set the user's password to `oracle` or a password of your choice. The script that you execute in a later step will reset the password.

a. Log in to SQL*Plus as the `SYS` user:

```
$ sqlplus / as sysdba
```

b. Unlock the user and set the password:

```
SQL> ALTER USER APEX_040200 ACCOUNT UNLOCK IDENTIFIED BY password
CONTAINER = ALL;
```

c. Log out of SQL*Plus.

9. Use Listener Control to determine the service name for each PDB

```
$ lsnrctl status
```

You should see output similar to:

```
Service "pdb1.opcdbaas.oraclecloud.internal" has 1 instance(s).
```

```
Instance "orcl", status READY, has 1 handler(s) for this service...
```

10. For each workspace you identified in step 6, execute the `workspace_export_4.2.sh` script.

```
./workspace_export_5.0.sh //localhost:port_number/PDB_service_name
APEX_040200 password workspace_id
```

where:

- `port_number` is the listener port number you specified when you created the database deployment (default is 1521).
- `PDB_service_name` is one of the service names you identified in step 9.
- `password` is the password you set for the `APEX_040200` user in step 8b.
- `workspace_id` is one of the APEX workspaces for the PDB that you identified in step 6b.

One zip file will be created for each workspace.

11. Remove APEX 4.2 from the root container (CDB$ROOT).

a. Change to the directory that contains the APEX 4.2.5.00.08 files.

```
cd $ORACLE_HOME/apex/
```

b. Log in to SQL*Plus as the `SYS` user:

```
$ sqlplus / as sysdba
```

c. Set the _oracle_script parameter to TRUE.

```
SQL> ALTER SESSION SET "_oracle_script"=true;
```

d. Uninstall APEX from the root container (CDB$ROOT).

```
SQL> @apxremov.sql
```

e. Log out of SQL*Plus.

12. Uninstall APEX 4.2 from the CDB seed (PDB$SEED).

a. Change to the directory that contains the APEX 4.2.5.00.08 files.

```
cd $ORACLE_HOME/apex/
```

b. Log in to SQL*Plus as the SYS user:

```
$ sqlplus / as sysdba
```

c. Set the _oracle_script parameter to TRUE.

```
SQL> ALTER SESSION SET "_oracle_script"=true;
```

d. Close the PDB.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;
```

e. Open the PDB in READ WRITE mode.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ WRITE;
```

f. Uninstall APEX from the PDB.

```
SQL> @apxremov.sql
```

g. Set the container to the root container (CDB$ROOT).

```
SQL> ALTER SESSION SET CONTAINER = CDB$ROOT;
```

h. Close the PDB.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;
```

i. Open the PDB in READ ONLY mode.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
```

j. Log out of SQL*Plus.

13. Uninstall APEX 4.2 from the PDBs identified in step 5.

a. Change to the directory that contains the APEX 4.2.5.00.08 files.

```
cd $ORACLE_HOME/apex/
```

b. Log in to SQL*Plus as the SYS user:

```
$ sqlplus / as sysdba
```

c. Set the _oracle_script parameter to TRUE.

```
SQL> ALTER SESSION SET "_oracle_script"=true;
```

d. Set the container to the PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

e. Uninstall APEX from the PDB.

```
SQL> @apxremov.sql
```

f. After repeating steps 13d and 13e for each identified PDB, log out of SQL*Plus.

14. Stop ORDS.

```
/etc/init.d/ords stop
```

15. Delete the APEX users for the previous version of APEX installed in the root container (CDB$ROOT).

a. Log in to SQL*Plus as the SYS user:

```
$ sqlplus / as sysdba
```

b. Query DBA_USERS.

```
SQL> select username from dba_users where username like '%APEX_0%';
```

c. Exit from SQL*Plus.

d. Drop the users identified in step 13b.

```
$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
drop_apex_user -- --x'drop user apex_user cascade'
```

where apex_user is the value in the USERNAME column from step 15b.

16. Remove the APEX users from all containers.

```
$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
drop_apex_listener -- --x'drop user apex_listener cascade'

$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
drop_apex_rest_public_user -- --x'drop user apex_rest_public_user
cascade'

$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
drop_apex_public_user -- --x'drop user apex_public_user cascade'
```

17. Change to the new 5.1.0.00.45 directory

```
cd /u01/app/oracle/product/apex/5.1.0.00.45
```

18. Install APEX in the CDB seed (PDB$SEED).

a. Log in to SQL*Plus as the SYS user.

```
$ sqlplus / as sysdba
```

b. Set the mode of PDB$SEED to OPEN READ WRITE.

```
SQL> ALTER SESSION SET "_oracle_script"=true;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ WRITE;
```

c. Connect to PDB$SEED.

```
SQL> ALTER SESSION SET CONTAINER = PDB$SEED;
```

d. Install APEX 5.1 in the PDB$SEED container

```
SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/

SQL> exit
```

e. Again, log in to SQL*Plus as the SYS user.

```
$ sqlplus / as sysdba
```

f. Set the mode of PDB$SEED back to OPEN READ ONLY.

```
SQL> ALTER SESSION SET "_oracle_script"=true;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
```

g. Exit from SQL*Plus.

19. Install APEX in each PDB you identified in step 5c.

a. Log in to SQL*Plus as the SYS user.

```
$ sqlplus / as sysdba
```

b. Connect to the PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

c. Install APEX 5.1 in the PDB.

```
SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/
```

d. After repeating steps 19b and 19c for each PDB you identified in step 5c, exit from SQL*Plus.

e. Log out of the compute node.

20. Reconstruct public synonyms in the CDB seed (PDB$SEED).

a. Log in to SQL*Plus as the SYS user:

```
$ sqlplus / as sysdba
```

b. Set the _oracle_script parameter to TRUE.

```
SQL> ALTER SESSION SET "_oracle_script"=true;
```

c. Close the PDB.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;
```

d. Open the PDB in READ WRITE mode.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ WRITE;
```

e. Set the container to PDB$SEED.

```
SQL> ALTER SESSION SET CONTAINER = PDB$SEED;
```

f. Set CURRENT_SCHEMA to APEX_050100.

```
SQL> alter session set current_schema = APEX_050100;
```

g. Execute the procedure to drop the public synonyms.

```
SQL> exec wwv_flow_upgrade.drop_public_synonyms(p_drop_all =>
true);
```

h. Execute the procedure to re-create the public synonyms.

```
SQL> exec wwv_flow_upgrade.recreate_public_synonyms('APEX_050100');
```

i. Set the container to the root container (CDB$ROOT).

```
SQL> ALTER SESSION SET CONTAINER = CDB$ROOT;
```

j. Close the PDB.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;
```

k. Open the PDB in READ ONLY mode.

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
```

l. Log out of SQL*Plus.

21. Reconstruct public synonyms for each PDB you identified in step 5

a. Log in to SQL*Plus as the `SYS` user:

```
$ sqlplus / as sysdba
```

b. Set the container to the PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

c. Set `CURRENT_SCHEMA` to `APEX_050100`.

```
SQL> alter session set current_schema = APEX_050100;
```

d. Execute the procedure to drop the public synonyms.

```
SQL> exec wwv_flow_upgrade.drop_public_synonyms(p_drop_all =>
true);
```

e. Execute the procedure to re-create the public synonyms.

```
SQL> exec wwv_flow_upgrade.recreate_public_synonyms('APEX_050100');
```

f. Log out of SQL*Plus.

22. Configure Oracle Application Express by executing the Oracle REST Data Services (ORDS) assistant:

a. Log in to the compute node as the `opc` user.

b. Start a root-user shell:

```
$ sudo -s
```

c. Change to the `ords` directory:

```
cd /var/opt/oracle/ocde/assistants/ords
```

d. Execute the ORDS assistant:

```
./ords -out="/var/opt/oracle/ocde/res/ords.out" -
ords_action="install"
```

e. Exit the root-user shell and log out of the compute node.

23. Install each APEX workspace into a PDB.

a. Log in to the compute node as the `oracle` user.

b. Change to the `workspaces` directory where the zip files from step 10 are located.

```
cd /home/oracle/workspaces
```

c. Unzip each of the workspace zip files into its corresponding directory.

```
unzip install_workspace_id_date.zip -d workspace_id
```

d. Change to one of the newly created `/home/oracle/workspaces/workspace_id` directories.

e. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

f. Connect to the PDB where the workspace should be installed. Refer to the list you compiled in step 6 to ensure you install each workspace in the correct PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

g. Execute the installation script to install the APEX workspace in the PDB.

```
@install_workspace_id.sql
```

h. After executing steps 23d through 23g for each APEX workspace, exit SQL*Plus.

i. Log out of the compute node.

Upgrading from Oracle Application Express 5.0 to 5.1 for Oracle Database 12*c*

These instructions are applicable if you have an Oracle Database 12*c* Release 1 or Release 2 database deployment and want to upgrade Oracle Application Express 5.0 in the root container (CDB$ROOT) to Oracle Application Express 5.1 in the pluggable databases (PDBs).

To upgrade from Oracle Application Express 5.0 in the root container (CDB$ROOT) to Oracle Application Express 5.1 in the PDBs:

1. Determine the version of Oracle Application Express installed in the root container (CDB$ROOT):
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user and query `DBA_REGISTRY`:

      ```
      $ sqlplus / as sysdba

      SQL> SELECT VERSION FROM DBA_REGISTRY WHERE COMP_ID = 'APEX';
      ```

      The version should be 5.0.0.00.31 or 5.0.4.00.12. If the version is 4.2.5.00.08, see Upgrading from Oracle Application Express 4.2 to 5.1 for Oracle Database 12*c*.
   c. Log out of SQL*Plus.
   d. Log out of the compute node.
2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the `opc` user.
   b. Check the version of the cloud tooling:

      ```
      rpm -q dbaastools
      ```

      You should see something similar to this: `dbaastools-1.0-1+17.3.1.0.0_170605.2102.x86_64`. Check the value between the + and _ for the version number. The tooling version must be `17.2.5.0.0` or higher. If your cloud tooling is at a lower version, see Updating the Cloud Tooling on Database Cloud Service for the steps to update to a later version.
3. Upload Oracle Application Express 5.1 to the compute node:
   a. Download Oracle Application Express 5.1.0.00.45 from Oracle Technology Network.

b. Log in to the compute node as the `oracle` user.

c. Use `scp` or `sftp` to upload the Oracle Application Express 5.1 zip file to the `/tmp` directory on the compute node.

4. Unzip the Oracle Application Express 5.1 zip file:

a. Create the directory required for Oracle Application Express:

```
mkdir -p /u01/app/oracle/product/apex/
```

b. Change to the `/tmp` directory where you uploaded the Oracle Application Express zip file.

c. Unzip the uploaded Oracle Application Express zip file into the `apex` directory:

```
unzip apex_5.1.zip -d /u01/app/oracle/product/apex/
```

d. Move the Oracle Application Express files to the `/u01/app/oracle/product/apex/5.1.0.00.45/` directory:

```
mv /u01/app/oracle/product/apex/apex/ /u01/app/oracle/product/apex/
5.1.0.00.45/
```

5. Determine which PDBs have an APEX instance:

a. Log in to the compute node as the `oracle` user.

b. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

c. Execute the query to determine which PDBs have an APEX instance:

```
SQL> SELECT P.PDB_NAME, R.VERSION, R.STATUS

2 FROM SYS.DBA_PDBS P, SYS.CDB_REGISTRY R

3 WHERE P.PDB_ID = R.CON_ID AND R.COMP_ID = 'APEX'

4 ORDER BY 1;
```

6. In each PDB, identify the APEX workspaces. Be sure to make note of which workspaces belong to which PDBs. You will need this information in a later step.

a. While still logged in to SQL*Plus, connect to each PDB by using the `ALTER SESSION` command:

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

b. Query `APEX_WORKSPACES` to determine the APEX workspace names:

```
SQL> SELECT WORKSPACE_ID FROM APEX_WORKSPACES WHERE WORKSPACE_ID >
100;
```

c. After connecting to each PDB and compiling a list of workspace IDs, log out of SQL*Plus.

7. Download a script that will be used to export the artifacts of the workspaces identified in the previous step into zip files:

a. Create a directory for the workspace zip files:

```
$ mkdir -p /home/oracle/workspaces
```

b. Using `wget`, download the `workspace_export_5.0.sh` file from Oracle Storage Cloud Service.

```
wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-
usoracle29538/dbaas_patch/apex_upg5_1/workspace_export_5.0.sh or
```

```
wget https://a88717.storage.oraclecloud.com/v1/Storage-a88717/
dbaas_patch/apex_upg5_1/workspace_export_5.0.sh
```

c. Move the script to the `/home/oracle/workspaces` directory.

d. Change to the `workspaces` directory:

```
$ cd /home/oracle/workspaces
```

e. Assign permissions to the `workspace_export_5.0.sh` file:

```
$ chmod 755 /home/oracle/workspaces/workspace_export_5.0.sh
```

```
$ chown oracle:oinstall /home/oracle/workspaces/
workspace_export_5.0.sh
```

8. Temporarily unlock the `APEX_050000` user and temporarily set the user's password to `oracle` or a password of your choice. The script that you execute in a later step will reset the password.

a. Log in to SQL*Plus as the `SYS` user:

```
$ sqlplus / as sysdba
```

b. Unlock the user and set the password:

```
SQL> ALTER USER APEX_050000 ACCOUNT UNLOCK IDENTIFIED BY password
CONTAINER = ALL;
```

c. Log out of SQL*Plus.

9. Use Listener Control to determine the service name for each PDB

```
$ lsnrctl status
```

You should see output similar to:

```
Service "pdb1.opcdbaas.oraclecloud.internal" has 1 instance(s).
```

```
Instance "orcl", status READY, has 1 handler(s) for this service...
```

10. For each workspace you identified in step 6, execute the `workspace_export_5.0.sh` script.

```
./workspace_export_5.0.sh //localhost:port_number/PDB_service_name
APEX_050000 password workspace_id
```

where:

- `port_number` is the listener port number you specified when you created the database deployment (default is 1521).
- `PDB_service_name` is one of the service names you identified in step 9.
- `password` is the password you set for the `APEX_050000` user in step 8b.
- `workspace_id` is one of the APEX workspaces for the PDB that you identified in step 6b.

One zip file will be created for each workspace.

11. Remove APEX 5.0 from the root container (CDB$ROOT).

a. Change to the directory that contains the APEX 5.0.0.00.31 or 5.0.4.00.12 files.

```
cd $ORACLE_HOME/apex/
```

b. Uninstall APEX.

```
sqlplus / as SYSDBA @apxremov.sql
```

12. Stop ORDS.

    ```
    /etc/init.d/ords stop
    ```

13. Delete the APEX users for the previous version of APEX installed in the root container (CDB$ROOT).

    a. Log in to SQL*Plus as the SYS user:

    ```
    $ sqlplus / as sysdba
    ```

    b. Query DBA_USERS.

    ```
    SQL> select username from dba_users where username like '%APEX_0%';
    ```

    c. Exit from SQL*Plus.

    d. Drop the users identified in step 13b.

    ```
    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_user -- --x'drop user apex_user cascade'
    ```

    where apex_user is the value in the USERNAME column from step 13b.

14. Remove the APEX users from all containers.

    ```
    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_listener -- --x'drop user apex_listener cascade'

    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_rest_public_user -- --x'drop user apex_rest_public_user
    cascade'

    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_public_user -- --x'drop user apex_public_user cascade'
    ```

15. Change to the new 5.1.0.00.45 directory

    ```
    cd /u01/app/oracle/product/apex/5.1.0.00.45
    ```

16. Install APEX in the CDB seed (PDB$SEED).

    a. Log in to SQL*Plus as the SYS user.

    ```
    $ sqlplus / as sysdba
    ```

    b. Set the mode of the CDB seed (PDB$SEED) to OPEN READ WRITE.

    ```
    SQL> ALTER SESSION SET "_oracle_script"=true;

    SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

    SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ WRITE;
    ```

    c. Connect to PDB$SEED.

    ```
    SQL> ALTER SESSION SET CONTAINER = PDB$SEED;
    ```

    d. Install APEX 5.1 in the CDB seed (PDB$SEED) container

    ```
    SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/

    SQL> exit
    ```

    e. Again, log in to SQL*Plus as the SYS user.

    ```
    $ sqlplus / as sysdba
    ```

f. Set the mode of the CDB seed (PDB$SEED) back to OPEN READ ONLY.

```
SQL> ALTER SESSION SET "_oracle_script"=true;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
```

g. Exit from SQL*Plus.

17. Install APEX in each PDB you identified in step 5c.

a. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

b. Connect to the PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

c. Install APEX 5.1 in the PDB.

```
SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/
```

d. After repeating steps 17b and 17c for each PDB you identified in step 5c, exit from SQL*Plus.

e. Log out of the compute node.

18. Configure Oracle Application Express by executing the Oracle REST Data Services (ORDS) assistant:

a. Log in to the compute node as the `opc` user.

b. Start a root-user shell:

```
$ sudo -s
```

c. Change to the `ords` directory:

```
cd /var/opt/oracle/ocde/assistants/ords
```

d. Execute the ORDS assistant:

```
./ords -out="/var/opt/oracle/ocde/res/ords.out" -
ords_action="install"
```

e. Exit the root-user shell and log out of the compute node.

19. Install each APEX workspace into a PDB.

a. Log in to the compute node as the `oracle` user.

b. Change to the `workspaces` directory where the zip files from step 10 are located.

```
cd /home/oracle/workspaces
```

c. Unzip each of the workspace zip files into its corresponding directory.

```
unzip install_workspace_id_date.zip -d workspace_id
```

d. Change to one of the newly created `/home/oracle/workspaces/workspace_id` directories.

e. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

f. Connect to the PDB where the workspace should be installed. Refer to the list you compiled in step 6 to ensure you install each workspace in the correct PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

g. Execute the installation script to install the APEX workspace in the PDB.

```
@install_workspace_id.sql
```

h. After executing steps 19d through 19g for each APEX workspace, exit SQL*Plus.

i. Log out of the compute node.

Upgrading from Oracle Application Express 5.1.0 or 5.1.3 to 5.1.4 for Oracle Database 11*g*

These instructions are applicable if you have an Oracle Database 11*g* database deployment and want to upgrade Oracle Application Express 5.1.0.00.45 or Oracle Application Express 5.1.3.00.05 to Oracle Application Express 5.1.4.00.08.

To upgrade from Oracle Application Express 5.1.0.00.45 or Oracle Application Express 5.1.3.00.05 to Oracle Application Express 5.1.4.00.08:

1. Determine the version of your current Oracle Application Express installation:
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user and query `DBA_REGISTRY`:

      ```
      $ sqlplus / as sysdba
      ```

      ```
      SQL> SELECT VERSION FROM DBA_REGISTRY WHERE COMP_ID = 'APEX';
      ```

      If the APEX version is lower than 5.1.0.00.45, you must first upgrade to APEX 5.1. See Upgrading from Oracle Application Express 4.2 or 5.0 to 5.1 for Oracle Database 11*g* for details.
   c. Log out of SQL*Plus.
   d. Log out of the compute node.
2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the `opc` user.
   b. Check the version of the cloud tooling:

      ```
      $ rpm -q dbaastools
      ```

      You should see something similar to this: `dbaastools-1.0-1+18.2.1.0.0_xxxxxx.xxxx.x86_64`. Check the value between the + and _ for the version number. The tooling version must be `18.2.1.0.0` or higher. If your cloud tooling is at a lower version, see Updating the Cloud Tooling on Database Cloud Service for the steps to update to a later version.
3. Upload Oracle Application Express 5.1.4.00.08 to the compute node:
   a. Download Oracle Application Express 5.1.4.00.08 from Oracle Technology Network.
   b. Download Patch 26795231: PATCH SET FOR APPLICATION EXPRESS (PATCH SET VERSION 5.1.4).

c. Log in to the compute node as the `oracle` user.

d. Upload the Oracle Application Express 5.1.4 zip file to the `/tmp` directory on the compute node. See Copying Files to or from a Database Cloud Service Database Deployment for instructions on how to copy files to the compute node.

e. Upload the patch zip file to the `/tmp` directory on the compute node. See Copying Files to or from a Database Cloud Service Database Deployment for instructions on how to copy files to the compute node.

4. Unzip the patch zip file and the Oracle Application Express 5.1.4 zip file:

a. Unzip the patch zip file.

```
$ unzip p26795231_514_Generic.zip -d ./
```

b. Create the directory required for Oracle Application Express:

```
$ mkdir -p /u01/app/oracle/product/apex/
```

c. Change to the `/tmp` directory where you uploaded the Oracle Application Express zip file.

d. Unzip the uploaded Oracle Application Express zip file into the `apex` directory:

```
$ unzip apex_5.1.4.zip -d /u01/app/oracle/product/apex/
```

5. Install Oracle Application Express 5.1.4:

a. Move the Oracle Application Express files to the `/u01/app/oracle/product/apex/5.1.4.00.08/` directory:

```
$ mv /u01/app/oracle/product/apex/apex/ /u01/app/oracle/product/
apex/5.1.4.00.08/
```

b. Optionally, delete the `zip` files:

```
$ rm -f p26795231_514_Generic.zip
```

```
$ rm -f apex_5.1.4.zip
```

c. Change to the `patch` directory:

```
$ cd ./patch
```

d. Log in to SQL*Plus as the `SYS` user install the patch:

```
$ sqlplus / as SYSDBA
```

```
SQL> @apxpatch.sql
```

e. Verify the version of APEX.

```
SQL> SELECT VERSION FROM DBA_REGISTRY WHERE COMP_ID = 'APEX';
```

f. Exit from SQL*Plus and log out of the compute node.

6. Configure Oracle Application Express by executing the Oracle REST Data Services (ORDS) assistant:

a. Log in to the compute node as the `opc` user.

b. Start a root-user shell:

```
$ sudo -s
```

c. Change to the `ords` directory:

```
# cd /var/opt/oracle/ocde/assistants/ords
```

d. Execute the ORDS assistant:

```
# ./ords -out="/var/opt/oracle/ocde/res/ords.out" -
ords_action="configure_apex"
```

e. Restart ORDS:

```
# /etc/init.d/ords restart
```

f. Exit the root-user shell and log out of the compute node.

Upgrading from Oracle Application Express 5.1.0 or 5.1.3 to 5.1.4 for Oracle Database 12*c* and Oracle Database 18*c*

These instructions are applicable if you have an Oracle Database 12*c* or Oracle Database 18*c* database deployment and want to upgrade Oracle Application Express 5.1.0.00.45 or Oracle Application Express 5.1.3.00.05 to Oracle Application Express 5.1.4.00.08.

To upgrade from Oracle Application Express 5.1.0.00.45 or Oracle Application Express 5.1.3.00.05 to Oracle Application Express 5.1.4.00.08:

1. Determine the version of your current Oracle Application Express installation:
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user.

      ```
      $ sqlplus / as sysdba
      ```

   c. If the Oracle Database version is 12.1.0.2, execute the following command to include `PDB$SEED` in queries:

      ```
      SQL> ALTER SESSION SET "EXCLUDE_SEED_CDB_VIEW"=FALSE;
      ```

   d. If the Oracle Database version is 12.2.0.1 or 18.1.0.0.0, execute the following command to include `PDB$SEED` in queries:

      ```
      SQL> ALTER SESSION SET "_EXCLUDE_SEED_CDB_VIEW"=FALSE;
      ```

   e. Execute the following query:

      ```
      SQL> SELECT R.CON_ID, P.NAME, R.VERSION, R.STATUS
        2  FROM SYS.CDB_REGISTRY R INNER JOIN V$PDBS P ON R.CON_ID =
      P.CON_ID
        3  WHERE R.COMP_ID = 'APEX' ORDER BY 1;
      ```

      If the APEX version is lower than 5.1.0.00.45, you must first upgrade to APEX 5.1. See Upgrading from Oracle Application Express 5.0 to 5.1 for Oracle Database 12*c* for details.

   f. Log out of SQL*Plus.
   g. Log out of the compute node.
2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the `opc` user.
   b. Check the version of the cloud tooling:

      ```
      $ rpm -q dbaastools
      ```

You should see something similar to this: `dbaastools-1.0-1+18.2.1.0.0_xxxxxx.xxxx.x86_64`. Check the value between the + and _ for the version number. The tooling version must be `18.2.1.0.0` or higher. If your cloud tooling is at a lower version, see Updating the Cloud Tooling on Database Cloud Service for the steps to update to a later version.

3. Upload Oracle Application Express 5.1.4.00.08 to the compute node:
   a. Download Oracle Application Express 5.1.4.00.08 from Oracle Technology Network.
   b. Download Patch 26795231: PATCH SET FOR APPLICATION EXPRESS (PATCH SET VERSION 5.1.4).
   c. Log in to the compute node as the `oracle` user.
   d. Upload the Oracle Application Express 5.1.4 zip file to the `/tmp` directory on the compute node. See Copying Files to or from a Database Cloud Service Database Deployment for instructions on how to copy files to the compute node.
   e. Upload the patch zip file to the `/tmp` directory on the compute node. See Copying Files to or from a Database Cloud Service Database Deployment for instructions on how to copy files to the compute node.
4. Unzip the patch zip file and the Oracle Application Express 5.1.4 zip file:
   a. Unzip the patch zip file.

      ```
      $ unzip p26795231_514_Generic.zip -d ./
      ```

   b. Create the directory required for Oracle Application Express:

      ```
      $ mkdir -p /u01/app/oracle/product/apex/
      ```

   c. Change to the `/tmp` directory where you uploaded the Oracle Application Express zip file.
   d. Unzip the uploaded Oracle Application Express zip file into the `apex` directory:

      ```
      $ unzip apex_5.1.4.zip -d /u01/app/oracle/product/apex/
      ```

5. Install Oracle Application Express 5.1.4:
   a. Move the Oracle Application Express files to the `/u01/app/oracle/product/apex/5.1.4.00.08/` directory:

      ```
      $ mv /u01/app/oracle/product/apex/apex/ /u01/app/oracle/product/
      apex/5.1.4.00.08/
      ```

   b. Optionally, delete the `zip` files:

      ```
      $ rm -f p26795231_514_Generic.zip
      $ rm -f apex_5.1.4.zip
      ```

   c. Change to the `patch` directory:

      ```
      $ cd ./patch
      ```

   d. Log in to SQL*Plus as the `SYS` user to install the patch in `PDB$SEED`:

      ```
      $ sqlplus / as SYSDBA
      SQL> ALTER SESSION SET "_oracle_script"=true;
      SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;
      ```

```
SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ WRITE;
SQL> ALTER SESSION SET CONTAINER = pdb$seed;
SQL> @apxpatch.sql
SQL> ALTER SESSION SET CONTAINER = cdb$root;
SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;
SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
```

e. Install the patch in each PDB you identified in step 1.e:

```
SQL> ALTER SESSION SET CONTAINER = PDB_NAME;

SQL> @apxpatch.sql
```

6. Verify that the version of APEX installed in each PDB is 5.1.4.00.08.

   a. If the Oracle Database version is 12.1.0.2, execute the following command to include `PDB$SEED` in queries:

   ```
   SQL> ALTER SESSION SET "EXCLUDE_SEED_CDB_VIEW"=FALSE;
   ```

   b. If the Oracle Database version is 12.2.0.1 or 18.1.0.0.0, execute the following command to include `PDB$SEED` in queries:

   ```
   SQL> ALTER SESSION SET "_EXCLUDE_SEED_CDB_VIEW"=FALSE;
   ```

   c. Execute the following query:

   ```
   SQL> SELECT R.CON_ID, P.NAME, R.VERSION, R.STATUS
     2  FROM SYS.CDB_REGISTRY R INNER JOIN V$PDBS P ON R.CON_ID =
   P.CON_ID
     3  WHERE R.COMP_ID = 'APEX' ORDER BY 1;
   ```

   d. Log out of SQL*Plus.

7. Optionally, delete the patch files:

   ```
   $ cd ..

   $ rm -rf ./patch
   ```

8. Configure Oracle Application Express by executing the Oracle REST Data Services (ORDS) assistant:

   a. Log in to the compute node as the `opc` user.

   b. Start a root-user shell:

   ```
   $ sudo -s
   ```

   c. Change to the `ords` directory:

   ```
   # cd /var/opt/oracle/ocde/assistants/ords
   ```

   d. Execute the ORDS assistant:

   ```
   # ./ords -out="/var/opt/oracle/ocde/res/ords.out" -
   ords_action="configure_apex"
   ```

   e. Restart ORDS:

   ```
   # /etc/init.d/ords restart
   ```

   f. Exit the root-user shell and log out of the compute node.

Upgrading from Oracle Application Express 5.1.0 or 5.1.3 or 5.1.4 to 18.1.0 for Oracle Database 12.2 and Oracle Database 18*c*

These instructions are applicable if you have an Oracle Database 12.2 or Oracle Database 18*c* database deployment and want to upgrade Oracle Application Express 5.1.0.00.45 or Oracle Application Express 5.1.3.00.05 or Oracle Application Express 5.1.4.00.08 to Oracle Application Express 18.1.0.00.45.

To upgrade from Oracle Application Express 5.1.0.00.45 or Oracle Application Express 5.1.3.00.05 or Oracle Application Express 5.1.4.00.08 to Oracle Application Express 18.1.0.00.45:

1. Determine the version of your current Oracle Application Express installation:
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user.

      ```
      $ sqlplus / as sysdba
      ```

   c. If the Oracle Database version is 12.2.0.1 or 18.1.0.0.0, execute the following command to include `PDB$SEED` in queries:

      ```
      SQL> ALTER SESSION SET "_EXCLUDE_SEED_CDB_VIEW"=FALSE;
      ```

   d. Execute the following query:

      ```
      SQL> SELECT CON_ID, VERSION, STATUS,
        2  SCHEMA FROM CDB_REGISTRY
        3  WHERE COMP_ID = 'APEX' ORDER BY CON_ID ASC;
      ```

      Sample output:

      ```
      CON_ID VERSION          STATUS      SCHEMA
           ------------------------------------------------------
           2 5.1.4.00.08           VALID       APEX_050100
           3 5.1.4.00.08           VALID       APEX_050100
      ```

      If the APEX version is lower than 5.1.0.00.45, you must first upgrade to APEX 5.1. See Upgrading from Oracle Application Express 5.0 to 5.1 for Oracle Database 12*c* for details.

   e. Log out of SQL*Plus.
   f. Log out of the compute node.
2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the `opc` user.
   b. Check the version of the cloud tooling:

      ```
      $ rpm -q dbaastools
      ```

      You should see something similar to this: `dbaastools-1.0-1+18.2.1.0.0_xxxxxx.xxxx.x86_64`. Check the value between the + and _ for the version number. The tooling version must be `18.2.1.0.0` or higher. If your cloud tooling is at a lower version, see Updating

the Cloud Tooling on Database Cloud Service for the steps to update to a later version.

3. Upload Oracle Application Express 18.1.0.00.45 to the compute node:
   a. Download Oracle Application Express 18.1.0.00.45 from Oracle Technology Network.
   b. Download the APEX 18.1 Archive.
   c. Log in to the compute node as the `oracle` user.
   d. Upload the Oracle Application Express 18.1.0.00.45 zip file to the `/tmp` directory on the compute node. See Copying Files to or from a Database Cloud Service Database Deployment for instructions on how to copy files to the compute node.
   e. Upload the patch zip file to the `/tmp` directory on the compute node. See Copying Files to or from a Database Cloud Service Database Deployment for instructions on how to copy files to the compute node.
4. Unzip the Oracle Application Express 18.1.0.00.45 zip file:
   a. Create the directory required for Oracle Application Express:

      ```
      $ mkdir -p /u01/app/oracle/product/apex/
      ```

   b. Change to the `/tmp` directory where you uploaded the Oracle Application Express zip file.
   c. Unzip the uploaded Oracle Application Express zip file into the `apex` directory:

      ```
      $ unzip apex_18.1.zip -d /u01/app/oracle/product/apex/
      ```

   d. Move the files into the Oracle Application Express 18.1.0.00.45:

      ```
      $ mv /u01/app/oracle/product/apex/apex/ /u01/app/oracle/product/
      apex/18.1.0.00.45/
      ```

   e. If the Oracle Application Express you are installing is the latest version you have on your system, run the following command as the `oracle` user:

      ```
      $ ln -sfn /u01/app/oracle/product/apex/18.1.0.00.45 /u01/app/
      oracle/product/apex/.latest_provisioned
      ```

   f. Optionally, delete the zip file:

      ```
      $ rm -f apex_18.1.zip
      ```

   g. Change to the directory where you moved the zip files:

      ```
      $ cd /u01/app/oracle/product/apex/18.1.0.00.45
      ```

5. Use SQL*Plus to make sure you can log into the `ORDS_PUBLIC_USER` schema.

   In the sample session below, use the password for the `ORDS_PUBLIC_USER` schema. By default, this password should be the same as the password you entered for the `SYS` user when you deployed your DBCS instance. Make sure you can login with this password by executing the following as the `oracle` user.

   ```
   $ sqlplus /nolog
   SQL> CONNECT ORDS_PUBLIC_USER
   Password:
   ```

```
Connected.
SQL>
```

In the following commands, <ORDS_PUBLIC_USER_PASSWORD> refers to the password you used here.

6. Install Oracle Application Express: :
   a. Execute the following command with SQL*Plus

      ```
      $ ALTER SESSION SET CONTAINER=<CONTAINER_NAME>;
      $@dbcsins.sql SYSAUX SYSAUX TEMP /i/18.1.0.00.45/
      <ORDS_PUBLIC_USER_PASSWORD>
      ```

   b. Verify that the installation went fine with the following command, using the same SQL*Plus session or running the command in the same container you installed Oracle Application Express.

      ```
      $ SELECT VERSION, STATUS, SCHEMA FROM DBA_REGISTRY WHERE COMP_ID =
      'APEX';
      ```

7. Create a link to the APEX images in ORDS docroot by executing the below command as the `oracle` user:

   ```
   ALTER SESSION SET CONTAINER=CDB$ROOT;

   ALTER SESSION SET "_oracle_script"=true;

   ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

   ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
   ```

Moving Oracle Application Express 5.1 from CDB$ROOT to PDBs

These instructions are applicable if you have an Oracle Database 12*c* Release 1 or Release 2 database deployment and you have previously upgraded to Oracle Application Express 5.1 in the root container (CDB$ROOT). These instructions tell you how to move Oracle Application Express to the pluggable databases (PDBs).

To move Oracle Application Express 5.1 from the root container (CDB$ROOT) to the pluggable databases (PDBs):

1. Determine the version of Oracle Application Express installed in the root container (CDB$ROOT):
   a. Log in to the compute node as the `oracle` user.
   b. Log in to SQL*Plus as the `SYS` user and query `DBA_REGISTRY`:

      ```
      $ sqlplus / as sysdba

      SQL> SELECT VERSION FROM DBA_REGISTRY WHERE COMP_ID = 'APEX';
      ```

      The version should be 5.1.0.00.45. If the version is 5.0.0.00.31 or 5.0.4.00.12, see Upgrading from Oracle Application Express 5.0 to 5.1 for Oracle Database 12*c*. If the version is 4.2.5.00.08, see Upgrading from Oracle Application Express 4.2 to 5.1 for Oracle Database 12*c*.

   c. Log out of SQL*Plus.
   d. Log out of the compute node.

2. Ensure that you have the required version of the cloud tooling:
   a. Log in to the compute node as the opc user.
   b. Check the version of the cloud tooling:

      ```
      rpm -q dbaastools
      ```

      You should see something similar to this: dbaastools-1.0-1+17.3.1.0.0_170605.2102.x86_64. Check the value between the + and _ for the version number. The tooling version must be 17.2.5.0.0 or higher. If your cloud tooling is at a lower version, see Updating the Cloud Tooling on Database Cloud Service for the steps to update to a later version.
3. Ensure the APEX 5.1.0.00.45 files are located in either the $ORACLE_HOME/apex directory or in the /u01/app/oracle/product/apex/5.1.0.00.45 directory.

   ```
   $ cat $ORACLE_HOME/apex/images/apex_version.txt
   ```

   ```
   $ cat /u01/app/oracle/product/apex/5.1.0.00.45/images/apex_version.txt
   ```

   The output from one of the commands should be Application Express Version: 5.1.
4. Determine which PDBs have an APEX instance:
   a. Log in to SQL*Plus as the SYS user.

      ```
      $ sqlplus / as sysdba
      ```
   b. Execute the query to determine which PDBs have an APEX instance:

      ```
      SQL> SELECT P.PDB_NAME, R.VERSION, R.STATUS
      2 FROM SYS.DBA_PDBS P, SYS.CDB_REGISTRY R
      3 WHERE P.PDB_ID = R.CON_ID AND R.COMP_ID = 'APEX'
      4 ORDER BY 1;
      ```
5. In each PDB, identify the APEX workspaces. Be sure to make note of which workspaces belong to which PDBs. You will need this information in a later step.
   a. While still logged in to SQL*Plus, connect to each PDB by using the ALTER SESSION command:

      ```
      SQL> ALTER SESSION SET CONTAINER = PDB_name;
      ```
   b. Query APEX_WORKSPACES to determine the APEX workspace names:

      ```
      SQL> SELECT WORKSPACE_ID FROM APEX_WORKSPACES WHERE WORKSPACE_ID >
      100;
      ```
   c. After connecting to each PDB and compiling a list of workspace IDs, log out of SQL*Plus.
6. Download a script that will be used to export the artifacts of the workspaces identified in the previous step into zip files:
   a. Create a directory for the workspace zip files:

      ```
      $ mkdir -p /home/oracle/workspaces
      ```
   b. Using wget, download the workspace_export_5.1.sh file from Oracle Storage Cloud Service.

      ```
      wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-
      usoracle29538/dbaas_patch/apex_upg5_1/workspace_export_5.1.sh or
      ```

```
wget https://a88717.storage.oraclecloud.com/v1/Storage-a88717/
dbaas_patch/apex_upg5_1/workspace_export_5.1.sh
```

c. Move the script to the `/home/oracle/workspaces` directory.

d. Change to the `workspaces` directory:

```
$ cd /home/oracle/workspaces
```

e. Assign permissions to the `workspace_export_5.1.sh` file:

```
$ chmod 755 /home/oracle/workspaces/workspace_export_5.1.sh
```

```
$ chown oracle:oinstall /home/oracle/workspaces/
workspace_export_5.1.sh
```

7. Temporarily unlock the `APEX_050100` user and temporarily set the user's password to `oracle` or a password of your choice. The script that you execute in a later step will reset the password.

a. Log in to SQL*Plus as the `SYS` user:

```
$ sqlplus / as sysdba
```

b. Unlock the user and set the password:

```
SQL> ALTER USER APEX_050100 ACCOUNT UNLOCK IDENTIFIED BY password
CONTAINER = ALL;
```

c. Log out of SQL*Plus.

8. Use Listener Control to determine the service name for each PDB

```
$ lsnrctl status
```

You should see output similar to:

```
Service "pdb1.opcdbaas.oraclecloud.internal" has 1 instance(s).
```

```
Instance "orcl", status READY, has 1 handler(s) for this service...
```

9. For each workspace you identified in step 5, execute the `workspace_export_5.1.sh` script.

```
./workspace_export_5.1.sh //localhost:port_number/PDB_service_name
APEX_050000 password workspace_id
```

where:

- `port_number` is the listener port number you specified when you created the database deployment (default is 1521).
- `PDB_service_name` is one of the service names you identified in step 8.
- `password` is the password you set for the `APEX_050100` user in step 7b.
- `workspace_id` is one of the APEX workspaces for the PDB that you identified in step 5b.

One zip file will be created for each workspace.

10. Change to the directory where the APEX 5.1.0.00.45 files are located, as you determined in step 3.

- If the files are located in `$ORACLE_HOME/apex/`: `cd $ORACLE_HOME/apex/`
- If the files are located in `/u01/app/oracle/product/apex/5.1.0.00.45/`: `cd /u01/app/oracle/product/apex/5.1.0.00.45/`

11. Delete the APEX users for the previous version of APEX installed in the root container (CDB$ROOT).
    a. Log in to SQL*Plus as the SYS user:

       ```
       $ sqlplus / as sysdba
       ```

    b. Query DBA_USERS.

       ```
       SQL> select username from dba_users where username like '%APEX_0%';
       ```

    c. Exit from SQL*Plus.
    d. Drop all of the users identified in step 11b except for the APEX_050100 user.

       ```
       $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
       drop_apex_user -- --x'drop user apex_user cascade'
       ```

       where apex_user is the value in the USERNAME column from step 11b. Do not drop the APEX_050100 user.
12. Remove APEX from the root container (CDB$ROOT).
    a. Execute the script to uninstall APEX.

       ```
       sqlplus / as SYSDBA @apxremov.sql
       ```

13. Stop ORDS.

    ```
    /etc/init.d/ords stop
    ```

14. Remove the APEX users from all containers.

    ```
    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_listener -- --x'drop user apex_listener cascade'

    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_rest_public_user -- --x'drop user apex_rest_public_user
    cascade'

    $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -b
    drop_apex_public_user -- --x'drop user apex_public_user cascade'
    ```

15. Install APEX in the CDB seed (PDB$SEED).
    a. Log in to SQL*Plus as the SYS user.

       ```
       $ sqlplus / as sysdba
       ```

    b. Set the mode of the CDB seed (PDB$SEED) to OPEN READ WRITE.

       ```
       SQL> ALTER SESSION SET "_oracle_script"=true;

       SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

       SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ WRITE;
       ```

    c. Connect to PDB$SEED.

       ```
       SQL> ALTER SESSION SET CONTAINER = PDB$SEED;
       ```

    d. Install APEX 5.1 in the CDB seed (PDB$SEED) container

       ```
       SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/

       SQL> exit
       ```

    e. Again, log in to SQL*Plus as the SYS user.

       ```
       $ sqlplus / as sysdba
       ```

f. Set the mode of the CDB seed (PDB$SEED) back to OPEN READ ONLY.

```
SQL> ALTER SESSION SET "_oracle_script"=true;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED CLOSE IMMEDIATE;

SQL> ALTER PLUGGABLE DATABASE PDB$SEED OPEN READ ONLY;
```

g. Exit from SQL*Plus.

16. Install APEX in each PDB you identified in step 4c.

a. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

b. Connect to the PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

c. Install APEX 5.1 in the PDB.

```
SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/
```

d. After repeating steps 16b and 16c for each PDB you identified in step 4b, exit from SQL*Plus.

e. Log out of the compute node.

17. Configure Oracle Application Express by executing the Oracle REST Data Services (ORDS) assistant:

a. Log in to the compute node as the `opc` user.

b. Start a root-user shell:

```
$ sudo -s
```

c. Change to the `ords` directory:

```
cd /var/opt/oracle/ocde/assistants/ords
```

d. Execute the ORDS assistant:

```
./ords -out="/var/opt/oracle/ocde/res/ords.out" -
ords_action="install"
```

e. Exit the root-user shell and log out of the compute node.

18. Install each APEX workspace into a PDB.

a. Log in to the compute node as the `oracle` user.

b. Change to the `workspaces` directory where the zip files from step 9 are located.

```
cd /home/oracle/workspaces
```

c. Unzip each of the workspace zip files into its corresponding directory.

```
unzip install_workspace_id_date.zip -d workspace_id
```

d. Change to one of the newly created `/home/oracle/workspaces/`*`workspace_id`* directories.

e. Log in to SQL*Plus as the `SYS` user.

```
$ sqlplus / as sysdba
```

f. Connect to the PDB where the workspace should be installed. Refer to the list you compiled in step 5 to ensure you install each workspace in the correct PDB.

```
SQL> ALTER SESSION SET CONTAINER = PDB_name;
```

g. Execute the installation script to install the APEX workspace in the PDB.

```
@install_workspace_id.sql
```

h. After executing steps 19d through 19g for each APEX workspace, exit SQL*Plus.

i. Log out of the compute node.

Using Oracle SQL Developer Web in Database Cloud Service

When you create an Oracle Database Cloud Service database deployment of a single-instance database, Oracle SQL Developer Web is installed for you.

SQL Developer Web is a web interface for Oracle SQL Developer and provides a subset of the features of the desktop version. It enables you to run SQL statements and scripts in the worksheet, export data, and design Data Modeler diagrams using new and existing objects. It also enables database administrators to monitor and manage the database and provides a real time SQL monitoring interface.

Access to SQL Developer Web is provided through schema-based authentication. Consequently, you can provide database developers and administrators access to databases without having to create and maintain Oracle Cloud accounts for them. However, before a developer or administrator can sign in to SQL Developer Web, you must enable their schema in the database for access.

Topics

Enabling a Schema for SQL Developer Web

When using Oracle SQL Developer Web in an Oracle Database Cloud Service database deployment, you sign in as a database user. Before you can do so, however, you must enable the database user's schema for SQL Developer Web.

> **Note:**
>
> You do not need to enable the PDBADMIN user's schema in the PDB created when a database deployment of Oracle Database 12.1 or later is created. Database Cloud Service automatically enables this schema for SQL Developer Web, including its DBA features.

Before you begin

Before attempting to enable a database user's schema for SQL Developer Web, you need to make sure the database deployment is using cloud tooling version 18.2.3 or later. For instructions on checking and updating the cloud tooling, see Updating the Cloud Tooling by Using the dbaascli Utility.

Procedure

1. Connect as the **opc** user to the deployment's compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Create a text file containing the password of the user whose schema you want to enable.

   a. Create the file:

      ```
      # touch /home/oracle/password.txt
      ```

   b. Restrict permissions on the file:

      ```
      # chmod 600 /home/oracle/password.txt
      ```

   c. Use a text editor (such as `vim`) to enter the password in the file. The file should consist of a single line containing the password without any whitespace characters.

4. Use the `ords` assistant to enable the schema.

   a. Go to the directory containing the `ords` assistant:

      ```
      # cd /var/opt/oracle/ocde/assistants/ords
      ```

   b. Run the `ords` assistant to enable the schema.

      For a schema in Oracle Database 11g, enter a command of this form:

      ```
      # ./ords -ords_action="enable_schema_for_sdw" \
      -ords_sdw_schema="schema-name" \
      -ords_sdw_schema_password="/home/oracle/password.txt" \
      -ords_sdw_schema_enable_dba="dba-boolean"
      ```

      For a schema in a PDB (pluggable database) in Oracle Database 12.1 or later, enter a command of this form:

      ```
      # ./ords -ords_action="enable_schema_for_sdw" \
      -ords_sdw_schema="schema-name" \
      -ords_sdw_schema_password="/home/oracle/password.txt" \
      -ords_sdw_schema_container="pdb-name" \
      -ords_sdw_schema_enable_dba="dba-boolean"
      ```

      In these command forms:

      - *schema-name* is the name of the schema you want to enable. If it doesn't exist, it will be created.

- *dba-boolean* is TRUE or FALSE. If you enter TRUE, the schema will be enabled to support the DBA (database administrator) features of SQL Developer Web.
- *pdb-name* is the name of the pluggable database (PDB) containing the schema you want to enable.

5. After the `ords` assistant finishes, take note of the sign-in information it provides. For example:

```
# ./ords -ords_action="enable_schema_for_sdw" \
-ords_sdw_schema="SDW" \
-ords_sdw_schema_password="/home/oracle/password.txt" \
-ords_sdw_schema_container="PDB1" \
-ords_sdw_schema_enable_dba="TRUE"

INFO:  To access SQL Developer Web through DBCS Landing Page, the schema "PDB1/
sdw" needs to be provided.
INFO: "SDW" schema in the "PDB1" container for SQL Developer Web was enabled
successfully.
#
```

6. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

Accessing SQL Developer Web

You can access Oracle SQL Developer Web in an Oracle Database Cloud Service database deployment in the following ways:

- Using the Database Deployment's Landing Page
- Using a Direct URL
- Using an SSH Tunnel

> **Tip:**
>
> The ways to access SQL Developer Web require you to provide the schema path reported by the `ords` assistant when the database user's schema was enabled for SQL Developer Web access. If you've forgotten this value, use these guidelines to determine it:
>
> - For a schema in Oracle Database 11g:
>
>   The schema path is the schema name with all letters lowercase and special characters changed to underscores. Multiple special characters in a row are changed to a single underscore. For example, the schema path for the C##CORPDBA1 schema is `c_corpdba1`.
>
> - For a schema in a PDB of Oracle Database 12c or later:
>
>   The schema path is the pdb name, a slash (/), and the schema name with all letters lowercase and special characters changed to underscores. Multiple special characters in a row are changed to a single underscore. For example, the schema path for the ORGDBA1 schema in the HRORG PDB is `hrorg/orgdba1`.

Using the Database Deployment's Landing Page

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access the deployment's landing page is blocked by default. To use the deployment's landing page, you must unblock port 443, either by enabling the deployment's **ora_p2_httpssl** predefined access rule or by creating your own access rule that opens port 443. For instructions, see Enabling Access to a Compute Node Port.

1. In your web browser, go to the following URL:

   `https://node-ip-address/`

   where *node-ip-address* is the IP address of the deployment's compute node as listed on the deployment's Overview page.
2. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.
3. In the SQL Developer box's **Schema** field, enter the schema path reported by the `ords` assistant when the database user's schema was enabled for SQL Developer Web access. Then click **Go**.
4. When prompted for a username and password, enter the user name and password of the database user whose schema path you entered in the previous step. Make sure to enter the user name in all-uppercase. Then click **Sign In**.
5. If the user's schema was enabled to support the DBA features of SQL Developer Web, the Database Cloud Service Dashboard page is displayed. Otherwise, the SQL Developer Home page is displayed.

Using a Direct URL

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access SQL Developer Web is blocked by default. To use a direct URL, you must unblock port 443, either by enabling the deployment's **ora_p2_httpssl** predefined access rule or by creating your own access rule that opens port 443. For instructions, see Enabling Access to a Compute Node Port.

1. In your web browser, go to the following URL:

   `https://node-ip-address/ords/schema-path/_sdw`

where *node-ip-address* is the IP address of the deployment's compute node as listed on the deployment's Overview page, and *schema-path* is the schema path reported by the `ords` assistant when the schema was enabled for SQL Developer Web access.

2. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.

3. When prompted for a username and password, enter the user name and password of the database user whose *schema-path* you gave in the URL. Make sure to enter the user name in all-uppercase. Then click **Sign In**.

4. If the user's schema was enabled to support the DBA features of SQL Developer Web, the Database Cloud Service Dashboard page is displayed. Otherwise, the SQL Developer Home page is displayed.

Using an SSH Tunnel

1. Create an SSH tunnel to port 443 on the compute node hosting SQL Developer Web. For information about creating an SSH tunnel, see Creating an SSH Tunnel to a Compute Node Port.

2. After creating the SSH tunnel, direct your browser to the following URL:

   ```
   https://localhost/ords/schema-path/_sdw
   ```

   where *schema-path* is the schema path reported by the `ords` assistant when the schema was enabled for SQL Developer Web access.

3. When prompted for a username and password, enter the user name and password of the database user whose *schema-path* you gave in the URL. Make sure to enter the user name in all-uppercase. Then click **Sign In**.

4. If the user's schema was enabled to support the DBA features of SQL Developer Web, the Database Cloud Service Dashboard page is displayed. Otherwise, the SQL Developer Home page is displayed.

Features of SQL Developer Web

Oracle SQL Developer Web provides a rich set of developer and administrator features for a database, a CDB (container database) or a PDB (pluggable database).

Developer Features

Users of the Oracle SQL Developer desktop product will find SQL Developer Web very familiar and easy-to-use. It provides similar Worksheet and Data Modeler features, and includes a Home page that provides pertinent database statistics and access to the worksheets and diagrams you have saved. For more information, see these topics in *Using Oracle SQL Developer Web*:

- Using the Worksheet
- Using Data Modeler
- About Home

Administrator Features

For users whose schemas have been enabled for DBA access, SQL Developer Web provides features for database and OS administration and monitoring:

- A dashboard that shows overview statistics for the database. This dashboard is displayed immediately after you sign in, and many of the statistics it displays are “hot”: you can click them to go directly to detail pages where you can investigate and take appropriate action. Additionally, the dashboard provides quick links to such resources as the Worksheet, Data Modeler and SQL Monitor.
- DBA features to manage the database and see information about the listener, backups, alerts and several other items of interest to DBAs. For more information, see Using DBA Features in *Using Oracle SQL Developer Web*.
- OS features to monitor RAM and CPU usage, OS storage and running processes. For more information, see Monitoring OS in *Using Oracle SQL Developer Web*.

Using the Demos PDB

The Demos PDB (pluggable database) contains demos that highlight some of the more popular features of Oracle Database 12c. You can have this PDB included when you create a Database Cloud Service database deployment hosting an Oracle Database 12c single-instance database or Oracle Data Guard configuration.

Some of these popular features include JSON in the database, XMLDB, APEX development, the In-Memory option and Data Mining. There is also a web page installed for you that provides access to labs and tutorials to be used with this PDB.

To install the Demos PDB, simply check the **Include "Demos" PDB option** on the Details page of the Create Instance wizard when creating a database deployment.

> **Note:**
>
> DEMOS PDB is only available for Oracle Database 12*c* Release 1 database deployments.

Accessing the Labs and Tutorials

You can access the web page to the Demos PDB’s labs and tutorials by directing your browser to the URL `https://node-ip-address/jet/home.html`. In this URL *`node-ip-address`* is the public IP address of the deployment’s compute node; it is listed on the deployment’s Oracle Database Cloud Service Overview page. To display this page, see Viewing Detailed Information for a Database Deployment.

If access to web pages is blocked on the deployment, you need to enable access to HTTPS port 443 before you can view this page. For instructions, see Enabling Access to a Compute Node Port.

Adding the Demos PDB to an Existing Database Deployment

> **Note:**
>
> DEMOS PDB is only available for Oracle Database 12*c* Release 1 database deployments.

If you didn't install the Demos PDB during the creation of your database deployment, you can add it later by following these steps:

1. Connect to the deployment's compute node as the **`oracle`** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Download the script to create and enable the Demos PDB:

   ```
   $ wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/pdb_demo/demo.pl
   ```

3. Set the file permissions on the script to make it executable:

   ```
   $ chmod a+x demo.pl
   ```

4. Execute the patching script:

   ```
   $ ./demo.pl
   ```

5. Disconnect from the compute node:

   ```
   $ exit
   ```

Using Oracle Enterprise Manager Cloud Control with Database Cloud Service

Oracle Enterprise Manager Cloud Control 12c Release 5 (12.1.0.5) includes Hybrid Cloud Management, which you can use to manage the Oracle Databases on Oracle Database Cloud Service from the same management console that you use for your on-premises databases.

Oracle Enterprise Manager is Oracle's integrated enterprise IT management product line, which provides the industry's only complete, integrated and business-driven enterprise cloud management solution. Oracle Enterprise Manager 12c delivers comprehensive capabilities to manage, migrate, test, and deploy applications and workloads across hybrid clouds. See EM Hybrid Cloud Management for information about its hybrid cloud features.

To use the Hybrid Cloud Management feature to manage the Oracle Databases on Database Cloud Service, you perform these high-level tasks:

1. Ensure that your on-premises Enterprise Manager Cloud Control Oracle Management Service (OMS) is of version 12.1.0.5, and that at least one 12.1.0.5 Management Agent exists in your enterprise.
2. Configure one or more version 12.1.0.5 Management Agents within your enterprise to act as Hybrid Cloud Gateway Agents, which provide an SSH-based

communication channel between Database Cloud Service compute nodes and the on-premises OMS.

3. Ensure that the Hybrid Cloud Gateway Agents and the on-premises OMS can communicate with the Database Cloud Service compute nodes.
4. Deploy Management Agents to Database Cloud Service compute nodes using the Add Host Targets Wizard or Enterprise Manager Command Line Interface.

For the detailed steps to perform these tasks, see Enabling Hybrid Cloud Management in *Oracle Enterprise Manager Cloud Control Administrator's Guide*, 12*c* Release 5 (12.1.0.5).

Preserving the Hybrid Cloud Agent Home When Patching a Database Cloud Service Deployment

If you installed the Oracle Enterprise Manager Cloud Control agent under `/u01/app/oracle` (the Oracle Base directory) the agent home will be moved to `/u01/app.ORG/oracle` when a database patch is applied to the database deployment using the cloud tooling.

If the agent home has already been moved to `/u01/app.ORG/oracle`, you can copy it back to `/u01/app/oracle` to restore it.

Perform the following steps to configure the patching tools so that they do not move the agent home to `/u01/app.ORG/oracle`:

1. Connect to the compute node as the **`oracle`** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).
2. Create a file named `/var/opt/oracle/patch/files_to_save.ora`.
3. Add the full path of the agent directory to the `/var/opt/oracle/patch/files_to_save.ora` file.
4. Disconnect from the compute node, and then reconnect as the **`opc`** user.
5. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```
6. Edit the patching configuration file, `/var/opt/oracle/patch/dbpatchm.cfg`, and search for the following lines:

   ```
   # create /var/opt/oracle/patch/files_to_save.ora with full path of directory or
   # files to preserve any special files you may have in your /u01/app directory.
   # set this to yes, if you have files_to_save.ora
   special_files="no"
   ```
7. Change `special_files="no"` to `special_files="yes"`.
8. Save and close the file.
9. Exit the root-user command shell and disconnect from the compute node.

After performing these steps, the agent home will be preserved in its original location whenever the database deployment is patched.

Using Oracle GoldenGate Cloud Service with Database Cloud Service

 This topic does not apply to Oracle Cloud Infrastructure.

Oracle GoldenGate Cloud Service is a secure, high performance data integration and replication service that can replicate data in real time from on-premises databases to single-instance databases in Oracle Database Cloud Service.

You must create a Database Cloud Service database deployment that is properly configured for use as a GoldenGate Cloud Service replication target before you create a GoldenGate Cloud Service instance.

To properly configure a Database Cloud Service database deployment for use as a replication target:

- You must configure the database deployment to use characteristics (like database release, database edition and so on) that are supported by Oracle GoldenGate Cloud Service.

  See Before You Begin with Oracle GoldenGate Cloud Service in *Using Oracle GoldenGate Cloud Service* for information about the Database Cloud Service characteristics that Oracle GoldenGate Cloud Service supports.

- You must configure the database deployment for use as a replication database.

  You can configure the database deployment for use as a replication database by setting the **Enable Oracle GoldenGate** option on the Instance Details page of the Create Instance wizard, or you can configure it manually after the database deployment is created by using the `dbaascli` utility. See Manually Configuring a Deployment's Database for Oracle GoldenGate Cloud Service Replication for instructions on configuring it manually.

- The target database must be network accessible on the listener port.

  To enable access on the listener port, you need to enable the **ora_p2_dblistener** security rule automatically created for the Database Cloud Service database deployment when the deployment was created. See Enabling Port Access by Enabling an Automatically Created Access Rulefor instructions.

Once you have created and properly configured a Database Cloud Service database deployment for use as a replication target, you can create an Oracle GoldenGate Cloud Service instance that uses it. See Provision an Oracle GoldenGate Cloud Service Instance in *Using Oracle GoldenGate Cloud Service*.

Manually Configuring a Deployment's Database for Oracle GoldenGate Cloud Service Replication

Before you can use an Oracle Database Cloud Service database deployment as a replication target in Oracle GoldenGate Cloud Service, you must configure its database as a valid replication database.

You can configure the database during database deployment creation by setting the **Enable Oracle GoldenGate** option on the Service Details page of the wizard. If you

do not, you can configure it manually after the database deployment is created by using the `dbaascli` utility.

To configure the database manually after the database deployment is created:

1. Connect as the **`oracle`** user to the database deployment's compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Confirm that the database is not yet configured as a valid replication database:

   ```
   $ dbaascli gg status
   DBAAS CLI version 1.0.0
   Executing command gg status

   Golden Gate status: disabled.
   ```

   If the status is listed as `disabled`, you need to configure the database; if it is listed as `enabled`, you do not.

3. Configure the database as a valid replication database by using the `dbaascli gg setup` command:

   ```
   $ dbaascli gg setup
   DBAAS CLI version 1.0.0
   Executing command gg setup
   Enter Golden Gate admin username: admin-username
   Enter Golden Gate admin password: admin-password
   Re-enter Golden Gate admin password: admin-password
   Setting up Golden Gate
   Updating the registry
   Successfully setup GG
   ```

   Where:

   - *`admin-username`* is the database user name for Oracle GoldenGate Cloud Service access to the database:
     - For Oracle Database 11g, specify **`ggadmin`**.
     - For Oracle Database 12c or later, specify **`c##ggadmin`**.
   - *`admin-password`* is the password to use for the database user. You can use the administrator password provided when the database deployment was created, or you can use a different password that conforms to password requirements for Oracle Database users.

4. Close your connection to the compute node.

9

Migrating Oracle Databases to Database Cloud Service

You can migrate your on-premises Oracle databases to Oracle Database Cloud Service using various different approaches based on different tools and technologies.

Topics

Choosing a Migration Method

You can migrate your on-premises Oracle Database database to an Oracle Database Cloud database using a number of different methods that use several different tools.

Not all migration methods apply to all migration scenarios. Many of the migration methods apply only if specific characteristics of the source and destination databases match or are compatible. Moreover, additional factors can affect which method you choose for your migration from among the methods that are technically applicable to your migration scenario.

Some of the characteristics and factors to consider when choosing a migration method are:

- On-premises database version
- Oracle Database Cloud database version
- On-premises host operating system and version
- On-premises database character set
- Quantity of data, including indexes
- Data types used in the on-premises database
- Storage for data staging
- Acceptable length of system outage
- Network bandwidth

To determine which migration methods are applicable to your migration scenario, gather the following information.

1. Database version of your on-premises database:
   - Oracle Database 11g Release 2 version lower than 11.2.0.3
   - Oracle Database 11g Release 2 version 11.2.0.3 or higher
   - Oracle Database 12c Release 1 version lower than 12.1.0.2
   - Oracle Database 12c Release 1 version 12.1.0.2 or higher
2. For on-premises Oracle Database 12c Release 1 databases, the architecture of the database:
   - Multitenant container database (CDB)
   - Non-CDB
3. Endian format (byte ordering) of your on-premises database's host platform

   Some platforms are little endian and others are big endian. Query `V$TRANSPORTABLE_PLATFORM` to identify the endian format, and to determine whether cross-platform tablespace transport is supported.

   Oracle Database Cloud uses the Linux platform, which is little endian.
4. Database character set of your on-premises database and Oracle Database Cloud Service database

   Some migration methods require that the source and target databases use compatible database character sets.
5. Database version of the Oracle Database Cloud database you are migrating to
   - Oracle Database 11g Release 2
   - Oracle Database 12c Release 1

   Oracle Database 12c Release 1 databases created on Oracle Database Cloud use CDB architecture. Databases created using the Enterprise Edition software edition are single-tenant, and databases created using the High Performance or Extreme Performance software editions are multitenant.

After gathering this information, use the "source" and "destination" database versions as your guide to see which migration methods apply to your migration scenario:

- Migrating from Oracle Database 11g to Oracle Database 11g in the Cloud
- Migrating from Oracle Database 11g to Oracle Database 12c in the Cloud
- Migrating from Oracle Database 12c CDB to Oracle Database 12c in the Cloud
- Migrating from Oracle Database 12c Non-CDB to Oracle Database 12c in the Cloud

Migrating from Oracle Database 11g to Oracle Database 11g in the Cloud

You can migrate Oracle Database 11g databases from on-premises to Oracle Database 11g databases in Oracle Database Cloud using several different methods.

The applicability of some of the migration methods depends on the on-premises database's database character set and platform endian format.

If you have not already done so, determine the database character set of your on-premises database, and determine the endian format of the platform your on-premises database resides on. Use this information to help you choose an appropriate method.

- Data Pump Conventional Export/Import

  This method can be used regardless of the endian format and database character set of the on-premises database.

  For the steps this method entails, see Data Pump Conventional Export/Import.

- Data Pump Transportable Tablespace

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Transportable Tablespace.

- RMAN Transportable Tablespace with Data Pump

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see RMAN Transportable Tablespace with Data Pump.

- RMAN `CONVERT` Transportable Tablespace with Data Pump

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  This method is similar to the Data Pump Transportable Tablespace method, with the addition of the RMAN `CONVERT` command to enable transport between platforms with different endianness. Query `V$TRANSPORTABLE_PLATFORM` to determine if the on-premises database platform supports cross-platform tablespace transport and to determine the endian format of the platform. The Oracle Database Cloud platform is little-endian format.

  For the steps this method entails, see RMAN CONVERT Transportable Tablespace with Data Pump.

Migrating from Oracle Database 11g to Oracle Database 12c in the Cloud

You can migrate Oracle Database 11g databases from on-premises to Oracle Database 12c databases in Oracle Database Cloud using several different methods.

The applicability of some of the migration methods depends on the on-premises database's database version, database character set and platform endian format.

If you have not already done so, determine the database version and database character set of your on-premises database, and determine the endian format of the platform your on-premises database resides on. Use this information to help you choose an appropriate method.

- Data Pump Conventional Export/Import

  This method can be used regardless of the endian format and database character set of the on-premises database.

For the steps this method entails, see Data Pump Conventional Export/Import.

- Data Pump Transportable Tablespace

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Transportable Tablespace.

- RMAN Transportable Tablespace with Data Pump

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see RMAN Transportable Tablespace with Data Pump.

- RMAN `CONVERT` Transportable Tablespace with Data Pump

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  This method is similar to the Data Pump Transportable Tablespace method, with the addition of the RMAN `CONVERT` command to enable transport between platforms with different endianness. Query `V$TRANSPORTABLE_PLATFORM` to determine if the on-premises database platform supports cross-platform tablespace transport and to determine the endian format of the platform. The Oracle Database Cloud platform is little-endian format.

  For the steps this method entails, see RMAN CONVERT Transportable Tablespace with Data Pump.

- Data Pump Full Transportable

  This method can be used only if the source database release version is 11.2.0.3 or later, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Full Transportable.

Migrating from Oracle Database 12c CDB to Oracle Database 12c in the Cloud

You can migrate Oracle Database 12c CDB databases from on-premises to Oracle Database 12c databases in Oracle Database Cloud using several different methods.

The applicability of some of the migration methods depends on the on-premises database's database character set and platform endian format.

If you have not already done so, determine the database character set of your on-premises database, and determine the endian format of the platform your on-premises database resides on. Use this information to help you choose an appropriate method.

- Data Pump Conventional Export/Import

  This method can be used regardless of the endian format and database character set of the on-premises database.

  For the steps this method entails, see Data Pump Conventional Export/Import.

- Data Pump Transportable Tablespace

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Transportable Tablespace.

- RMAN Transportable Tablespace with Data Pump

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see RMAN Transportable Tablespace with Data Pump.

- RMAN `CONVERT` Transportable Tablespace with Data Pump

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  This method is similar to the Data Pump Transportable Tablespace method, with the addition of the RMAN `CONVERT` command to enable transport between platforms with different endianness. Query `V$TRANSPORTABLE_PLATFORM` to determine if the on-premises database platform supports cross-platform tablespace transport and to determine the endian format of the platform. The Oracle Database Cloud platform is little-endian format.

  For the steps this method entails, see RMAN CONVERT Transportable Tablespace with Data Pump.

- RMAN Cross-Platform Transportable Tablespace Backup Sets

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see RMAN Cross-Platform Transportable Tablespace Backup Sets.

- Data Pump Full Transportable

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Full Transportable.

- Unplugging/Plugging (CDB)

  This method can be used only if the on-premises platform is little endian, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

  For the steps this method entails, see Unplugging/Plugging a PDB.

- Remote Cloning (CDB)

  This method can be used only if the on-premises platform is little endian, the on-premises database release is 12.1.0.2 or higher, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

  For the steps this method entails, see Remote Cloning a PDB.

- RMAN Cross-Platform Transportable PDB

This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

For the steps this method entails, see RMAN Cross-Platform Transportable PDB.

- SQL Developer and SQL*Loader to Migrate Selected Objects

  You can use SQL Developer to create a cart into which you add selected objects to be loaded into your Oracle Database 12c database on the cloud. In this method, you use SQL*Loader to load the data into your cloud database.

  For the steps this method entails, see SQL Developer and SQL*Loader to Migrate Selected Objects.

- SQL Developer and `INSERT` Statements to Migrate Selected Objects

  You can use SQL Developer to create a cart into which you add selected objects to be loaded into your Oracle Database 12c database on the cloud. In this method, you use SQL `INSERT` statements to load the data into your cloud database.

  For the steps this method entails, see SQL Developer and INSERT Statements to Migrate Selected Objects.

Migrating from Oracle Database 12c Non-CDB to Oracle Database 12c in the Cloud

You can migrate Oracle Database 12c non-CDB databases from on-premises to Oracle Database 12c databases in Oracle Database Cloud using several different methods.

The applicability of some of the migration methods depends on the on-premises database's database character set and platform endian format.

If you have not already done so, determine the database character set of your on-premises database, and determine the endian format of the platform your on-premises database resides on. Use this information to help you choose an appropriate method.

- Data Pump Conventional Export/Import

  This method can be used regardless of the endian format and database character set of the on-premises database.

  For the steps this method entails, see Data Pump Conventional Export/Import.

- Data Pump Transportable Tablespace

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Transportable Tablespace.

- RMAN Transportable Tablespace with Data Pump

  This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see RMAN Transportable Tablespace with Data Pump.

- RMAN `CONVERT` Transportable Tablespace with Data Pump

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  This method is similar to the Data Pump Transportable Tablespace method, with the addition of the RMAN `CONVERT` command to enable transport between platforms with different endianness. Query `V$TRANSPORTABLE_PLATFORM` to determine if the on-premises database platform supports cross-platform tablespace transport and to determine the endian format of the platform. The Oracle Database Cloud platform is little-endian format.

  For the steps this method entails, see RMAN CONVERT Transportable Tablespace with Data Pump.

- RMAN Cross-Platform Transportable Tablespace Backup Sets

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see RMAN Cross-Platform Transportable Tablespace Backup Sets.

- Data Pump Full Transportable

  This method can be used only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

  For the steps this method entails, see Data Pump Full Transportable.

- Unplugging/Plugging (non-CDB)

  This method can be used only if the on-premises platform is little endian, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

  You can use the unplug/plug method to migrate an Oracle Database 12c non-CDB database to Oracle Database 12c in the cloud. This method provides a way to consolidate several non-CDB databases into a single Oracle Database 12c CDB on the cloud.

  For the steps this method entails, see Unplugging/Plugging Non-CDB.

- Remote Cloning (non-CDB)

  This method can be used only if the on-premises platform is little endian, the on-premises database release is 12.1.0.2 or higher, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

  You can use the remote cloning method to copy an Oracle Database 12c non-CDB on-premises database to your Oracle Database 12c database in the cloud.

  For the steps this method entails, see Remote Cloning Non-CDB.

- SQL Developer and SQL*Loader to Migrate Selected Objects

  You can use SQL Developer to create a cart into which you add selected objects to be loaded into your Oracle Database 12c database on the cloud. In this method, you use SQL*Loader to load the data into your cloud database.

  For the steps this method entails, see SQL Developer and SQL*Loader to Migrate Selected Objects.

- SQL Developer and `INSERT` Statements to Migrate Selected Objects

You can use SQL Developer to create a cart into which you add selected objects to be loaded into your Oracle Database 12c database on the cloud. In this method, you use SQL `INSERT` statements to load the data into your cloud database.

For the steps this method entails, see SQL Developer and INSERT Statements to Migrate Selected Objects.

Migration Methods

Many methods exist to migrate Oracle databases to Oracle Database Cloud Service.

Which of these methods apply to a given migration scenario depends on several factors, including the version, character set, and platform endian format of the source and target databases.

Topics

Data Pump Conventional Export/Import

You can use this method regardless of the endian format and database character set of the on-premises database.

To migrate an on-premises source database, tablespace, schema, or table to the database on an Oracle Database Cloud Service database deployment using Data Pump Export and Import, you perform these tasks:

1. On the on-premises database host, invoke Data Pump Export and export the on-premises database.
2. Use a secure copy utility to transfer the dump file to the Database Cloud Service compute node.
3. On the Database Cloud Service compute node, invoke Data Pump Import and import the data into the database.
4. After verifying that the data has been imported successfully, you can delete the dump file.

For information about Data Pump Import and Export, see these topics:

- "Data Pump Export Modes" in *Oracle Database Utilities* for Release 12.2, 12.1 or 11.2.
- "Data Pump Import Modes" in *Oracle Database Utilities* for Release 12.2, 12.1 or 11.2.

Data Pump Conventional Export/Import: Example

This example provides a step-by-step demonstration of the tasks required to migrate a schema from an on-premises Oracle database to an Oracle Database Cloud Service database.

This example illustrates a schema mode export and import. The same general procedure applies for a full database, tablespace, or table export and import.

In this example, the on-premises database is on a Linux host.

1. On the on-premises database host, invoke Data Pump Export to export the schemas.
   a. On the on-premises database host, create an operating system directory to use for the on-premises database export files.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/dpdump/for_cloud
      ```

   b. On the on-premises database host, invoke SQL*Plus and log in to the on-premises database as the SYSTEM user.

      ```
      $ sqlplus system
      Enter password: <enter the password for the SYSTEM user>
      ```

   c. Create a directory object in the on-premises database to reference the operating system directory.

      ```
      SQL> CREATE DIRECTORY dp_for_cloud AS '/u01/app/oracle/admin/orcl/dpdump/
      for_cloud';
      ```

   d. Exit from SQL*Plus.
   e. On the on-premises database host, invoke Data Pump Export as the SYSTEM user or another user with the DATAPUMP_EXP_FULL_DATABASE role and export the on-premises schemas. Provide the password for the user when prompted.

      ```
      $ expdp system SCHEMAS=fsowner DIRECTORY=dp_for_cloud
      ```

2. Use a secure copy utility to transfer the dump file to the Database Cloud Service compute node.

   In this example the dump file is copied to the /u01 directory. Choose the appropriate location based on the size of the file that will be transferred.

   a. On the Database Cloud Service compute node, create a directory for the dump file.

      ```
      $ mkdir /u01/app/oracle/admin/ORCL/dpdump/from_onprem
      ```

   b. Before using the scp command to copy the export dump file, make sure the SSH private key that provides access to the Database Cloud Service compute node is available on your on-premises host. For more information about SSH keys, see About Network Access to Database Cloud Service.

c. On the on-premises database host, use the SCP utility to transfer the dump file to the Database Cloud Service compute node.

```
$ scp -i private_key_file \
/u01/app/oracle/admin/orcl/dpdump/for_cloud/expdat.dmp \
oracle@IP_address_DBaaS_VM:/u01/app/oracle/admin/ORCL/dpdump/from_onprem
```

3. On the Database Cloud Service compute node, invoke Data Pump Import and import the data into the database.

   a. On the Database Cloud Service compute node, invoke SQL*Plus and log in to the database as the `SYSTEM` user.

   ```
   $ sqlplus system
   Enter password: <enter the password for the SYSTEM user>
   ```

   b. Create a directory object in the Database Cloud Service database.

   ```
   SQL> CREATE DIRECTORY dp_from_onprem AS '/u01/app/oracle/admin/ORCL/dpdump/
   from_onprem';
   ```

   c. If they do not exist, create the tablespace(s) for the objects that will be imported.

   d. Exit from SQL*Plus.

   e. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database. Import the data into the database.

   ```
   $ impdp system SCHEMAS=fsowner DIRECTORY=dp_from_onprem
   ```

4. After verifying that the data has been imported successfully, you can delete the `expdat.dmp` file.

Data Pump Full Transportable

You can use this method only if the source database release version is 11.2.0.3 or later, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

You can use the Data Pump full transportable method to copy an entire database from your on-premises host to the database on an Oracle Database Cloud Service database deployment.

To migrate an Oracle Database 11g on-premises database to the Oracle Database 12c database on a Database Cloud Service database deployment using the Data Pump full transportable method, you perform these tasks:

1. On the on-premises database host, prepare the database for the Data Pump full transportable export by placing the user-defined tablespaces in `READ ONLY` mode.
2. On the on-premises database host, invoke Data Pump Export to perform the full transportable export.
3. Use a secure copy utility to transfer the Data Pump Export dump file and the datafiles for all of the user-defined tablespaces to the Database Cloud Service compute node.
4. Set the on-premises tablespaces back to `READ WRITE`.
5. On the Database Cloud Service compute node, prepare the database for the tablespace import.

6. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database.
7. After verifying that the data has been imported successfully, you can delete the dump file.

Data Pump Full Transportable: Example

This example provides a step-by-step demonstration of the tasks required to migrate an Oracle Database 11*g* database to an Oracle Database Cloud Service 12*c* database.

In this example, the source database is on a Linux host.

1. On the source database host, prepare the database for the Data Pump full transportable export.
   a. On the source database host, create a directory in the operating system to use for the source export.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/dpdump/for_cloud
      ```

   b. On the source database host, invoke SQL*Plus and log in to the source database as the `SYSTEM` user.

      ```
      $ sqlplus system
      Enter password: <enter the password for the SYSTEM user>
      ```

   c. Create a directory object in the source database to reference the operating system directory.

      ```
      SQL> CREATE DIRECTORY dp_for_cloud AS '/u01/app/oracle/admin/orcl/dpdump/
      for_cloud';
      ```

   d. Determine the name(s) of the tablespaces and data files that belong to the user-defined tablespaces by querying `DBA_DATA_FILES`. These files will also be listed in the export output.

      ```
      SQL> SELECT tablespace_name, file_name FROM dba_data_files;
      TABLESPACE_NAME  FILE_NAME
      ---------------  --------------------------------------------------
      USERS            /u01/app/oracle/oradata/orcl/users01.dbf
      UNDOTBS1         /u01/app/oracle/oradata/orcl/undotbs01.dbf
      SYSAUX           /u01/app/oracle/oradata/orcl/sysaux01.dbf
      SYSTEM           /u01/app/oracle/oradata/orcl/system01.dbf
      EXAMPLE          /u01/app/oracle/oradata/orcl/example01.dbf
      FSDATA           /u01/app/oracle/oradata/orcl/fsdata01.dbf
      FSINDEX          /u01/app/oracle/oradata/orcl/fsindex01.dbf
      SQL>
      ```

   e. On the source database host, set all tablespaces that will be transported (the transportable set) to `READ ONLY` mode.

      ```
      SQL> ALTER TABLESPACE example READ ONLY;
      Tablespace altered.
      SQL> ALTER TABLESPACE fsindex READ ONLY;
      Tablespace altered.
      SQL> ALTER TABLESPACE fsdata READ ONLY;
      Tablespace altered.
      SQL> ALTER TABLESPACE users READ ONLY;
      Tablespace altered.
      SQL>
      ```

f. Exit from SQL*Plus.

2. On the source database host, invoke Data Pump Export to perform the full transportable export. Specify `FULL=y` and `TRANSPORTABLE=always`. Because this is an Oracle Database 11g database and full transportable is an Oracle Database 12c feature, specify `VERSION=12`. Provide the password for the `SYSTEM` user when prompted.

```
$ expdp system FULL=y TRANSPORTABLE=always VERSION=12 DUMPFILE=expdat.dmp
DIRECTORY=dp_for_cloud
```

3. Use a secure copy utility to transfer the Data Pump Export dump file and the datafiles for all of the user-defined tablespaces to the Database Cloud Service compute node.

   In this example the dump file is copied to the `/u01` directory. Choose the appropriate location based on the size of the file that will be transferred.

   a. On the Database Cloud Service compute node, create a directory for the dump file.

```
$ mkdir /u01/app/oracle/admin/ORCL/dpdump/from_source
```

   b. Before using the `scp` utility to copy files, make sure the SSH private key that provides access to the Database Cloud Service compute node is available on your source host. For more information about SSH keys, see About Network Access to Database Cloud Service.

   c. On the source database host, use the `scp` utility to transfer the dump file and all datafiles of the transportable set to the Database Cloud Service compute node.

```
$ scp -i private_key_file \
/u01/app/oracle/admin/orcl/dpdump/for_cloud/expdat.dmp \
oracle@compute_node_IP_address:/u01/app/oracle/admin/ORCL/dpdump/from_source

$ scp -i private_key_file \
/u01/app/oracle/oradata/orcl/example01.dbf \
oracle@compute_node_IP_address:/u02/app/oracle/oradata/ORCL/PDB2

$ scp -i private_key_file \
/u01/app/oracle/oradata/orcl/fsdata01.dbf \
oracle@compute_node_IP_address:/u02/app/oracle/oradata/ORCL/PDB2

$ scp -i private_key_file \
/u01/app/oracle/oradata/orcl/fsindex01.dbf \
oracle@compute_node_IP_address:/u02/app/oracle/oradata/ORCL/PDB2

$ scp -i private_key_file \
/u01/app/oracle/oradata/orcl/users01.dbf \
oracle@compute_node_IP_address:/u02/app/oracle/oradata/ORCL/PDB2
```

4. Set the source tablespaces back to `READ WRITE`.

   a. Invoke SQL*Plus and log in as the `SYSTEM` user.

   b. Set the user-defined tablespaces back to `READ WRITE` mode.

```
SQL> ALTER TABLESPACE example READ WRITE;
Tablespace altered.
SQL> ALTER TABLESPACE fsdata READ WRITE;
Tablespace altered.
SQL> ALTER TABLESPACE fsindex READ WRITE;
Tablespace altered.
```

```
SQL> ALTER TABLESPACE users READ WRITE;
Tablespace altered.
```

c. Exit from SQL*Plus.

5. On the Database Cloud Service compute node, prepare the PDB for the tablespace import.

   a. On the Database Cloud Service compute node, invoke SQL*Plus and log in to the PDB as the `SYSTEM` user.

   b. Create a directory object in the PDB.

   ```
   SQL> CREATE DIRECTORY dp_from_source AS '/u01/app/oracle/admin/ORCL/dpdump/
   from_source';
   ```

6. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the PDB.

   Import the data into the database using the `TRANSPORT_DATAFILES` option.

   ```
   $ impdp system@PDB2 FULL=y DIRECTORY=dp_from_source \
   TRANSPORT_DATAFILES='/u02/app/oracle/oradata/ORCL/PDB2/example01.dbf', \
   '/u02/app/oracle/oradata/ORCL/PDB2/fsdata01.dbf',\
   '/u02/app/oracle/oradata/ORCL/PDB2/fsindex01.dbf,'\
   '/u02/app/oracle/oradata/ORCL/PDB2/users01.dbf'
   ```

7. After verifying that the data has been imported successfully, you can delete the `expdat.dmp` dump file.

Data Pump Transportable Tablespace

You can use this method only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

The Transportable Tablespace method is generally much faster than a conventional export/import of the same data because the data files containing all of the actual data are simply copied to the destination location. You use Data Pump to transfer only the metadata of the tablespace objects to the new database.

To migrate an on-premises source database to the database deployment on Oracle Database Cloud Service using the Data Pump Transportable Tablespace method, you perform these tasks:

1. On the on-premises database host, prepare the database for the Data Pump transportable tablespace export.
2. On the on-premises database host, invoke Data Pump Export to perform the transportable tablespace export.
3. Use a secure copy utility to transfer the Data Pump Export dump file and the tablespace datafiles to the Database Cloud Service compute node.
4. Set the on-premises tablespaces back to `READ WRITE`.
5. On the Database Cloud Service compute node, prepare the database for the tablespace import.
6. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database.
7. Set the tablespaces on the Database Cloud Service database to `READ WRITE` mode.

8. After verifying that the data has been imported successfully, you can delete the dump file.

Data Pump Transportable Tablespace: Example

This example provides a step-by-step demonstration of the tasks required to migrate tablespaces in an on-premises Oracle database to an Oracle Database Cloud Service database.

This example performs a migration of the FSDATA and FSINDEX tablespaces.

In this example, the on-premises database is on a Linux host.

1. On the on-premises database host, prepare the database for the Data Pump transportable tablespace export.
   a. On the on-premises database host, create a directory in the operating system to use for the on-premises export.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/dpdump/for_cloud
      ```

   b. On the on-premises database host, invoke SQL*Plus and log in to the on-premises database as the SYSTEM user.

      ```
      $ sqlplus system
      Enter password: <enter the password for the SYSTEM user>
      ```

   c. Create a directory object in the on-premises database to reference the operating system directory.

      ```
      SQL> CREATE DIRECTORY dp_for_cloud AS '/u01/app/oracle/admin/orcl/dpdump/
      for_cloud';
      ```

   d. Determine the name(s) of the datafiles that belong to the FSDATA and FSINDEX tablespaces by querying DBA_DATA_FILES. These files will also be listed in the export output.

      ```
      SQL> SELECT file_name FROM dba_data_files
        2  WHERE tablespace_name = 'FSDATA';

      FILE_NAME
      ----------------------------------------------------------------
      /u01/app/oracle/oradata/orcl/fsdata01.dbf

      SQL> SELECT file_name FROM dba_data_files
        2  WHERE tablespace_name = 'FSINDEX';

      FILE_NAME
      ----------------------------------------------------------------
      /u01/app/oracle/oradata/orcl/fsindex01.dbf
      ```

   e. On the on-premises database host, set all tablespaces that will be transported (the transportable set) to READ ONLY mode.

      ```
      SQL> ALTER TABLESPACE fsindex READ ONLY;
      Tablespace altered.
      SQL> ALTER TABLESPACE fsdata READ ONLY;
      Tablespace altered.
      ```

   f. Exit from SQL*Plus.
2. On the on-premises database host, invoke Data Pump Export to perform the transportable tablespace export.

On the on-premises database host, invoke Data Pump Export and connect to the on-premises database. Export the on-premises tablespaces using the `TRANSPORT_TABLESPACES` option. Provide the password for the `SYSTEM` user when prompted.

```
$ expdp system TRANSPORT_TABLESPACES=fsdata,fsindex TRANSPORT_FULL_CHECK=YES
DIRECTORY=dp_for_cloud
```

3. Use a secure copy utility to transfer the Data Pump Export dump file and the tablespace datafiles to the Database Cloud Service compute node.

   In this example the dump file is copied to the `/u01` directory. Choose the appropriate location based on the size of the file that will be transferred.

   a. On the Database Cloud Service compute node, create a directory for the dump file.

      ```
      $ mkdir /u01/app/oracle/admin/ORCL/dpdump/from_onprem
      ```

   b. Before using the `scp` utility to copy files, make sure the SSH private key that provides access to the Database Cloud Service compute node is available on your on-premises host. For more information about SSH keys, see About Network Access to Database Cloud Service.

   c. On the on-premises database host, use the `scp` utility to transfer the dump file and all datafiles of the transportable set to the Database Cloud Service compute node.

      ```
      $ scp -i private_key_file \
      /u01/app/oracle/admin/orcl/dpdump/for_cloud/expdat.dmp \
      oracle@IP_address_DBaaS_VM:/u01/app/oracle/admin/ORCL/dpdump/from_onprem

      $ scp -i private_key_file \
      /u01/app/oracle/oradata/orcl/fsdata01.dbf \
      oracle@IP_address_DBaaS_VM:/u02/app/oracle/oradata/ORCL

      $ scp -i private_key_file \
      /u01/app/oracle/oradata/orcl/fsindex01.dbf \
      oracle@IP_address_DBaaS_VM:/u02/app/oracle/oradata/ORCL
      ```

4. Set the on-premises tablespaces back to `READ WRITE`.

   a. Invoke SQL*Plus and log in as the `SYSTEM` user.

   b. Set the `FSDATA` and `FSINDEX` tablespaces back to `READ WRITE` mode.

      ```
      SQL> ALTER TABLESPACE fsdata READ WRITE;
      Tablespace altered.
      SQL> ALTER TABLESPACE fsindex READ WRITE;
      Tablespace altered.
      ```

   c. Exit from SQL*Plus.

5. On the Database Cloud Service compute node, prepare the database for the tablespace import.

   a. On the Database Cloud Service compute node, invoke SQL*Plus and log in to the database as the `SYSTEM` user.

   b. Create a directory object in the Database Cloud Service database.

      ```
      SQL> CREATE DIRECTORY dp_from_onprem AS '/u01/app/oracle/admin/ORCL/dpdump/
      from_onprem';
      ```

c. If the owners of the objects that will be imported do not exist in the database, create them before performing the import. The transportable tablespace mode of import does not create the users.

```
SQL> CREATE USER fsowner
  2  PROFILE default
  3  IDENTIFIED BY fspass
  4  TEMPORARY TABLESPACE temp
  5  ACCOUNT UNLOCK;
```

6. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database.

   Import the data into the database using the `TRANSPORT_DATAFILES` option.

```
$ impdp system DIRECTORY=dp_from_onprem \
TRANSPORT_DATAFILES='/u02/app/oracle/oradata/ORCL/fsdata01.dbf', \
'/u02/app/oracle/oradata/ORCL/fsindex01.dbf'
```

7. Set the tablespaces on the Database Cloud Service database to `READ WRITE` mode.

   a. Invoke SQL*Plus and log in as the `SYSTEM` user.

   b. Set the `FSDATA` and `FSINDEX` tablespaces to `READ WRITE` mode.

```
SQL> ALTER TABLESPACE fsdata READ WRITE;
Tablespace altered.
SQL> ALTER TABLESPACE fsindex READ WRITE;
Tablespace altered.
```

   c. Exit from SQL*Plus.

8. After verifying that the data has been imported successfully, you can delete the `expdat.dmp` dump file.

Remote Cloning a PDB

You can use this method only if the on-premises platform is little endian, the on-premises database release is 12.1.0.2 or higher, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

You can use the remote cloning method to copy a PDB from your on-premises Oracle Database 12c database to a PDB in an Oracle Database 12c database on Oracle Database Cloud Service.

To migrate an Oracle Database 12c PDB to a PDB in a Database Cloud Service database deployment using the remote cloning method, you perform these tasks:

1. On the on-premises database host, invoke SQL*Plus and close the on-premises PDB and then reopen it in `READ ONLY` mode.
2. On the Database Cloud Service compute node, invoke SQL*Plus and create a database link that enables a connection to the on-premises database.
3. On the Database Cloud Service compute node, execute the `CREATE PLUGGABLE DATABASE` command to clone the on-premises PDB.
4. On the Database Cloud Service compute node, open the new PDB by executing the `ALTER PLUGGABLE DATABASE OPEN` command.

5. Optionally, on the on-premises database host invoke SQL*Plus and set the on-premises PDB back to `READ WRITE` mode.

For more information, see "Cloning a Remote PDB or Non-CDB" in *Oracle Database Administrator's Guide* for Release 12.2 or 12.1.

Remote Cloning Non-CDB

You can use this method only if the on-premises platform is little endian, the on-premises database release is 12.1.0.2 or higher, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

You can use the remote cloning method to copy an Oracle Database 12c non-CDB on-premises database to a PDB in an Oracle Database 12c database on Oracle Database Cloud Service.

To migrate an Oracle Database 12c non-CDB database to a Database Cloud Service database deployment using the remote cloning method, you perform these tasks:

1. On the on-premises database host, invoke SQL*Plus and set the on-premises database to `READ ONLY` mode.
2. On the Database Cloud Service compute node, invoke SQL*Plus and create a database link that enables a connection to the on-premises database.
3. On the Database Cloud Service compute node, execute the `CREATE PLUGGABLE DATABASE` command to clone the on-premises non-CDB database.
4. On the Database Cloud Service compute node, execute the `$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql` script.
5. On the Database Cloud Service compute node, open the new PDB by executing the `ALTER PLUGGABLE DATABASE OPEN` command.
6. Optionally, on the on-premises database host invoke SQL*Plus and set the on-premises database back to `READ WRITE` mode.

For more information, see "Cloning a Remote PDB or Non-CDB" in *Oracle Database Administrator's Guide* for Release 12.2 or 12.1.

RMAN Cross-Platform Transportable PDB

This method can be used only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

To migrate an Oracle Database 12c PDB to a PDB in an Oracle Database 12c database on an Oracle Database Cloud Service deployment using the RMAN cross-platform transportable PDB method, you perform these tasks:

1. On the on-premises database host, invoke SQL*Plus and close the on-premises PDB.
2. On the on-premises database host, execute the `ALTER PLUGGABLE DATABASE UNPLUG` command to generate an XML file containing the list of datafiles that will be plugged in on the cloud database.
3. On the on-premises database host, invoke RMAN and connect to the root. Execute the `BACKUP FOR TRANSPORT PLUGGABLE DATABASE` command.

4. Use a secure copy utility to transfer the XML file and the backup set to the Database Cloud Service compute node.
5. On the Database Cloud Service compute node, invoke RMAN and connect to the root. Execute the `RESTORE ALL FOREIGN DATAFILES` command.
6. the Database Cloud Service compute node, invoke SQL*Plus and connect to the root. Execute the `CREATE PLUGGABLE DATABASE` command.
7. the Database Cloud Service compute node, execute the `ALTER PLUGGABLE DATABASE OPEN` command.

For more information, see " Performing Cross-Platform Data Transport in CDBs and PDBs" in *Oracle Database Backup and Recovery User's Guide* for Release 12.2 or 12.1.

RMAN Cross-Platform Transportable Tablespace Backup Sets

You can use this method only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

> **Note:**
>
> See *Oracle Database 12c Backup and Recovery User's Guide* for Release 12.2 or 12.1 for detailed information on a similar method that enables you to perform a cross-platform transport of an entire database. When you transport an entire database to a different platform, the source platform and the destination platform must use the same endian format.

To migrate Oracle Database 12c on-premises tablespaces to an Oracle Database 12c database on an Oracle Database Cloud Service deployment using the RMAN cross-platform transportable backup sets method, you perform these tasks:

1. On the on-premises database host, prepare the database by placing the user-defined tablespaces that you intend to transport in `READ ONLY` mode.
2. On the on-premises database host, invoke RMAN and use the `BACKUP` command with the `TO PLATFORM` or `FOR TRANSPORT` clause and the `DATAPUMP` clause to create a backup set for cross-platform transport. See in "BACKUP" in *Oracle Database Backup and Recovery Reference* for Release 12.2 or 12.1 for more information on the `BACKUP` command.
3. Use a secure copy utility to transfer the backup sets, including the Data Pump export dump file, to the Database Cloud Service compute node.
4. Set the on-premises tablespaces back to `READ WRITE`.
5. On the Database Cloud Service compute node, prepare the database by creating the required schemas.
6. On the Database Cloud Service compute node, invoke RMAN and use the `RESTORE` command with the *`foreignFileSpec`* subclause to restore the cross-platform backup.
7. On the Database Cloud Service compute node, set the tablespaces on the database to `READ WRITE` mode.

For more information, see "Overview of Cross-Platform Data Transport Using Backup Sets" in *Oracle Database Backup and Recovery User's Guide* for Release 12.2 or 12.1.

RMAN Cross-Platform Transportable Tablespace Backup Sets: Example

This example provides a step-by-step demonstration of the tasks required to migrate tablespaces in an Oracle Database PDB to an Oracle Database Cloud Service database.

This example performs a migration of the FSDATA and FSINDEX tablespaces.

In this example, the on-premises database is on a Linux host.

1. On the on-premises database host, prepare the database by creating a directory for the export dump file and placing the user-defined tablespaces that you intend to transport in READ ONLY mode..
   a. On the on-premises database host, create a directory in the operating system to use for the export dump.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/dpdump/for_cloud
      ```

   b. On the on-premises data host, invoke SQL*Plus and log in to the PDB as the SYSTEM user..

      ```
      $ sqlplus system@pdb_servicename
      Enter password: enter the password for the SYSTEM user
      ```

   c. Create a directory object in the on-premises database to reference the operating system directory.

      ```
      SQL> CREATE DIRECTORY dp_for_cloud AS '/u01/app/oracle/admin/orcl/dpdump/
      for_cloud';
      ```

   d. On the on-premises database host, set all tablespaces that will be transported (the transportable set) to READ ONLY mode.

      ```
      SQL> ALTER TABLESPACE fsindex READ ONLY;
      SQL> ALTER TABLESPACE fsdata READ ONLY;
      ```

   e. Exit from SQL*Plus.
2. On the on-premises database host, invoke RMAN and use the BACKUP command with the TO PLATFORM or FOR TRANSPORT clause and the DATAPUMP clause to create a backup set for cross-platform transport.
   a. On the on-premises database host, create an operating system directory for the datafiles.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/rman_transdest
      ```

   b. Invoke RMAN and log in as a user that has been granted the SYSDBA or SYSBACKUP privilege.

      ```
      $ rman target username@pdb_servicename
      ```

   c. Execute the BACKUP command.

      ```
      RMAN> BACKUP FOR TRANSPORT
      2> FORMAT '/u01/app/oracle/admin/orcl/rman_transdest/fs_tbs.bck'
      3> TABLESPACE fsdata,fsindex
      4> DATAPUMP FORMAT '/u01/app/oracle/admin/orcl/rman_transdest/fs_tbs.dmp';
      ```

   d. Log out of RMAN.

e. Optionally, navigate to the directory you specified in the BACKUP command to view the files that were created.

```
$ cd /u01/app/oracle/admin/orcl/rman_transdest
$ ls
fs_tbs.bck  fs_tbs.dmp
```

3. Use a secure copy utility to transfer the backup set, including the Data Pump export dump file, to the Database Cloud Service compute node.

a. On the Database Cloud Service compute node, create a directory for the backup set and dump file.

```
$ mkdir /tmp/from_onprem
```

b. Before using the scp command to copy files, make sure the SSH private key that provides access to the Database Cloud Service compute node is available on your on-premises host. For more information about SSH keys, see About Network Access to Database Cloud Service.

c. On the on-premises database host, use the SCP utility to transfer the backup set and the dump file to the Database Cloud Service compute node.

```
$ scp -i private_key_file \
/u01/app/oracle/admin/orcl/rman_transdest/fs_tbs.bck \
oracle@IP_address_DBaaS_VM:/tmp/from_onprem

$ scp -i private_key_file \
/u01/app/oracle/admin/orcl/rman_transdest/fs_tbs.dmp \
oracle@IP_address_DBaaS_VM:/tmp/from_onprem

$
```

4. Set the on-premises tablespaces back to READ WRITE.

a. Invoke SQL*Plus and log in to the PDB as the SYSTEM user.

b. Set the FSDATA and FSINDEX tablespaces back to READ WRITE mode.

```
SQL> ALTER TABLESPACE fsdata READ WRITE;
SQL> ALTER TABLESPACE fsindex READ WRITE;
```

c. Exit from SQL*Plus.

5. On the Database Cloud Service compute node, prepare the database by creating the required schemas.

a. On the Database Cloud Service compute node, invoke SQL*Plus and log in to the PDB as the SYSTEM user.

b. If the owners of the objects that will be imported do not exist in the database, create them before performing the RESTORE.

```
SQL> CREATE USER fsowner
  2  PROFILE default
  3  IDENTIFIED BY fspass
  4  TEMPORARY TABLESPACE temp
  5  ACCOUNT UNLOCK;
```

6. On the Database Cloud Service compute node, invoke RMAN and use the RESTORE command with the *foreignFileSpec* subclause to restore the cross-platform backup.

a. Create an operating system directory for the Data Pump Dump file.

```
$ mkdir /tmp/from_onprem
```

b. Invoke RMAN and log in to the PDB as a user that has been granted the SYSDBA or SYSBACKUP privilege.

```
$ rman target username@pdb_servicename
```

c. Execute the RESTORE command.

```
RMAN> RESTORE FOREIGN TABLESPACE fsdata,fsindex TO NEW
2> FROM BACKUPSET '/tmp/from_onprem/fs_tbs.bck'
3> DUMP FILE DATAPUMP DESTINATION '/tmp/datapump'
4> FROM BACKUPSET '/tmp/from_onprem/fs_tbs.dmp';
```

d. Exit from RMAN.

7. On the Database Cloud Service compute node, set the tablespaces to READ WRITE mode.

a. Invoke SQL*Plus and log in to the PDB as the SYSTEM user.

b. Set the FSDATA and FSINDEX tablespaces to READ WRITE.

```
SQL> ALTER TABLESPACE fsdata READ WRITE;
SQL> ALTER TABLESPACE fsindex READ WRITE;
```

c. Exit from SQL*Plus.

8. After verifying that the data has been imported successfully, you can delete the backup set files that were transported from the on-premises host.

RMAN Transportable Tablespace with Data Pump

You can use this method only if the on-premises platform is little endian, and the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

You can use this method to eliminate placing the tablespaces in READ ONLY mode, as required by the Data Pump Transportable Tablespace method.

To migrate an on-premises source database to a database deployment on Oracle Database Cloud Service using the RMAN Transportable Tablespace with Data Pump method, you perform these tasks:

1. On the on-premises database host, invoke RMAN and create the transportable tablespace set.
2. Use a secure copy utility to transfer the Data Pump Export dump file and the tablespace datafiles to the Database Cloud Service compute node.
3. On the Database Cloud Service compute node, prepare the database for the tablespace import.
4. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database. Import the data into the database using the TRANSPORT_DATAFILES option.
5. After verifying that the data has been imported successfully, you can delete the dump file.

RMAN Transportable Tablespace with Data Pump: Example

This example provides a step-by-step demonstration of the tasks required to migrate tablespaces in an on-premises Oracle database to an Oracle Database Cloud Service database.

This example performs a migration of the FSDATA and FSINDEX tablespaces.

In this example, the on-premises database is on a Linux host.

1. On the on-premises database host, invoke RMAN and create the transportable tablespace set.
   a. On the on-premises database host, create an operating system directory for the datafiles.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/rman_transdest
      ```

   b. On the on-premises data host, create an operating system directory for the RMAN auxiliary instance files.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/rman_auxdest
      ```

   c. Invoke RMAN and log in as the SYSTEM user. Enter the password for the SYSTEM user when prompted.

      ```
      $ rman target system
      ```

   d. Execute the TRANSPORT TABLESPACE command.

      ```
      RMAN> TRANSPORT TABLESPACE fsdata, fsindex
      2> TABLESPACE DESTINATION '/u01/app/oracle/admin/orcl/rman_transdest'
      3> AUXILIARY DESTINATION '/u01/app/oracle/admin/orcl/rman_auxdest';
      ```

   e. Log out of RMAN.

   f. Optionally, navigate to the directory you specified for the TABLESPACE DESTINATION and view the files that were created by the TRANSPORT TABLESPACE operation.

      ```
      $ cd /u01/app/oracle/admin/orcl/rman_transdest
      $ ls
      dmpfile.dmp  fsdata01.dbf  fsindex01.dbf  impscrpt.sql
      ```

2. Use a secure copy utility to transfer the Data Pump Export dump file and the tablespace datafiles to the Database Cloud Service compute node.

   In this example the dump file is copied to the /u01 directory. Choose the appropriate location based on the size of the file that will be transferred.

   a. On the Database Cloud Service compute node, create a directory for the dump file.

      ```
      $ mkdir /u01/app/oracle/admin/ORCL/dpdump/from_onprem
      ```

   b. Before using the scp command to copy files, make sure the SSH private key that provides access to the Database Cloud Service compute node is available on your on-premises host. For more information about SSH keys, see About Network Access to Database Cloud Service.

   c. On the on-premises database host, use the SCP utility to transfer the dump file and all datafiles of the transportable set to the Database Cloud Service compute node.

      ```
      $ scp -i private_key_file \
      /u01/app/oracle/admin/orcl/rman_transdest/dmpfile.dmp \
      oracle@IP_address_DBaaS_VM:/u01/app/oracle/admin/ORCL/dpdump/from_onprem

      $ scp -i private_key_file \
      /u01/app/oracle/admin/orcl/rman_transdest/fsdata01.dbf \
      oracle@IP_address_DBaaS_VM:/u02/app/oracle/oradata/ORCL
      ```

```
$ scp -i private_key_file \
/u01/app/oracle/admin/orcl/rman_transdest/fsindex01.dbf \
oracle@IP_address_DBaaS_VM:/u02/app/oracle/oradata/ORCL
```

3. On the Database Cloud Service compute node, prepare the database for the tablespace import.
   a. On the Database Cloud Service compute node, invoke SQL*Plus and log in to the database as the `SYSTEM` user.
   b. Create a directory object in the Database Cloud Service database.

      ```
      SQL> CREATE DIRECTORY dp_from_onprem AS '/u01/app/oracle/admin/ORCL/dpdump/
      from_onprem';
      ```

   c. If the owners of the objects that will be imported do not exist in the database, create them before performing the import. The transportable tablespace mode of import does not create the users.

      ```
      SQL> CREATE USER fsowner
        2  PROFILE default
        3  IDENTIFIED BY fspass
        4  TEMPORARY TABLESPACE temp
        5  ACCOUNT UNLOCK;
      ```

4. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database.

   Import the data into the database using the `TRANSPORT_DATAFILES` option.

   ```
   $ impdp system DIRECTORY=dp_from_onprem DUMPFILE='dmpfile.dmp' \
   TRANSPORT_DATAFILES='/u02/app/oracle/oradata/ORCL/fsdata01.dbf', \
   '/u02/app/oracle/oradata/ORCL/fsindex01.dbf'
   ```

5. After verifying that the data has been imported successfully, you can delete the `dmpfile.dmp` dump file.

RMAN CONVERT Transportable Tablespace with Data Pump

You can use this method only if the database character sets of your on-premises database and Oracle Database Cloud Service database are compatible.

This method is similar to the Data Pump Transportable Tablespace method, with the addition of the RMAN `CONVERT` command to enable transport between platforms with different endianness. Query `V$TRANSPORTABLE_PLATFORM` to determine if the on-premises database platform supports cross-platform tablespace transport and to determine the endian format of the platform. The Oracle Database Cloud Service platform is little-endian format.

To migrate tablespaces from your on-premises Oracle database to a database deployment on Database Cloud Service using RMAN, you perform these tasks:

1. On the on-premises database host, prepare the database for the Data Pump transportable tablespace export.
2. On the on-premises database host, invoke Data Pump Export to perform the transportable tablespace export.
3. On the on-premises database host, invoke RMAN and use the `CONVERT TABLESPACE` command to convert the tablespace datafile to the Oracle Database

Cloud platform format. Refer to the *Oracle Database Backup and Recovery Reference* for more information on the `CONVERT` command.

4. Use a secure copy utility to transfer the Data Pump Export dump file and the converted tablespace datafiles to the Database Cloud Service compute node.
5. Set the on-premises tablespaces back to `READ WRITE`.
6. On the Database Cloud Service compute node, prepare the database for the tablespace import.
7. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database.
8. On the Database Cloud Service compute node, set the tablespaces in the database to `READ WRITE` mode.
9. After verifying that the data has been imported successfully, you can delete the dump file.

RMAN CONVERT Transportable Tablespace with Data Pump: Example

This example provides a step-by-step demonstration of the tasks required to migrate tablespaces in an on-premises Oracle database to an Oracle Database Cloud Service database.

In this example, the on-premises database is on a Linux host.

1. On the on-premises database host, prepare the database for the Data Pump transportable tablespace export.
   a. On the on-premises database host, create a directory in the operating system to use for the on-premises export.

      ```
      $ mkdir /u01/app/oracle/admin/orcl/dpdump/for_cloud
      ```

   b. On the on-premises database host, invoke SQL*Plus and log in to the on-premises database as the `SYSTEM` user.

      ```
      $ sqlplus system
      Enter password: <enter the password for the SYSTEM user>
      ```

   c. Create a directory object in the on-premises database to reference the operating system directory.

      ```
      SQL> CREATE DIRECTORY dp_for_cloud AS '/u01/app/oracle/admin/orcl/dpdump/
      for_cloud';
      ```

   d. On the on-premises database host, set all tablespaces that will be transported (the transportable set) to `READ ONLY` mode.

      ```
      SQL> ALTER TABLESPACE fsindex READ ONLY;
      Tablespace altered.
      SQL> ALTER TABLESPACE fsdata READ ONLY;
      Tablespace altered.
      ```

   e. Exit from SQL*Plus.
2. On the on-premises database host, invoke Data Pump Export to perform the transportable tablespace export.

   On the on-premises database host, invoke Data Pump Export and connect to the on-premises database. Export the on-premises tablespaces using the

TRANSPORT_TABLESPACES option. Provide the password for the SYSTEM user when prompted.

```
$ expdp system TRANSPORT_TABLESPACES=fsdata,fsindex TRANSPORT_FULL_CHECK=YES
DIRECTORY=dp_for_cloud
```

3. On the on-premises database host, invoke RMAN and use the CONVERT TABLESPACE command to convert the tablespace datafile to the Oracle Database Cloud platform format.

   a. Invoke RMAN.

   ```
   $ rman target /
   ```

   b. Execute the RMAN CONVERT TABLESPACE command to convert the datafiles and store the converted files in a temporary location on the on-premises database host.

   ```
   RMAN> CONVERT TABLESPACE fsdata, fsindex
      2> TO PLATFORM 'Linux x86 64-bit'
      3> FORMAT '/tmp/%U ';
   …
   input datafile file number=00006 name=/u01/app/oracle/oradata/orcl/
   fsdata01.dbf
   converted datafile=/tmp/data_D-ORCL_I-1410251631_TS-FSDATA_FNO-6_0aqc9un3
   …
   input datafile file number=00007 name=/u01/app/oracle/oradata/orcl/
   fsindex01.dbf
   converted datafile=/tmp/data_D-ORCL_I-1410251631_TS-FSINDEX_FNO-7_0bqc9un6
   …
   ```

   c. Take note of the names of the converted files. You will copy these files to the Database Cloud Service compute node in the next step.

   d. Exit RMAN.

4. Use a secure copy utility to transfer the Data Pump Export dump file and the converted tablespace datafiles to the Database Cloud Service compute node.

   In this example the dump file is copied to the /u01 directory. Choose the appropriate location based on the size of the file that will be transferred.

   a. On the Database Cloud Service compute node, create a directory for the dump file.

   ```
   $ mkdir /u01/app/oracle/admin/ORCL/dpdump/from_onprem
   ```

   b. Before using the scp command to copy files, make sure the SSH private key that provides access to the Database Cloud Service compute node is available on your on-premises host. For more information about SSH keys, see About Network Access to Database Cloud Service.

   c. On the on-premises database host, use the scp utility to transfer the dump file and all datafiles of the transportable set to the Database Cloud Service compute node.

   ```
   $ scp -i private_key_file \
   /u01/app/oracle/admin/orcl/dpdump/for_cloud/expdat.dmp \
   oracle@IP_address_DBaaS_VM:/u01/app/oracle/admin/ORCL/dpdump/from_onprem

   $ scp -i private_key_file \
   /tmp/data_D-ORCL_I-1410251631_TS-FSDATA_FNO-6_0aqc9un3 \
   oracle@IP_address_DBaaS_VM:/u02/app/oracle/oradata/ORCL/fsdata01.dbf
   ```

```
$ scp -i private_key_file \
/tmp/data_D-ORCL_I-1410251631_TS-FSINDEX_FNO-7_0bqc9un6 \
oracle@IP_address_DBaaS_VM:/u02/app/oracle/oradata/ORCL/fsindex01.dbf
```

5. Set the on-premises tablespaces back to `READ WRITE`.
   a. Invoke SQL*Plus and log in as the `SYSTEM` user.
   b. Set the `FSDATA` and `FSINDEX` tablespaces back to `READ WRITE` mode.

```
SQL> ALTER TABLESPACE fsdata READ WRITE;
Tablespace altered.
SQL> ALTER TABLESPACE fsindex READ WRITE;
Tablespace altered.
```

   c. Exit from SQL*Plus.
6. On the Database Cloud Service compute node, prepare the database for the tablespace import.
   a. On the Database Cloud Service compute node, invoke SQL*Plus and log in to the database as the `SYSTEM` user.
   b. Create a directory object in the Database Cloud Service database.

```
SQL> CREATE DIRECTORY dp_from_onprem AS '/u01/app/oracle/admin/ORCL/dpdump/
from_onprem';
```

   c. If the owners of the objects that will be imported do not exist in the database, create them before performing the import. The transportable tablespace mode of import does not create the users.

```
SQL> CREATE USER fsowner
  2  PROFILE default
  3  IDENTIFIED BY fspass
  4  TEMPORARY TABLESPACE temp
  5  ACCOUNT UNLOCK;
```

7. On the Database Cloud Service compute node, invoke Data Pump Import and connect to the database.

   Import the data into the DBaaS database using the `TRANSPORT_DATAFILES` option

```
$ impdp system DIRECTORY=dp_from_onprem \
TRANSPORT_DATAFILES='/u02/app/oracle/oradata/ORCL/fsdata01.dbf', \
'/u02/app/oracle/oradata/ORCL/fsindex01.dbf'
```

8. On the Database Cloud Service compute node, set the tablespaces in the database to `READ WRITE` mode.
   a. Invoke SQL*Plus and log in as the `SYSTEM` user.
   b. Set the `FSDATA` and `FSINDEX` tablespaces to `READ WRITE` mode.

```
SQL> ALTER TABLESPACE fsdata READ WRITE;
Tablespace altered.
SQL> ALTER TABLESPACE fsindex READ WRITE;
Tablespace altered.
```

   c. Exit from SQL*Plus.
9. After verifying that the data has been imported successfully, you can delete the `expdat.dmp` dump file.

SQL Developer and INSERT Statements to Migrate Selected Objects

You can use SQL Developer to create a cart into which you add selected objects to be loaded into an Oracle Database 12c database on Oracle Database Cloud Service.

In this method, you use SQL `INSERT` statements to load the data into your cloud database.

To migrate selected objects to an Oracle Database 12c database on a Database Cloud Service deployment using SQL Developer and `INSERT` statements, you perform these tasks:

1. Launch SQL Developer, connect to your on-premises database and create a cart containing the objects you want to migrate.
2. In SQL Developer, click the Export Cart icon and select “Insert” in the Format menu.
3. In SQL Developer, open a connection to the Oracle Database 12c database on Database Cloud Service and execute the generated script to create the database objects.
4. In SQL Developer, open a connection to the Oracle Database 12c database on Database Cloud Service and run the generated script to create the objects and load the data.

SQL Developer and SQL*Loader to Migrate Selected Objects

You can use SQL Developer to create a cart into which you add selected objects to be loaded into an Oracle Database 12c database on Oracle Database Cloud Service.

In this method, you use SQL*Loader to load the data into your cloud database.

To migrate selected objects to an Oracle Database 12c database on a Database Cloud Service deployment using SQL Developer and SQL*Loader, you perform these tasks:

1. Launch SQL Developer, connect to your on-premises database and create a cart containing the objects you want to load into your cloud database.
2. In SQL Developer, click the Export Cart icon and select “loader” in the Format menu.
3. In SQL Developer, open a connection to the Oracle Database 12c database on Database Cloud Service and execute the generated script to create the database objects.
4. Use a secure copy utility to transfer the SQL*Loader control files and the SQL*Loader data files to the Database Cloud Service compute node.
5. On the Database Cloud Service compute node, invoke SQL*Loader to load the data using the SQL*Loader control files and data files for each object.

Unplugging/Plugging a PDB

You can use this method only if the on-premises platform is little endian, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

You can use the unplug/plug method to migrate an Oracle Database 12c PDB to a PDB in an Oracle Database 12c database on an Oracle Database Cloud Service database deployment.

If your source PDB is encrypted, you must export the master encryption key and then import it on the database deployment. See Exporting and Importing a Master Encryption Key for a PDB for details.

To migrate an Oracle Database 12c PDB to a PDB in the Oracle Database 12c database on a Database Cloud Service database deployment using the plug/unplug method, you perform these tasks:

1. On the on-premises database host, invoke SQL*Plus and close the on-premises PDB.
2. On the on-premises database host, execute the `ALTER PLUGGABLE DATABASE UNPLUG` command to generate an XML file containing the list of datafiles that will be plugged in to the database on Database Cloud Service.
3. Use a secure copy utility to transfer the XML file and the datafiles to the Database Cloud Service compute node.
4. On the Database Cloud Service compute node, invoke SQL*Plus and execute the `CREATE PLUGGABLE DATABASE` command to plug the database into the CDB.
5. On the Database Cloud Service compute node, open the new PDB by executing the `ALTER PLUGGABLE DATABASE OPEN` command.

For more information, see "Creating a PDB by Plugging an Unplugged PDB into a CDB" in *Oracle Database Administrator's Guide* for Release 12.2 or 12.1.

Unplugging/Plugging Non-CDB

You can use this method only if the on-premises platform is little endian, and the on-premises database and Database Cloud Service database have compatible database character sets and national character sets.

You can use the unplug/plug method to migrate an Oracle Database 12c non-CDB database to a PDB in an Oracle Database 12c database on an Oracle Database Cloud Service database deployment. This method provides a way to consolidate several non-CDB databases into a single Oracle Database 12c multitenant database on Database Cloud Service.

To migrate an Oracle Database 12c non-CDB database to the Oracle Database 12c database on a Database Cloud Service database deployment using the plug/unplug method, you perform these tasks:

1. On the on-premises database host, invoke SQL*Plus and set the on-premises database to `READ ONLY` mode.
2. On the on-premises database host, execute the `DBMS_PDB.DESCRIBE` procedure to generate an XML file containing the list of datafiles that will be plugged in on the cloud database.
3. Use a secure copy utility to transfer the XML file and the datafiles to the Database Cloud Service compute node.
4. On the Database Cloud Service compute node, invoke SQL*Plus and execute the `CREATE PLUGGABLE DATABASE` command to plug the database into the CDB.

5. On the Database Cloud Service compute node, execute the `$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql` script to delete unnecessary metadata from the `SYSTEM` tablespace of the new PDB.
6. On the Database Cloud Service compute node, open the new PDB by executing the `ALTER PLUGGABLE DATABASE OPEN` command.
7. Optionally, on the on-premises database host invoke SQL*Plus and set the on-premises database back to `READ WRITE` mode.

For more information, see "Creating a PDB Using a Non-CDB" in *Oracle Database Administrator's Guide* for Release 12.2 or 12.1.

10

Frequently Asked Questions for Database Cloud Service

To see a list of frequently asked questions for Oracle Database Cloud Service, see the FAQ page for Oracle Database Cloud at cloud.oracle.com.

11

Troubleshooting Database Cloud Service

This section describes common problems that you might encounter when using Oracle Database Cloud Service and explains how to solve them.

Topics

Problems Creating Deployments

The following solutions apply to problems with creating database deployments on Oracle Database Cloud Service.

I cannot create a deployment when I have many database deployments

Your account might not have enough compute quota to create the deployment.

If you have database deployments you do not need, delete them. If you need all your database deployments, contact Oracle Sales and Services to buy more quota for your account.

I cannot create a deployment, even after waiting for an hour

If deployment creation fails after one hour, the system might be experiencing a heavy load, and resources are not yet available.

Wait before you try again to create the deployment. If this doesn't work, contact Oracle Support.

I get a "SCRIPT execution errors" message when I try to create an deployment with backups to cloud storage

If the password you specify for **Cloud Storage Password** when attempting to create a Database Cloud Service deployment contains certain special characters, the creation attempt fails. Such failures occur after the “SSH access to VM(s)” phase, during the “Configuring Oracle Database Server” phase when configuring bkup assistant. The log for these failures includes an error message that begins with the text “SCRIPT execution errors”.

To circumvent this problem, follow these steps:

1. Create a database deployment and specify **Block Store Only** as the Backup Destination.
2. After the deployment is created, connect as the `oracle` user to the compute node and patch the bkup assistant:

```
$ curl -O https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/
dbaas_patch/bkup/dbaas_patch-21866900.sh
$ chmod +x dbaas_patch-21866900.sh
$ ./dbaas_patch-21866900.sh
Patching /var/opt/oracle/perl_lib/DBAAS/opc_installer.pm
Patch applied successfully.
$
```

3. Change the deployment's backup destination to **Both Cloud Storage and Block Storage** by following the instructions in Changing the Backup Configuration to a Different Backup Destination.

Problems Administering Deployments

The following solutions apply to problems with administering database deployments on Oracle Database Cloud Service.

- I am required to change the password for the oracle user when I try to connect to a compute node
- I get a Linux error 30, Read-only file system, when trying to connect to or work in my environment
- I can't use dbaascli to update my cloud tooling

I am required to change the password for the oracle user when I try to connect to a compute node

You cannot change the password as required because the `oracle` user does not have a password. Instead, change the properties of the `oracle` user so that its password does not expire:

1. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Change the expiration properties of the `oracle` user by using the `chage` command:

   ```
   # /usr/bin/chage -I -1 -m 0 -M 99999 -E -1 oracle
   ```

4. Confirm that the expiration properties have been changed by using the `chage` command again:

   ```
   # /usr/bin/chage -l oracle
   Last password change                                    : date
   Password expires                                        : never
   Password inactive                                       : never
   Account expires                                         : never
   Minimum number of days between password change          : 0
   Maximum number of days between password change          : 99999
   Number of days of warning before password expires       : 7
   ```

5. Close your connection to the compute node.

I get a Linux error 30, Read-only file system, when trying to connect to or work in my environment

In certain rare cases, Oracle Compute Cloud Service sets the access of storage volumes attached to a Database Cloud Service deployment to read-only. When this situation arises, you can restore read-write access by restarting the compute node, as described in Rebooting a Compute Node.

I can’t use dbaascli to update my cloud tooling

Normally, you use the `dbaascli` utility to update the cloud tooling on Database Cloud Service database deployments hosting a single-instance database or Oracle Data Guard configuration, as described in Updating the Cloud Tooling by Using the dbaascli Utility. However, older deployments don’t support the two `dbaascli` subcommands you use:

```
dbaascli patch tools list
dbaascli patch tools apply
```

Nor do older deployments support the following two deprecated `dbaascli` subcommands:

```
dbaascli dbpatchm --run -list_tools
dbaascli dbpatchm --run -toolsinst
```

If you get an error when you run the first of either of these pairs of commands, you must use the following method to update the cloud tooling. After using this method one time, you will be able to use the `dbaascli` utility the next time you want to update the cloud tooling.

1. Connect as the **opc** user to the compute node.

   See Connecting to a Compute Node Through Secure Shell (SSH).

2. Start a root-user command shell:

   ```
   $ sudo -s
   #
   ```

3. Navigate to the **/tmp** directory:

   ```
   # cd /tmp
   ```

4. Download the RPM file containing the latest version of the cloud tooling:

   ```
   # wget https://storage.us2.oraclecloud.com/v1/dbcsswlibp-usoracle29538/
   dbaas_patch/dbaastools.rpm
   ```

5. Get information about the cloud tooling in the downloaded RPM file:

   ```
   # rpm -qpi ./dbaastools.rpm
   Name        : dbaastools                   Relocations: (not relocatable)
   Version     : version_number                    Vendor: Oracle
   Release     : release_number                Build Date: ...
   ...
   ```

6. Get information about the installed cloud tooling:

   ```
   # rpm -qa|grep -i dbaastools
   dbaastools-version_number-release_number
   ```

7. Compare the version and release values of the downloaded cloud tooling and the installed cloud tooling. If the downloaded tooling is newer than the installed tooling, remove the installed tooling and then install the downloaded tooling:
   a. Remove the installed cloud tooling:

      ```
      # rpm -ev installed-info
      ```

      where *installed-info* is the information you noted down about the installed cloud tooling; that is, the output from the `rpm -qa|grep -i dbaastools` command you entered earlier.
   b. Install the cloud tooling in the downloaded RPM file:

      ```
      # rpm -ivh ./dbaastools.rpm
      ```
8. Reset the backup configuration:

   ```
   # /var/opt/oracle/ocde/assistants/bkup/bkup
   ```
9. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

Problems with Scaling

The following solutions apply to problems with scaling an Oracle Database Cloud Service environment.

- My scaling operation does not start
- My deployment is too busy to allow scaling
- After scaling the shape of my Data Guard configuration, I get an ORA-16792 warning when I check the status of the configuration

My scaling operation does not start

The system is overloaded with requests.

Wait before you try to scale again. If that doesn't work, contact Oracle Support.

My deployment is too busy to allow scaling

Your database deployment has a pending maintenance operation such as backup or patching.

Wait until maintenance has completed before you try scaling again.

After scaling the shape of my Data Guard configuration, I get an ORA-16792 warning when I check the status of the configuration

After scaling the shape of a Data Guard configuration, both the DGMGRL `SHOW CONFIGURATION` command and the dbaascli `dataguard status` command report the warning: "ORA-16792: configurable property value is inconsistent with database setting."

Perform the following steps to resolve the inconsistent setting warning:

1. Connect as the `oracle` user to the compute node hosting the primary database. For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).
2. Invoke DGMGRL.

   ```
   $ dgmgrl /
   ```

3. Use the DGMGRL `SHOW DATABASE` command to determine which Data Guard broker property is inconsistent with the database setting.

   ```
   DGMGRL> SHOW DATABASE 'database-name' 'InconsistentProperties';
   ```

   Where *database-name* is the name of the primary database (SID).

   For example:

   ```
   DGMGRL> SHOW DATABASE 'ORCL' 'InconsistentProperties';
   INCONSISTENT PROPERTIES
   INSTANCE_NAME        PROPERTY_NAME         MEMORY_VALUE
   SPFILE_VALUE         BROKER_VALUE
            ORCL     ArchiveLagTarget
   0                                      0
   ```

4. Use the DGMGRL `EDIT DATABASE` command to reset the Data Guard broker property value, which in turn sets the value in the server parameter file (SPFILE).

   ```
   DGMGRL> EDIT DATABASE 'database-name' SET PROPERTY 'property-name'=value;
   ```

   Where:

   - *database-name* is the name of the database (SID)
   - *property-name* is the name of the Data Guard broker property displayed in the output from the `SHOW DATABASE` command
   - *value* is the value displayed in `MEMORY_VALUE` in the output from the `SHOW DATABASE` command.

   For example:

   ```
   DGMGRL> EDIT DATABASE 'ORCL' SET PROPERTY 'ArchiveLagTarget'=0;
   Property "ArchiveLagTarget" updated
   ```

5. Exit from DGMGRL and close your connection to the compute node.

Problems with Patching and Rollback

The following solutions apply to problems with patching and rollback operations on Oracle Database Cloud Service.

- I receive a message stating that the virtual machines are unhealthy
- I receive a message stating that the instance is busy with another operation
- I cannot apply a patch due to a lack of storage space
- My attempt to roll back the January 2015 Patch Set Update (Jan 2015 PSU) fails
- My attempt to roll back the April 2015 Patch Set Update (Apr 2015 PSU) fails

I receive a message stating that the virtual machines are unhealthy

You cannot apply a patch if the compute nodes are not in a healthy state.

Restore the deployment using a backup and try patching again.

I receive a message stating that the instance is busy with another operation

You cannot apply a patch when the deployment is under maintenance, for example, scaling or backup.

Wait until the deployment is no longer under maintenance and try patching again.

I cannot apply a patch due to a lack of storage space

When you apply a patch, storage space is required for temporary files that are created and used during the patching operation. If you receive a message indicating that you don't have sufficient space to patch, either when applying the patch or checking its prerequisites, take these steps:

1. **Check whether you might actually have enough space.**

   The patching tools check whether you have 15 GB free space on `/u01`, but the space actually required for temporary use during patching is somewhat less than 15 GB. To check whether you might have enough space:

   a. Connect as the **`opc`** user to the compute node .

      For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

   b. Display the mounted filesystems:

      ```
      $ df -hT
      ```

   c. Locate the row for `/u01` and check its space available, as reported in the `Avail` column.

   d. If the space available is 13 GB or greater, you have enough space to patch and you can continue to Step e. If you do not have enough space, close your connection and go to Step 2.

   e. Start a root-user command shell:

      ```
      $ sudo -s
      #
      ```

   f. Navigate to the `/var/opt/oracle/patch` directory:

      ```
      # cd /var/opt/oracle/patch
      ```

   g. Use an editor such as `vim` to change the value of the `ignore_space_less_than_15g` key in the `dbpatchm.cfg` file from `0` (zero) to `1` (one):

      ```
      ignore_space_less_than_15g=1
      ```

   h. Apply the patch according to the instructions in Applying a Patch.

2. **Add temporary storage for the temporary files.**

   If you do not have enough space on /u01, you can add temporary storage to the compute node, apply the patch, and then remove the temporary storage:

a. Add 20 GB of temporary storage to the compute node by following these instructions: Adding Temporary Storage to a Database Deployment. When following these instructions, use `/patchtemp` as the mount point directory.

b. Connect to the compute node as the **opc** user.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

c. Start a root-user command shell:

```
$ sudo -s
#
```

d. Navigate to the `/var/opt/oracle/patch` directory:

```
# cd /var/opt/oracle/patch
```

e. Use an editor such as `vim` to edit the `dbpatchm.cfg` file:

   - Change the value of the `temporary_space` key from `"/u01/download"` to `"/patchtemp/download"`
   - Change the value of the `ignore_space_less_than_15g` key from `0` (zero) to `1` (one)

   For example:

```
temporary_space="/patchtemp/download";
...
ignore_space_less_than_15g=1
```

f. Apply the patch according to the instructions in Applying a Patch.

g. Remove the 20 GB of temporary storage you added to the compute node by following these instructions: Deleting Temporary Storage from a Database Deployment

My attempt to roll back the January 2015 Patch Set Update (Jan 2015 PSU) fails

The January 2015 Patch Set Update includes overlay patches and so you need to include the update's overlay patch numbers in the rollback operation. To do so, you must add the overlay patch numbers to the `rollbackpatches.txt` file before rolling back the patch.

1. Connect as the **oracle** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Navigate to the `/var/opt/oracle/patch` directory:

```
$ cd /var/opt/oracle/patch
```

3. Change permissions on the `rollbackpatches.txt` file to make it editable:

```
$ chmod +w rollbackpatches.txt
```

4. Use an editor such as `vim` to add the following lines to the end of the `rollbackpatches.txt` file, making sure to include the colons:

   - For the Jan 2015 PSU on Oracle Database 12.1.0.2:

```
20281121:
19877336:datapatch
```

- For the Jan 2015 PSU on Oracle Database 11.2.0.4:

```
19770063:
19877440:../../sqlpatch/19877440/postdeinstall.sql
```

5. Disconnect from the compute node:

```
$ exit
```

My attempt to roll back the April 2015 Patch Set Update (Apr 2015 PSU) fails

The April 2015 Patch Set Update includes overlay patches and so you need to include the update's overlay patch numbers in the rollback operation. To do so, you must add the overlay patch numbers to the `rollbackpatches.txt` file before rolling back the patch.

1. Connect as the **`oracle`** user to the compute node.

   For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Navigate to the `/var/opt/oracle/patch` directory:

```
$ cd /var/opt/oracle/patch
```

3. Change permissions on the `rollbackpatches.txt` file to make it editable:

```
$ chmod +w rollbackpatches.txt
```

4. Use an editor such as `vim` to add the following lines to the end of the `rollbackpatches.txt` file, making sure to include the colons:

   - For the Apr 2015 PSU on Oracle Database 12.1.0.2:

```
20281121:
20415564:datapatch
```

   - For the Apr 2015 PSU on Oracle Database 11.2.0.4, Standard Edition:

```
19665921:
20406239:../../sqlpatch/20406239/postdeinstall.sql
```

   - For the Apr 2015 PSU on Oracle Database 11.2.0.4, all Enterprise Editions:

```
19665921:
20406239:../../sqlpatch/20406239/postdeinstall.sql
19770063:../../sqlpatch/19770063/postdeinstall.sql
```

5. Disconnect from the compute node:

```
$ exit
```

Problems with Backing Up and Restoring

The following solutions apply to problems with backup and restore operations on Oracle Database Cloud Service.

- There is not enough space for my backup

- I receive a message stating that there was an unexpected error during the duplicate command (ORA messages)
- I receive a message stating that there was an unexpected error during the duplicate command (RMAN messages)
- A backup fails with an ORA-19914 and ORA-28361

There is not enough space for my backup

The backup storage area does not have enough space for the backup operation to create the archive.

Do one of the following:

- Delete any unwanted backups.
- Archive one or more backups to another location.

I receive a message stating that there was an unexpected error during the duplicate command (ORA messages)

There may have been a restart of the database instance while the duplicate operation was running. A datafile may be marked ONLINE or being recovered.

Perform the following steps:

1. Review the `/var/opt/oracle/log/orec/orec.log` file, checking for the following error messages:

   ```
   ORA-01121: cannot rename database file string - file is in use or
   recovery
   ```

   ```
   ORA-01110: data file string
   ```

   If you see those messages, complete the remaining steps in this section. If not, see I receive a message stating that there was an unexpected error during the duplicate command (RMAN messages).
2. Connect to the node hosting the standby database as the `opc` user.
3. Start a root-user command shell and then switch to the **`oracle`** user:

   ```
   $ sudo -s
   # su - oracle
   $
   ```
4. Delete the data file name specified in *`string`* in the error message:

   ```
   $ $ rm -rf filename
   ```
5. Return to being the root user:

   ```
   $ exit
   #
   ```
6. Run the `duplicate` option of `orec`

   ```
   # dbaascli orec --args -duplicate -dbrole standby
   ```
7. Exit the root-user command shell and disconnect from the compute node:

```
# exit
$ exit
```

I receive a message stating that there was an unexpected error during the duplicate command (RMAN messages)

The `orec` tool may have had a problem identifying the target instance.

Perform the following steps:

1. Review the `orec.log` file, checking for the following error messages:

   ```
   RMAN-05501: aborting duplication of target database
   ```

   ```
   RMAN-05502: the target database must be mounted when issuing a
   DUPLICATE
   ```

   If you see those messages, complete the remaining steps in this section. If not, see I receive a message stating that there was an unexpected error during the duplicate command (ORA messages).

2. Connect to the node hosting the standby database as the `opc` user.
3. Start a root-user command shell and then switch to the **`oracle`** user:

   ```
   $ sudo -s
   # su - oracle
   $
   ```

4. Invoke SQL*Plus, connecting as the `SYSDBA` user:

   ```
   $ sqlplus / as sysdba
   ```

5. Shut down the database instance using the `ABORT` option:

   ```
   SQL> shutdown abort;
   ```

6. Exit from SQL*Plus.

   ```
   SQL> exit
   ```

7. Return to being the root user:

   ```
   $ exit
   #
   ```

8. Run the `duplicate` option of `orec`:

   ```
   # dbaascli orec --args -duplicate -dbrole standby
   ```

9. Exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

A backup fails with an ORA-19914 and ORA-28361

The ORA-19914: unable to encrypt backup and ORA-28361: master key not yet set errors indicate that the master encryption keys of a new PDB are not yet set. If you plugged in an encrypted PDB from another CDB and did not import the keys, see Exporting and Importing a Master Encryption Key for a PDB. If you created a new PDB and did not create and activate the master encryption key, see Creating and Activating a Master Encryption Key for a PDB.

Problems with Oracle Data Guard Role Transitions

The following solutions apply to problems with role transitions (failover, reinstate, and switchover) in an Oracle Database Cloud Service deployment hosting an Oracle Data Guard configuration.

- A message in the Activity area indicates that the reinstate operation failed
- A message indicates a problem with the SYS password on the standby database
- After a role transition operation, I get an ORA-16792 warning when I check the status of the configuration

A message in the Activity area indicates that the reinstate operation failed

"Successfully Reinstated" is displayed after the reinstate, but the Activity area on the Overview page indicates that the reinstate operation failed.

Perform the following steps to resolve this discrepancy:

1. On the Database Service console Overview page, click the Refresh Configuration icon.
2. Refresh the Overview page.
3. Click the Refresh Configuration icon again and observe that the Database Role has changed from Reinstate to Standby.

You can ignore the message in the Activity area indicating that the reinstate failed.

A message indicates a problem with the SYS password on the standby database

If you did not use the `database changepassword` subcommand of the `dbaascli` utility to change the password of the `SYS` user in your Oracle Data Guard configuration, the password file on the standby database may not have been updated correctly. See Changing the SYS Password for detailed information on changing the `SYS` user's password.

After a role transition operation, I get an ORA-16792 warning when I check the status of the configuration

After attempting a role transition, both the DGMGRL `SHOW CONFIGURATION` command and the dbaascli `dataguard status` command report the warning: "ORA-16792: configurable property value is inconsistent with database setting."

Perform the following steps to resolve the inconsistent setting warning:

1. Connect as the `oracle` user to the compute node hosting the primary database. For detailed instructions, see Connecting to a Compute Node Through Secure Shell (SSH).
2. Invoke DGMGRL.

```
$ dgmgrl /
```

3. Use the DGMGRL `SHOW DATABASE` command to determine which Data Guard broker property is inconsistent with the database setting.

   ```
   DGMGRL> SHOW DATABASE 'database-name' 'InconsistentProperties';
   ```

   Where *database-name* is the name of the primary database (SID).

   For example:

   ```
   DGMGRL> SHOW DATABASE 'ORCL' 'InconsistentProperties';
   INCONSISTENT PROPERTIES
   INSTANCE_NAME        PROPERTY_NAME         MEMORY_VALUE
   SPFILE_VALUE         BROKER_VALUE
            ORCL     ArchiveLagTarget
   0                                      0
   ```

4. Use the DGMGRL `EDIT DATABASE` command to reset the Data Guard broker property value, which in turn sets the value in the server parameter file (SPFILE).

   ```
   DGMGRL> EDIT DATABASE 'database-name' SET PROPERTY 'property-name'=value;
   ```

   Where:

   - *database-name* is the name of the database (SID)
   - *property-name* is the name of the Data Guard broker property displayed in the output from the `SHOW DATABASE` command
   - *value* is the value displayed in `MEMORY_VALUE` in the output from the `SHOW DATABASE` command.

   For example:

   ```
   DGMGRL> EDIT DATABASE 'ORCL' SET PROPERTY 'ArchiveLagTarget'=0;
   Property "ArchiveLagTarget" updated
   ```

5. Exit from DGMGRL and close your connection to the compute node.

A

Characteristics of a Newly Created Deployment

This section provides information about the content and configuration of a newly created database deployment on Oracle Database Cloud Service. It provides information for the various types of database deployments, beginning with information about features common to most or all types of database deployments.

Topics

Characteristics Common Across Database Deployment Types

While many characteristics of an Oracle Database Cloud Service database deployment depend on the database type of the deployment, some characteristics apply across most or all types of database deployments.

Topics

Data Security

In Oracle Database Cloud Service databases, data security is provided for data in transit and data at rest. Security of data in transit is achieved through network encryption. Security of data at rest is achieved through encryption of data stored in database data files and backups.

Data in Oracle Database files, including backups, is secured by the use of encryption implemented through a key management framework. Security of data across the network is provided by native Oracle Net Services encryption and integrity capabilities.

Topics

- Security of Data in Transit

Security of Data at Rest

Oracle Database Cloud Service uses Oracle Transparent Data Encryption (TDE) to encrypt data in the database data files and in backups. Encrypted data is also protected in temporary tablespaces, undo segments, redo logs and during internal database operations such as JOIN and SORT.

TDE includes a keystore (referred to as a wallet in Oracle Database 11g and previous releases) to securely store master encryption keys, and a management framework to securely and efficiently manage the keystore and perform key maintenance operations.

TDE is the underlying mechanism used for default tablespace encryption and encrypted backups. It uses a two-tiered, key-based architecture to transparently encrypt and decrypt data. The master encryption key is stored in the software keystore. For tablespace encryption, this master encryption key is used to encrypt the tablespace encryption key, which in turn is used to encrypt and decrypt data in the tablespace. Refer to Tablespace Encryption for details on the implementation of tablespace encryption by default in Database Cloud Service.

When a database deployment is created on Database Cloud Service, a local auto-login software keystore is created. The keystore is local to the compute node and is protected by a system-generated password. The auto-login software keystore is automatically opened when accessed.

The keystore location is specified in the `ENCRYPTION_WALLET_LOCATION` parameter in the `$ORACLE_HOME/network/admin/sqlnet.ora` file.

The Oracle keystore stores a history of retired TDE master encryption keys, which enables you to change them and still be able to decrypt data that was encrypted under an earlier TDE master encryption key.

For additional information on TDE and the keystore, refer to "Introduction to Transparent Data Encryption" in *Oracle Database Advanced Security Guide* for Release 18, 12.2 or 12.1 or "Securing Stored Data Using Transparent Data Encryption" in *Oracle Database Advanced Security Administrator's Guide* for Release 11.2.

By default, backups to Cloud Storage for Enterprise Edition databases are encrypted. Recovery Manager (RMAN) performs transparent encryption using the auto-login software keystore. Refer to "Configuring Backup Encryption" in *Oracle Database Backup and Recovery User's Guide* for Release 18, 12.2 or 12.1 or "Encrypting RMAN Backups" in *Oracle Database Backup and Recovery User's Guide* for Release 11.2.

Security of Data in Transit

Oracle Database Cloud Service uses native Oracle Net Services encryption and integrity capabilities to secure connections to the database.

Refer to Using Network Encryption and Integrity for details on how to check your configuration and verify the use of native Oracle Net Services encryption and integrity.

Hybrid Columnar Compression (HCC)

Hybrid Columnar Compression (HCC) is a storage-related Oracle Database feature that causes the database to store the same column for a group of rows together.

Storing column data together in this way can dramatically increase the storage savings achieved from compression. Because database operations work transparently against compressed objects, no application changes are required.

The HCC feature is available on database deployments created on Oracle Database Cloud Service using the Enterprise Edition - High Performance or Enterprise Edition - Extreme Performance software edition.

For more information about HCC, see:

- "Hybrid Columnar Compression" in *Oracle Database Concepts* for Release 18, 12.2 or 12.1
- "Consider Using Table Compression" in *Oracle Database Administrator's Guide* for Release 18, 12.2 or 12.1
- "*table_compression*" in "CREATE TABLE" in *Oracle Database SQL Language Reference* for Release 18, 12.2 or 12.1

Tablespace Encryption

By default, all new tablespaces that you create in a Database Cloud Service database are encrypted.

However, not all of the tablespaces created when you create a database deployment are encrypted:

- In an Oracle Database 11*g* database, none of the tablespaces created when you create a database deployment are encrypted.
- In an Oracle Database 12c Release 1 database, none of the tablespaces created when you create a database deployment are encrypted. This includes the tablespaces in the root (`CDB$ROOT`), the seed (`PDB$SEED`), and the PDB created when you create a database deployment.
- In an Oracle Database 12c Release 2 or later database, only the USERS tablespaces created when you create a database deployment are encrypted. None of the other tablespaces are encrypted. This includes the tablespaces in the root (`CDB$ROOT`), the seed (`PDB$SEED`), and the PDB created when you create a database deployment.

Topics

- Creating Encrypted Tablespaces
- Managing Tablespace Encryption

Creating Encrypted Tablespaces

User-created tablespaces are encrypted by default.

By default, any new tablespaces created by using the SQL `CREATE TABLESPACE` command are encrypted with the AES128 encryption algorithm. You do not need to include the `USING 'encrypt_algorithm'` clause to use the default encryption.

You can specify another supported algorithm by including the `USING 'encrypt_algorithm'` clause in the `CREATE TABLESPACE` command. Supported algorithms are AES256, AES192, AES128, and 3DES168.

Managing Tablespace Encryption

You can manage the software keystore (known as an Oracle wallet in Oracle Database 11*g*), the master encryption key, and control whether encryption is enabled by default.

Managing the Software Keystore and Master Encryption Key

Tablespace encryption uses a two-tiered, key-based architecture to transparently encrypt (and decrypt) tablespaces. The master encryption key is stored in an external security module (software keystore). This master encryption key is used to encrypt the tablespace encryption key, which in turn is used to encrypt and decrypt data in the tablespace.

When the database deployment is created on Database Cloud Service, a local auto-login software keystore is created. The keystore is local to the compute node and is protected by a system-generated password. The auto-login software keystore is automatically opened when accessed.

You can change (rotate) the master encryption key by using the `tde rotate masterkey` subcommand of the `dbaascli` utility. When you execute this subcommand you will be prompted for the keystore password. Enter the password specified during the database deployment creation process. For example:

```
DBAAS>tde rotate masterkey
Executing command tde rotate masterkey
Enter keystore password:
Successfully rotated TDE masterkey
```

For more information about changing the master encryption key, see "Managing the TDE Master Encryption Key" in *Oracle Database Advanced Security Guide* for Release 18, 12.2 or 12.1 or "Setting and Resetting the Master Encryption Key" in *Oracle Database Advanced Security Administrator's Guide* for Release 11.2.

Controlling Default Tablespace Encryption

The `ENCRYPT_NEW_TABLESPACES` initialization parameter controls default encryption of new tablespaces. In Database Cloud Service databases, this parameter is set to `CLOUD_ONLY` by default. See Viewing and Modifying Initialization Parameters for additional information.

Values of this parameter are as follows.

Value	Description
`ALWAYS`	During creation, tablespaces are transparently encrypted with the AES128 algorithm unless a different algorithm is specified in the `ENCRYPTION` clause.

Value	Description
CLOUD_ONLY	Tablespaces created in a Database Cloud Service database are transparently encrypted with the AES128 algorithm unless a different algorithm is specified in the ENCRYPTION clause. For non-cloud databases, tablespaces are only encrypted if the ENCRYPTION clause is specified. This is the default value.
DDL	During creation, tablespaces are not transparently encrypted by default, and are only encrypted if the ENCRYPTION clause is specified.

Note:

With Oracle Database 12c Release 2 (12.2), or later, you can no longer create a new unencrypted tablespace on Database Cloud Service. An error message is returned if you set ENCRYPT_NEW_TABLESPACES to DDL and issue a CREATE TABLESPACE command without specifying an ENCRYPTION clause.

Characteristics of a Single Instance Database Deployment

This section provides information about the content and configuration of a newly created Oracle Database Cloud Service database deployment that uses the Single Instance database type.

When you create a database deployment using the Single Instance database type, Database Cloud Service creates a single-instance Oracle database. The database is housed on a compute node that Database Cloud Service creates using Oracle Compute Cloud Service resources. In brief, Database Cloud Service:

- Creates a compute node that has its own public IP address with a name of this form:

  deployment-name db 1

  This Compute Cloud Service instance uses the compute shape specified during the database deployment creation process.
- Installs Oracle Linux 6.6, Oracle Database 18, 12.2.0.1, 12.1.0.2 or 11.2.0.4 (depending on which version was selected), and cloud tooling software on the compute node.
- Creates storage for database data, the fast recovery area, and the redo logs.
- Creates Oracle Compute Cloud Service networking resources to provide access to the compute node, setting all except SSH access on port 22 to a disabled status.
- Creates and starts an Oracle database on the compute node and starts the network listener for the node.

Topics

- Network Access
- Oracle Database Characteristics
- Location of Diagnostic and Log Files

Linux User Accounts

This section provides information about Linux user accounts that are provisioned on Oracle Database Cloud Service.

Every Database Cloud Service compute node is provisioned with the following operating system user accounts.

User	Description
`opc`	The system administrator account you use with the `sudo` command to perform operations that require root-user access.
`oracle`	The Oracle Database administrator account you use to access the system and perform non-root database administration tasks. A home directory, `/home/oracle`, is created for this user. This user cannot use the `sudo` command to perform operations that require root-user access.
`root`	The root administrator for the system. You do not have direct access to this account. To perform operations that require root-user access, use the `sudo` command as the `opc` user.

The following environment variables are created for the `opc` and `oracle` users.

Variable	Description
`HOME`	The home directory of the user, either `/home/opc` or `/home/oracle`.
`HOSTNAME`	The host name of the compute node.
`LANG`	The system language, `en_US.UTF-8`.
`OH`	Short form for `ORACLE_HOME`.
`ORACLE_HOME`	The Oracle Database home directory: • Oracle Database 18c: `/u01/app/oracle/product/18.0.0/dbhome_1` • Oracle Database 12c Release 2: `/u01/app/oracle/product/12.2.0/dbhome_1` • Oracle Database 12c Release 1: `/u01/app/oracle/product/12.1.0/dbhome_1` • Oracle Database 11g Release 2: `/u01/app/oracle/product/11.2.0/dbhome_1`
`ORACLE_SID`	The database system identifier (SID) provided during the database deployment creation process.

Variable	Description
PATH	The paths to search for executables; set to include: • /sbin • /usr/sbin • /bin • /usr/bin • $ORACLE_HOME/bin • $ORACLE_HOME/OPatch • $HOME
SHELL	The default shell, /bin/bash.
USER	The user name, either opc or oracle.

Storage Volumes and File System Layout

This topic does not apply to Oracle Cloud Infrastructure.

This section provides information about the storage volumes and file system layout of a newly created database deployment on Oracle Database Cloud Service.

Compute Cloud Service Storage Volumes

When a Database Cloud Service deployment is created using the Oracle Database Cloud Service service level, the following storage volumes are created.

Storage Volume	Description
bits	60 GB volume completely allocated to /u01 on the virtual machine.
boot	32 GB volume allocated to the following file system mounts on the virtual machine: • / (root) • /boot • swap space
data	GB size equal to the value provided in the **Usable Data Storage** field during the database deployment creation process, with a minimum of 15 GB. This volume is completely allocated to /u02 on the virtual machine.
fra	GB size depends on the choice of **Backup Destination** during the database deployment creation process: • **Both Cloud Storage and Local Storage**: GB size equal to 1.7 times the size of the **data** volume. • **Cloud Storage Only** or **None**: GB size equal to the size of the **data** volume up to a maximum of 1000 GB. This volume is completely allocated to /u03 on the virtual machine.
redo	25 GB volume completely allocated to /u04 on the virtual machine. (If the data volume is 25 GB, then the redo volume is allocated 26 GB instead.)

File System Layout

When a database deployment is created using the Oracle Database Cloud Service service level, Oracle Cloud Service storage volumes are created and allocated as follows.

File System Mount	Description
swap	Swap space; 4 GB allocated from the **boot** Compute Cloud storage volume.
/ (root)	Operating system files; 25.5 GB allocated from the **boot** Compute Cloud storage volume.
`/boot`	Operating system kernel; 500 MB allocated from the **boot** Compute Cloud storage volume.
`/u01`	Oracle product software; the entire **bits** Compute Cloud storage volume.
`/u02`	Oracle Database data storage; the entire **data** Compute Cloud storage volume.
`/u03`	Database backup storage; the entire **fra** Compute Cloud storage volume.
`/u04`	Database redo logs; the entire **redo** Compute Cloud storage volume.

Locations of Installed Software

This section provides information about the locations of installed software on a newly created Oracle Database Cloud Service database deployment.

When a database deployment is created using the Oracle Database Cloud Service service level, software is installed in the following locations.

Software	Installation Location
Oracle Database	`$ORACLE_HOME`: • Oracle Database 12c Release 2: `/u01/app/oracle/product/12.2.0/dbhome_1` • Oracle Database 12c Release 1: `/u01/app/oracle/product/12.1.0/dbhome_1` • Oracle Database 11g Release 2: `/u01/app/oracle/product/11.2.0/dbhome_1`
Oracle REST Data Services	`/u01/app/oracle/product/apex_listener`
`dbaascli` utility	`/var/opt/oracle/dbaascli`
`bkup_api` utility	`/var/opt/oracle/bkup_api`

Network Access


This topic does not apply to Oracle Cloud Infrastructure.

This section provides information about network access to Oracle Database Cloud Service.

By default, compute node network access on Database Cloud Service is limited to Secure Shell (SSH) connections on port 22. This access restriction ensures that the deployment is secure by default. To access other ports, you can create an SSH tunnel to the port or you can enable access to the port using the Oracle Database Cloud Service console. For more information, see:

- Creating an SSH Tunnel to a Compute Node Port
- Enabling Access to a Compute Node Port

Additionally, the NAT prerouting rules are configured to redirect TCP and UDP on port 80 to port 8080 so that Oracle REST Data Services (ORDS) can service HTTP communication.

To provide network access to the compute node, the following Oracle Compute Cloud Service networking resources are created:

- A permanent IP reservation named **ipreservation** is created and associated with the Compute Cloud Service instance (VM).
- A security list named **ora_db** is created and associated with the compute node. This security list is used in security rules to enable access to specific security applications (port specifications) on the compute node. It is configured with its inbound policy set to DENY and its outbound policy set to PERMIT.
- The following security applications (port specifications) are created so that they can be used in security rules to enable access to specific ports on the compute node:
  - **ora_dbconsole** provides TCP access using port 1158
  - **ora_dbexpress** provides TCP access using port 5500
  - **ora_dblistener** provides TCP access using the listener port that you specified when you created the database deployment
  - **ora_http** provides TCP access using port 80
  - **ora_httpssl** provides TCP access using port 443
- The following security rules are created to enable access to specific ports on the compute node. With the exception of ora_p2_ssh, all these security rules are disabled by default to ensure network security of a newly created deployment. For information about enabling one of these security rules, see Enabling Access to a Compute Node Port.
  - **ora_p2_dbconsole** controls access from the public internet to the **ora_db** security list on the **ora_dbconsole** security application (port 1158 TCP).
  - **ora_p2_dbexpress** controls access from the public internet to the **ora_db** security list on the **ora_dbexpress** security application (port 5500 TCP).
  - **ora_p2_dblistener** controls access from the public internet to the **ora_db** security list on the **ora_dblistener** security application.
  - **ora_p2_http** controls access from the public internet to the **ora_db** security list on the **ora_http** security application (port 80 TCP).
  - **ora_p2_httpssl** controls access from the public internet to the **ora_db** security list on the **ora_httpssl** security application (port 443 TCP).

  - **ora_p2_ssh** controls access from the public internet to the **ora_db** security list on the **ssh** security application (port 22 TCP).

- In addition to the SSH key at the Oracle Database Cloud Service service level, which is referred to or uploaded during the database deployment creation process, a second key is created to permit access to the deployment by Oracle Cloud tools. This key has a name of the form:

  domain-name.dbaas.*deployment-name*.db.tresources.sshkey.ora_tools

Oracle Database Characteristics

When a database deployment is created on Oracle Database Cloud Service, an Oracle database is created using information provided in the Create Instance wizard:

Wizard Page and Field	How Used When Creating the Database
Software Release on the Instance page	Determines which version of Oracle Database is used, 12c Release 2, 12c Release 1 or 11g Release 2.
Software Edition on the Instance page	Determines which database edition is used. The edition determines what database features and options are available. For more information, see Oracle Database Edition.
Usable Data Storage (GB) on the Instance Details page	The amount of data storage for the database data files. A storage volume of this size is created and mounted on /u02.
Administrator Password on the Instance Details page	The password used for the SYS and SYSTEM database users.
DB Name (SID) on the Instance Details page	The database system identifier (SID) of the database.
PDB Name on the Instance Details page	(Oracle Database 12c only) The name of the default pluggable database (PDB) created in the database.

Location of Diagnostic and Log Files

When a database deployment is created on Oracle Database Cloud Service, log files from the creation operation are stored in subdirectories of /var/opt/oracle/log.

By default, Oracle Database trace files and log files are stored in subdirectories of /u01/app/oracle/diag.

Characteristics of a Single Instance with Data Guard Standby Database Deployment

This section provides information about the content and configuration of a newly created Oracle Database Cloud Service database deployment that uses the Oracle Database Cloud Service service level and the Single Instance with Data Guard Standby database type.

When you create a database deployment using the Oracle Database Cloud Service service level and choose the Single Instance with Data Guard Standby database type, Database Cloud Service creates an Oracle Data Guard configuration with a primary database and one physical standby database. Each database is a single-instance

Oracle database housed on a compute node that Database Cloud Service creates using Oracle Compute Cloud Service resources. In brief, Database Cloud Service:

- Creates two compute nodes that are alike in all respects except that each one has its own public IP address.
- Installs Oracle Linux 6.6, Oracle Database 18, 12.2.0.1, 12.1.0.2 or 11.2.0.4 (depending on which version was selected), and cloud tooling software on each of the compute nodes.
- Creates storage for database data, the fast recovery area, and the redo logs.
- Creates Oracle Compute Cloud Service networking resources to provide access to the compute nodes, setting all except SSH access on port 22 to a disabled status.
- Creates and starts an Oracle database on each compute node and starts the network listener for each node.

The following topics provide more detail about this configuration:

Topics

Linux User Accounts

This section provides information about Linux user accounts that are provisioned on an Oracle Database Cloud Service database deployment that uses Oracle Data Guard.

Every Database Cloud Service compute node is provisioned with the following operating system user accounts.

User	Description
`opc`	The system administrator account you use with the `sudo` command to perform operations that require root-user access.
`oracle`	The Oracle Database administrator account you use to access the system and perform non-root database administration tasks. A home directory, `/home/oracle`, is created for this user. This user cannot use the `sudo` command to perform operations that require root-user access.
`root`	The root administrator for the system. You do not have direct access to this account. To perform operations that require root-user access, use the `sudo` command as the `opc` user.

The following environment variables are created for the `opc` and `oracle` users.

Variable	Description
`HOME`	The home directory of the user, either `/home/opc` or `/home/oracle`.
`HOSTNAME`	The host name of the compute node.
`LANG`	The system language, `en_US.UTF-8`.

Variable	Description
OH	Short form for ORACLE_HOME.
ORACLE_HOME	The Oracle Database home directory: • For Oracle Database 18c, /u01/app/oracle/product/18.0.0/dbhome_1 • For Oracle Database 12c Release 2, /u01/app/oracle/product/12.2.0/dbhome_1 • For Oracle Database 12c Release 1, /u01/app/oracle/product/12.1.0/dbhome_1 • For Oracle Database 11g Release 2, /u01/app/oracle/product/11.2.0/dbhome_1
ORACLE_SID	The database system identifier (SID) provided when the deployment was created.
PATH	The paths to search for executables; set to include: • /sbin • /usr/sbin • /bin • /usr/bin • $ORACLE_HOME/bin • $ORACLE_HOME/OPatch • $HOME
SHELL	The default shell, /bin/bash.
USER	The user name, either opc or oracle.

Storage Volumes and File System Layout


This topic does not apply to Oracle Cloud Infrastructure.

This section provides information about the storage volumes and file system layout of a newly created Oracle Database Cloud Service database deployment that uses Oracle Data Guard.

Compute Cloud Service Storage Volumes

When a Database Cloud Service database deployment is created at the Oracle Database Cloud Service service level, the following storage volumes are created.

Storage Volume	Description
bits	60 GB volume completely allocated to /u01 on the virtual machine.
boot	32 GB volume allocated to the following file system mounts on the virtual machine: • / (root) • /boot • swap space

Storage Volume	Description
data	GB size equal to the value provided in the **Usable Data Storage** field during the database deployment creation process, with a minimum of 15 GB. This volume is completely allocated to `/u02` on the virtual machine.
fra	GB size depends on the choice of **Backup Destination** during the database deployment creation process: • **Both Cloud Storage and Local Storage**: GB size equal to 1.7 times the size of the **data** volume. • **Cloud Storage Only** or **None**: GB size equal to the size of the **data** volume up to a maximum of 1000 GB. . This volume is completely allocated to `/u03` on the virtual machine.
redo	25 GB volume completely allocated to `/u04` on the virtual machine.

File System Layout

When a database deployment is created using the Oracle Database Cloud Service service level, Oracle Cloud Service storage volumes are created and allocated as follows.

File System Mount	Description
swap	Swap space; 4 GB allocated from the **boot** Compute Cloud storage volume.
/ (root)	Operating system files; 25.5 GB allocated from the **boot** Compute Cloud storage volume.
`/boot`	Operating system kernel; 500 MB allocated from the **boot** Compute Cloud storage volume.
`/u01`	Oracle product software; the entire **bits** Compute Cloud storage volume.
`/u02`	Oracle Database data storage; the entire **data** Compute Cloud storage volume.
`/u03`	Database backup storage; the entire **fra** Compute Cloud storage volume.
`/u04`	Database redo logs; the entire **redo** Compute Cloud storage volume.

Network Access


This topic does not apply to Oracle Cloud Infrastructure.

This section provides information about network access to a newly created Oracle Database Cloud Service database deployment that uses Oracle Data Guard.

When a Database Cloud Service database deployment is created, compute node network access is limited to Secure Shell (SSH) connections on port 22 by default. This access restriction ensures that the deployment is secure by default. To access other ports, you can create an SSH tunnel to the port or you can enable access to the port using the Oracle Database Cloud Service console. For more information, see:

- Creating an SSH Tunnel to a Compute Node Port

- Enabling Access to a Compute Node Port

Additionally, the NAT prerouting rules are configured to redirect TCP and UDP on port 80 to port 8080 so that Oracle REST Data Services (ORDS) can service HTTP communication.

To provide network access to the compute node, the following Oracle Compute Cloud Service networking resources are created:

- A permanent IP reservation named **ipreservation** is created and associated with the Compute Cloud Service instance (VM).
- A security list named **ora_db** is created and associated with the compute node. This security list is used in security rules to enable access to specific security applications (port specifications) on the compute node. It is configured with its inbound policy set to DENY and its outbound policy set to PERMIT.
- The following security applications (port specifications) are created so that they can be used in security rules to enable access to specific ports on the compute node:
  - **ora_dbconsole** provides TCP access using port 1158
  - **ora_dbexpress** provides TCP access using port 5500
  - **ora_dblistener** provides TCP access using the listener port that you specified when you created the database deployment
  - **ora_http** provides TCP access using port 80
  - **ora_httpssl** provides TCP access using port 443
- The following security rules are created to enable access to specific ports on the computer node. With the exception of ora_p2_ssh, all these security rules are disabled by default to ensure network security of a newly created deployment. For information about enabling one of these security rules, see Enabling Access to a Compute Node Port.
  - **ora_p2_dbconsole** controls access from the public internet to the **ora_db** security list on the **ora_dbconsole** security application (port 1158 TCP).
  - **ora_p2_dbexpress** controls access from the public internet to the **ora_db** security list on the **ora_dbexpress** security application (port 5500 TCP).
  - **ora_p2_dblistener** controls access from the public internet to the **ora_db** security list on the **ora_dblistener** security application.
  - **ora_p2_http** controls access from the public internet to the **ora_db** security list on the **ora_http** security application (port 80 TCP).
  - **ora_p2_httpssl** controls access from the public internet to the **ora_db** security list on the **ora_httpssl** security application (port 443 TCP).
  - **ora_p2_ssh** controls access from the public internet to the **ora_db** security list on the **ssh** security application (port 22 TCP).
- In addition to the SSH key at the Oracle Database Cloud Service service level, which is referred to or uploaded during the database deployment creation process, a second key is created to permit access to the deployment by Oracle Cloud tools. This key has a name of the form:

  domain-name.dbaas.*deployment-name*.db.tresources.sshkey.ora_tools

Oracle Data Guard Configuration

The Oracle Data Guard configuration in an Oracle Database Cloud Service deployment includes a primary database and a single physical standby database. The Oracle Data Guard configuration in an Oracle Database Cloud Service database deployment has the following characteristics:

- Standby Database Type: Physical. The Oracle Data Guard configuration includes a physical standby database. A physical standby database provides a physically identical copy of the primary database, with on-disk database structures that are identical to the primary database on a block-for-block basis.
- Data Protection Mode: Maximum Performance. The Oracle Data Guard configuration uses maximum performance protection mode. This protection mode provides the highest level of data protection that is possible without affecting the performance of a primary database. This is accomplished by allowing transactions to commit as soon as all redo data generated by those transactions has been written to the online log. See Oracle Data Guard Protection Modes in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2 for more information on data protection modes.
- Redo Transport Services Mode: Asynchronous (ASYNC). Redo transport services control the automated transfer of redo data from the primary database to one or more archival destinations in an Oracle Data Guard configuration. The Oracle Data Guard configuration is set to asynchronous (`ASYNC` attribute of the `LOG_ARCHIVE_DEST_`*n* initialization parameter). The asynchronous redo transport mode transmits redo data asynchronously with respect to transaction commitment. A transaction can commit without waiting for the redo generated by that transaction to be successfully sent to any redo transport destination that uses the asynchronous redo transport mode. See Introduction to Redo Transport Services in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2 for more information on redo transport services modes.

Characteristics of a Database Clustering with RAC Database Deployment

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about the content and configuration of a newly created Oracle Database Cloud Service database deployment that uses the Oracle Database Cloud Service service level and the Database Clustering with RAC database type.

When you create a database deployment using the Oracle Database Cloud Service service level and choose the Database Clustering with RAC database type, Database Cloud Service creates a two-node cluster database using Oracle RAC. Each node of the database is housed on a compute node that Database Cloud Service creates using Oracle Compute Cloud Service resources. In brief, Database Cloud Service:

- Creates two compute nodes that are alike in all respects except that each one has its own public IP address.

- Installs Oracle Linux 6.6, Oracle Grid Infrastructure 18, 12.2.0.1 or 12.1.0.2 (depending on which database version was selected), Oracle Database 18, 12.2.0.1, 12.1.0.2 or 11.2.0.4 (depending on which version was selected), and cloud tooling software on each of the compute nodes.
- Creates three Oracle Automatic Storage Management (ASM) disk groups to provide shared storage for database data, the fast recovery area, and the redo logs, and mounts the disk groups as shared file systems on the two compute nodes using Oracle ASM Cluster File System (ACFS).
- Creates Oracle Compute Cloud Service networking resources to provide access to the compute nodes, setting all except SSH access on port 22 to a disabled status.
- Creates and starts a two-node Oracle RAC database on the compute nodes and starts the network listeners for the nodes.

The following topics provide more detail about this configuration:

Topics

Linux User Accounts

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about Linux user accounts that are provisioned on an Oracle Database Cloud Service deployment that hosts an Oracle Real Application Clusters (Oracle RAC) database.

Both compute nodes are provisioned with the following operating system user accounts.

User	Description
`opc`	The system administrator account you use to connect to the compute node using SSH. This user can use the `sudo` command to perform operations that require root-user access.
`oracle`	The Oracle Database administrator account you use to access the system and perform non-root database administration tasks. A home directory, `/home/oracle`, is created for this user. This user cannot use the `sudo` command to perform operations that require root-user access. Additionally, by default you cannot connect as this user to the compute node using SSH. You can add the public key to the user's `$HOME/.ssh/authorized_keys` file to grant persistent SSH access, or you can connect as the `opc` user and then use the `sudo -s` command to start a root-user command shell, followed by an `su - oracle` command to switch to the `oracle` user.

User	Description
`grid`	The Oracle Grid Infrastructure administrator account you use to perform ASM, ACFS, and clusterware administration tasks. A home directory, `/home/grid`, is created for this user. This user cannot use the `sudo` command to perform operations that require root-user access. Additionally, by default you cannot connect as this user to the compute node using SSH. You can add the public key to the user's `$HOME/.ssh/authorized_keys` file to grant persistent SSH access, or you can connect as the `opc` user and then use the `sudo -s` command to start a root-user command shell, followed by an `su - grid` command to switch to the `grid` user.
`root`	The root administrator for the system. You do not have direct access to this account. To perform operations that require root-user access, use the `sudo` command as the `opc` user.

The following environment variable settings are created for the `opc`, `oracle` and `grid` users.

Variable	Description
`HOME`	The home directory of the user, either `/home/opc`, `/home/oracle` or `/home/grid`.
`HOSTNAME`	The host name of the compute node: • *`deployment-name`*`1` for the first compute node • *`deployment-name`*`2` for the second compute node
`LANG`	The system language, `en_US.UTF-8`.
`SHELL`	The default shell, `/bin/bash`.
`USER`	The user name, either `opc`, `oracle` or `grid`.

In addition, the `PATH` variable is also created for all three users, but its value differs (line breaks added to improve clarity):

- For the `opc` user:

```
/opt/oracle/dcs/client/bin:/usr/java/jdk1.7.0_72/bin:
/usr/lib64/qt-3.3/bin:
/usr/local/bin:/bin:/usr/bin:
/usr/local/sbin:/usr/sbin:/sbin:
/home/opc/bin
```

- For the `oracle` user:

```
/usr/lib64/qt-3.3/bin:
/usr/local/bin:/bin:/usr/bin:
/usr/local/sbin:/usr/sbin:/sbin:
/u01/app/oracle/product/db-version/dbhome_1/bin:
/home/oracle/bin
```

  where *`db-version`* is `18.0.0`, `12.2.0.1`, `12.1.0.2` or `11.2.0.2`, depending on which version of Oracle Database was installed.

- For the `grid` user:

```
/usr/lib64/qt-3.3/bin:
/usr/local/bin:/bin:/usr/bin:
/usr/local/sbin:/usr/sbin:/sbin:
```

```
/u01/app/12.1.0.2/grid/bin:
/home/grid/bin
```

In addition, the following environment variable settings are created for the `oracle` user.

Variable	Description
`LD_LIBRARY_PATH`	The Oracle Database library directory: • For Oracle Database 18c, `/u01/app/oracle/product/18.0.0/dbhome_1/lib` • For Oracle Database 12c Release 2, `/u01/app/oracle/product/12.2.0.1/dbhome_1/lib` • For Oracle Database 12c Release 1, `/u01/app/oracle/product/12.1.0.2/dbhome_1/lib` • For Oracle Database 11g Release 2, `/u01/app/oracle/product/11.2.0.2/dbhome_1/lib`
`ORACLE_HOME`	The Oracle Database home directory: • For Oracle Database 18c, `/u01/app/oracle/product/18.0.0/dbhome_1` • For Oracle Database 12c Release 2, `/u01/app/oracle/product/12.2.0.1/dbhome_1` • For Oracle Database 12c Release 1, `/u01/app/oracle/product/12.1.0.2/dbhome_1` • For Oracle Database 11g Release 2, `/u01/app/oracle/product/11.2.0.2/dbhome_1`
`ORACLE_SID`	The database system identifier (SID) for the database instance on the compute node: • *db-sid*1 for the first compute node • *db-sid*2 for the second compute node where *db-sid* is the database system identifer (SID) provided as the **DB Name (SID)** value during the database deployment creation process.
`ORACLE_UNQNAME`	The database system identifer (SID) provided as the **DB Name (SID)** value during the database deployment creation process.

In addition, the following environment variable settings are created for the `grid` user.

Variable	Description
`LD_LIBRARY_PATH`	Set to `/u01/app/12.1.0.2/grid/lib`.
`ORACLE_HOME`	The Oracle Grid Infrastructure home directory: `/u01/app/12.1.0.2/grid`.

Storage Volumes and File System Layout


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about the storage volumes and file system layout of a newly created database deployment on Oracle Database Cloud Service that hosts an Oracle Real Application Clusters (Oracle RAC) database.

When the database deployment is created, the following Oracle Compute Cloud Service storage volumes are created and attached to the two compute nodes.

Storage Volume	Description
boot (two volumes, one for each compute node)	25 GB volume for operating system files, user directories and swap space. This volume appears as the `/dev/xvdb` block device on each compute node.
bits (two volumes, one for each compute node)	70 GB volume for Oracle Database and Oracle Grid Infrastructure software. This volume appears as the `/dev/xvdc` block device on each compute node.
data (one volume accessed by both compute nodes)	Shared storage for database files. GB size equal to the value provided in the **Usable Data Storage** field during the database deployment creation process, with a minimum of 15 GB. This volume appears as the `/dev/xvdd` block device on each compute node.
fra (one volume accessed by both compute nodes)	Shared storage for the fast recovery area. GB size depends on the choice of **Backup Destination** during the database deployment creation process: • **Both Cloud Storage and Local Storage**: GB size equal to 1.7 times the size of the **data** volume. • **None**: GB size equal to 0.7 times the size of the **data** volume, with a minimum of 7 GB This volume appears as the `/dev/xvde` block device on each compute node.
redo (one volume accessed by both compute nodes)	20 GB shared storage volume for redo logs. This volume appears as the `/dev/xvdf` block device on each compute node.

These storage volumes are mounted on the compute nodes as follows.

File System Mount	Description
swap	Swap space; 4 GB allocated from the **boot** Compute Cloud storage volume.
`/` (root)	Operating system files; 16 GB allocated from the **boot** Compute Cloud storage volume.
`/boot`	Operating system kernel; 500 MB allocated from the **boot** Compute Cloud storage volume.
`/u01`	Oracle product software; the entire **bits** Compute Cloud storage volume.
`/u02`	Oracle Database data storage; the entire **data** Compute Cloud storage volume. An Oracle ASM diskgroup named DATA is created on the storage volume and Oracle ACFS is used to mount it.
`/u03`	Database backup storage; the entire **fra** Compute Cloud storage volume. An Oracle ASM diskgroup named FRA is created on the storage volume and Oracle ACFS is used to mount it.
`/u04`	Database redo logs; the entire **redo** Compute Cloud storage volume. An Oracle ASM diskgroup named REDO is created on the storage volume and Oracle ACFS is used to mount it.

Network Access

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about network access to a newly created database deployment on Oracle Database Cloud Service that hosts an Oracle Real Application Clusters (Oracle RAC) database.

By default, compute node network access is limited to Secure Shell (SSH) connections by the `opc` user on port 22. This access restriction ensures that the deployment is secure by default. To access other ports, you can create an SSH tunnel to the port or you can enable access to the port using the Oracle Database Cloud Service console. To provide SSH access to the `oracle` and `grid` users, you can add the public key to the user's `$HOME/.ssh/authorized_keys` file.

To provide network access to the compute nodes, the following Oracle Compute Cloud Service networking resources are created:

- A permanent IP reservation named **ipreservation** is created and associated with the Compute Cloud Service instance (VM).
- A security list named **ora_db** is created and associated with both the compute nodes. This security list permits the two compute nodes to communicate with each other inside the Oracle Cloud, and it is used in security rules to enable access to specific security applications (port specifications) on the compute nodes. It is configured with its inbound policy set to DENY and its outbound policy set to PERMIT.
- The following security applications (port specifications) are created so that they can be used in security rules to enable access to specific ports on the compute nodes:
  - **ora_dbconsole** provides TCP access using port 1158
  - **ora_dbexpress** provides TCP access using port 5500
  - **ora_dblistener** provides TCP access using the listener port that you specified when you created the database deployment
- The following security rules are created to enable access to specific ports on the compute nodes. With the exception of ora_p2_ssh, all these security rules are disabled by default to ensure network security of a newly created deployment. For information about enabling one of these security rules, see Enabling Access to a Compute Node Port.
  - **ora_p2_dbconsole** controls access from the public internet to the **ora_db** security list on the **ora_dbconsole** security application (port 1158 TCP).
  - **ora_p2_dbexpress** controls access from the public internet to the **ora_db** security list on the **ora_dbexpress** security application (port 5500 TCP).
  - **ora_p2_dblistener** controls access from the public internet to the **ora_db** security list on the **ora_dblistener** security application.
  - **ora_p2_ssh** controls access from the public internet to the **ora_db** security list on the **ssh** security application (port 22 TCP).

- In addition to the SSH key maintained at the Oracle Database Cloud Service service level, which is referred to or uploaded during the database deployment creation process, a second key is created to permit access to the deployment by Oracle Cloud tools. This key has a name of the form:

  domain-name.dbaas.*deployment-name*.db.tresources.sshkey.ora_tools

Characteristics of a Database Clustering with RAC and Data Guard Standby Database Deployment

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about the content and configuration of a newly created Oracle Database Cloud Service database deployment that the Oracle Database Cloud Service service level and the Database Clustering with RAC and Data Guard Standby database type.

When you create a database deployment using the Oracle Database Cloud Service service level and choose the Database Clustering with RAC and Data Guard Standby, Database Cloud Service creates two two-node cluster databases using Oracle RAC, one acting as the primary database and one acting as a physical standby database in an Oracle Data Guard configuration. Each node of each cluster database is housed on a compute node that Database Cloud Service creates using Oracle Compute Cloud Service resources. In brief, Database Cloud Service:

- Creates four compute nodes that are alike in all respects except that each one has its own public IP address.
- Installs Oracle Linux 6.6, Oracle Grid Infrastructure 18, 12.2.0.1 or 12.1.0.2 (depending on which database version was selected), Oracle Database 18, 12.2.0.1, 12.1.0.2 or 11.2.0.4 (depending on which version was selected), and cloud tooling software on each of the compute nodes.
- For each cluster database, creates three Oracle Automatic Storage Management (ASM) disk groups to provide shared storage for database data, the fast recovery area, and the redo logs, and mounts the disk groups as shared file systems on the two compute nodes of the cluster using Oracle ASM Cluster File System (ACFS).
- Creates Oracle Compute Cloud Service networking resources to provide access to the compute nodes, setting all except SSH access on port 22 to a disabled status.
- Creates two two-node Oracle RAC databases on the compute nodes and starts the network listeners for the nodes.
- Configures one cluster database as an Oracle Data Guard primary database.
- Configures the other cluster as an Oracle Data Guard physical standby database.
- Starts the databases and the network listeners for the nodes.

The following topics provide more detail about this configuration:

Topics

- Network Access
- Oracle Data Guard Configuration

Linux User Accounts

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about Linux user accounts that are provisioned on an Oracle Database Cloud Service deployment that uses Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard.

All four compute nodes are provisioned with the following operating system user accounts.

User	Description
`opc`	The system administrator account you use to connect to the compute node using SSH. This user can use the `sudo` command to perform operations that require root-user access.
`oracle`	The Oracle Database administrator account you use to access the system and perform non-root database administration tasks. A home directory, `/home/oracle`, is created for this user. This user cannot use the `sudo` command to perform operations that require root-user access. Additionally, by default you cannot connect as this user to the compute node using SSH. You can add the public key to the user's `$HOME/.ssh/authorized_keys` file to grant persistent SSH access, or you can connect as the `opc` user and then use the `sudo -s` command to start a root-user command shell, followed by an `su - oracle` command to switch to the `oracle` user.
`grid`	The Oracle Grid Infrastructure administrator account you use to perform ASM, ACFS, and clusterware administration tasks. A home directory, `/home/grid`, is created for this user. This user cannot use the `sudo` command to perform operations that require root-user access. Additionally, by default you cannot connect as this user to the compute node using SSH. You can add the public key to the user's `$HOME/.ssh/authorized_keys` file to grant persistent SSH access, or you can connect as the `opc` user and then use the `sudo -s` command to start a root-user command shell, followed by an `su - grid` command to switch to the `grid` user.
`root`	The root administrator for the system. You do not have direct access to this account. To perform operations that require root-user access, use the `sudo` command as the `opc` user.

The following environment variable settings are created for the `opc`, `oracle` and `grid` users.

Variable	Description
`HOME`	The home directory of the user, either `/home/opc`, `/home/oracle` or `/home/grid`.
`HOSTNAME`	The host name of the compute node: • *`deployment-name`*`1` for the first compute node • *`deployment-name`*`2` for the second compute node

Variable	Description
LANG	The system language, en_US.UTF-8.
SHELL	The default shell, /bin/bash.
USER	The user name, either opc, oracle or grid.

In addition, the PATH variable is also created for all three users, but its value differs (line breaks added to improve clarity):

- For the opc user:

```
/opt/oracle/dcs/client/bin:/usr/java/jdk1.7.0_72/bin:
/usr/lib64/qt-3.3/bin:
/usr/local/bin:/bin:/usr/bin:
/usr/local/sbin:/usr/sbin:/sbin:
/home/opc/bin
```

- For the oracle user:

```
/usr/lib64/qt-3.3/bin:
/usr/local/bin:/bin:/usr/bin:
/usr/local/sbin:/usr/sbin:/sbin:
/u01/app/oracle/product/db-version/dbhome_1/bin:
/home/oracle/bin
```

  where *db-version* is 18.0.0, 12.2.0.1, 12.1.0.2 or 11.2.0.4, depending on which version of Oracle Database was installed.

- For the grid user:

```
/usr/lib64/qt-3.3/bin:
/usr/local/bin:/bin:/usr/bin:
/usr/local/sbin:/usr/sbin:/sbin:
/u01/app/12.1.0.2/grid/bin:
/home/grid/bin
```

In addition, the following environment variable settings are created for the oracle user.

Variable	Description
LD_LIBRARY_PATH	The Oracle Database library directory: • For Oracle Database 18c, /u01/app/oracle/product/18.0.0/dbhome_1/lib • For Oracle Database 12c Release 1, /u01/app/oracle/product/12.2.0.1/dbhome_1/lib • For Oracle Database 12c Release 1, /u01/app/oracle/product/12.1.0.2/dbhome_1/lib • For Oracle Database 11g Release 2, /u01/app/oracle/product/11.2.0.4/dbhome_1/lib

Variable	Description
ORACLE_HOME	The Oracle Database home directory: • For Oracle Database 18c, /u01/app/oracle/product/18.0.0/dbhome_1 • For Oracle Database 12c Release 2, /u01/app/oracle/product/12.2.0.1/dbhome_1 • For Oracle Database 12c Release 1, /u01/app/oracle/product/12.1.0.2/dbhome_1 • For Oracle Database 11g Release 2, /u01/app/oracle/product/11.2.0.4/dbhome_1
ORACLE_SID	The database system identifier (SID) for the database instance on the compute node: • *db-sid*1 for the first compute node • *db-sid*2 for the second compute node where *db-sid* is the database system identifer (SID) provided as the **DB Name (SID)** value during the database deployment creation process.
ORACLE_UNQNAME	The database system identifer (SID) provided as the **DB Name (SID)** value during the database deployment creation process.

In addition, the following environment variable settings are created for the grid user.

Variable	Description
LD_LIBRARY_PATH	Set to /u01/app/12.1.0.2/grid/lib.
ORACLE_HOME	The Oracle Grid Infrastructure home directory: /u01/app/12.1.0.2/grid.

Storage Volumes and File System Layout


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about the storage volumes and file system layout of a newly created Oracle Database Cloud Service deployment that uses Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard.

When the database deployment is created, the following Oracle Compute Cloud Service storage volumes are created and attached to the four compute nodes.

Storage Volume	Description
boot (four volumes, one for each compute node)	25 GB volume for operating system files, user directories and swap space. This volume appears as the /dev/xvdb block device on each compute node.
bits (four volumes, one for each compute node)	70 GB volume for Oracle Database and Oracle Grid Infrastructure software. This volume appears as the /dev/xvdc block device on each compute node.

Storage Volume	Description
data (two volumes, one for each cluster database)	Shared storage for database files. GB size equal to the value provided in the **Usable Data Storage** field during the database deployment creation process, with a minimum of 15 GB. This volume appears as the `/dev/xvdd` block device on each compute node.
fra (two volumes, one for each cluster database)	Shared storage for the fast recovery area. GB size depends on the choice of **Backup Destination** during the database deployment creation process: • **Both Cloud Storage and Local Storage**: GB size equal to 1.7 times the size of the **data** volume. • **None**: GB size equal to 0.7 times the size of the **data** volume, with a minimum of 7 GB This volume appears as the `/dev/xvde` block device on each compute node.
redo (two volumes, one for each cluster database)	20 GB shared storage volume for redo logs. This volume appears as the `/dev/xvdf` block device on each compute node.

These storage volumes are mounted on the compute nodes as follows.

File System Mount	Description
swap	Swap space; 4 GB allocated from the **boot** Compute Cloud storage volume.
`/` (root)	Operating system files; 16 GB allocated from the **boot** Compute Cloud storage volume.
`/boot`	Operating system kernel; 500 MB allocated from the **boot** Compute Cloud storage volume.
`/u01`	Oracle product software; the entire **bits** Compute Cloud storage volume.
`/u02`	Oracle Database data storage; the entire **data** Compute Cloud storage volume. An Oracle ASM diskgroup named DATA is created on the storage volume and Oracle ACFS is used to mount it.
`/u03`	Database backup storage; the entire **fra** Compute Cloud storage volume. An Oracle ASM diskgroup named FRA is created on the storage volume and Oracle ACFS is used to mount it.
`/u04`	Database redo logs; the entire **redo** Compute Cloud storage volume. An Oracle ASM diskgroup named REDO is created on the storage volume and Oracle ACFS is used to mount it.

Network Access


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

This section provides information about network access to a newly created Oracle Database Cloud Service deployment that uses Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard.

By default, compute node network access is limited to Secure Shell (SSH) connections by the `opc` user on port 22. This access restriction ensures that the deployment is

secure by default. To access other ports, you can create an SSH tunnel to the port or you can enable access to the port using the Oracle Database Cloud Service console. To provide SSH access to the `oracle` and `grid` users, you can add the public key to the user's `$HOME/.ssh/authorized_keys` file.

To provide network access to the compute nodes, the following Oracle Compute Cloud Service networking resources are created:

- A permanent IP reservation named **ipreservation** is created and associated with each Compute Cloud Service instance (VM).
- A security list named **ora_db** is created and associated with all the compute nodes. This security list permits the compute nodes to communicate with each other inside the Oracle Cloud, and it is used in security rules to enable access to specific security applications (port specifications) on the compute nodes. It is configured with its inbound policy set to DENY and its outbound policy set to PERMIT.
- The following security applications (port specifications) are created so that they can be used in security rules to enable access to specific ports on the compute nodes:
  - **ora_dbconsole** provides TCP access using port 1158
  - **ora_dbexpress** provides TCP access using port 5500
  - **ora_dblistener** provides TCP access using the listener port that you specified when you created the database deployment
- The following security rules are created to enable access to specific ports on the compute nodes. With the exception of ora_p2_ssh, all these security rules are disabled by default to ensure network security of a newly created deployment. For information about enabling one of these security rules, see Enabling Access to a Compute Node Port.
  - **ora_p2_dbconsole** controls access from the public internet to the **ora_db** security list on the **ora_dbconsole** security application (port 1158 TCP).
  - **ora_p2_dbexpress** controls access from the public internet to the **ora_db** security list on the **ora_dbexpress** security application (port 5500 TCP).
  - **ora_p2_dblistener** controls access from the public internet to the **ora_db** security list on the **ora_dblistener** security application.
  - **ora_p2_ssh** controls access from the public internet to the **ora_db** security list on the **ssh** security application (port 22 TCP).
- In addition to the SSH key maintained at the Oracle Database Cloud Service service level, which is referred to or uploaded during the database deployment creation process, a second key is created to permit access to the deployment by Oracle Cloud tools. This key has a name of the form:

  domain-name`.dbaas.`*deployment-name*`.db.tresources.sshkey.ora_tools`

Oracle Data Guard Configuration


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The Oracle Data Guard configuration in an Oracle Database Cloud Service deployment that uses Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard has the following characteristics.

- Standby Database Type: Physical. The Oracle Data Guard configuration includes a physical standby database. A physical standby database provides a physically identical copy of the primary database, with on disk database structures that are identical to the primary database on a block-for-block basis.
- Data Protection Mode: Maximum Performance. The Oracle Data Guard configuration uses maximum performance protection mode. This protection mode provides the highest level of data protection that is possible without affecting the performance of a primary database. This is accomplished by allowing transactions to commit as soon as all redo data generated by those transactions has been written to the online log. See "Oracle Data Guard Protection Modes" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2 for more information on data protection modes.
- Redo Transport Services Mode: Asynchronous (ASYNC). Redo transport services control the automated transfer of redo data from the production database to one or more archival destinations in an Oracle Data Guard configuration. The Oracle Data Guard configuration is set to asynchronous (`ASYNC` attribute of the `LOG_ARCHIVE_DEST_`*n* initialization parameter). The asynchronous redo transport mode transmits redo data asynchronously with respect to transaction commitment. A transaction can commit without waiting for the redo generated by that transaction to be successfully sent to any redo transport destination that uses the asynchronous redo transport mode. See "Introduction to Redo Transport Services" in *Oracle Data Guard Concepts and Administration* for Release 18, 12.2, 12.1 or 11.2 for more information on redo transport services modes.

B

Oracle Cloud Pages for Administering Database Cloud Service

This section provides information about what you can do and what you see on each of the Oracle Cloud pages for administering Oracle Database Cloud Service.

Topics

Instances Page

The Oracle Database Cloud Service Instances page displays all deployments on Oracle Database Cloud Service.

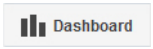
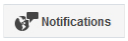
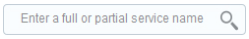
Topics

What You Can Do From the Oracle Database Cloud Service Instances Page

Use the Oracle Database Cloud Service Instances page to perform the tasks described in the following topics:

What You See on the Oracle Database Cloud Service Instances Page

The following table describes the key information shown on the Oracle Database Cloud Service Instances page.

Element	Description
 navigation menu	Navigation menu providing access to My Services and other Oracle Cloud services in the identity domain.
	User menu providing access to help, accessibility options, console version information and sign-out.
	Click to go to the My Services Dashboard page.
	Click to go to the My Services Users page.
	Click to go to the My Services Notifications page.
Activity	Click to go to the Activity Page.
SSH Access	Click to go to the SSH Access Page.
Welcome!	Click to go to the Oracle Database Cloud Service console Welcome page.
REST APIs	Click to go to the API Catalog Cloud Service.
 menu after **REST APIs**	Menu that provides access to Platform Services.
Instances, **OCPUs**, **Memory**, **Storage** and **Public IPs**	Summary of resources being used: • **Instances** — Total number of configured deployments. • **OCPUs** — Total number of Oracle CPUs allocated across all deployments. • **Memory** — Total amount of compute node memory allocated across all deployments. • **Storage** — Total amount of Oracle Compute Cloud Service storage allocated across all deployments. • **Public IPs** — Number of public IP addresses allocated across all deployments.
	Enter a full or partial deployment name to filter the list of deployments to include only those that contain the string in their name.
Create Instance	Click to create a new database deployment on Database Cloud Service. See Creating a Customized Database Deployment.
	Click to view details for the deployment.
Status	Status of the deployment if it is not running. Status values include “In Progress”, “Maintenance”, “Stopped”, and “Terminating”.
Version	Version of Oracle Database configured on the deployment. For example: 12.1.0.2 or 11.2.0.4.
Edition	Software edition of Oracle Database configured on the deployment. For example: Enterprise Edition or Standard Edition.
Created On or **Submitted On**	Date when the deployment was created. During the creation process, the date when the creation request was submitted.
OCPUs	Number of Oracle CPUs associated with the deployment.

Element	Description
Memory	Amount of compute node memory in GBs associated with the deployment.
Storage	Amount of storage in GBs associated with the deployment.
 menu for each deployment	Menu that provides the following options: • **Open DBaaS Monitor Console** — Open the Oracle Cloud Database Monitor for the deployment. • **Open Application Express Console** — Open the Oracle Application Express home page for the deployment. • **Open EM Console** — Open the database console, either Enterprise Manager Database Express 12c or Enterprise Manager 11g Database Control. • **SSH Access** — Add an SSH public key. See Adding an SSH Public Key. • **Access Rules** — Manage access rules that control network access to service components. • **Delete** — Delete the deployment. See Deleting a Database Deployment. **Note:** To ensure security by default, the ports required to access these consoles are initially blocked. To use any of the consoles, you must first enable network access to the console's port or create an SSH tunnel to the console's port. See Accessing Database Cloud Service
Service create and delete history	Listing of attempts to create or delete a deployment. Click the triangle icon next to the title to view the history listing.

Activity Page

The Activity page displays activities for all Oracle Database Cloud Service deployments in your identity domain. You can restrict the list of activities displayed using search filters.

Topics

- What You Can Do From the Activity Page
- What You See on the Activity Page

What You Can Do From the Activity Page

Use the Activity page to view operations for all Database Cloud Service deployments in your identity domain.

You can use the page's Search Activity Log section to filter the list of displayed operations based on:

- The time the operation was started
- The status of the operation
- The name of the deployment on which the operation was performed
- The type of the operation

In the table of results, you can:

- Click any column heading to sort the table by that column.
- Click the triangle at the start of an operation's row to see more details about that operation.

What You See on the Activity Page

The following table describes the key information shown on the Activity page.

Element	Description
Start Time Range	Filters activity results to include only operations started within a specified time range. The range defaults to the previous 24 hours.
Status	Filters operations by status of the operation: • All • Scheduled • Running • Succeeded • Failed You can select any subset of status types. The default value is `All`.
Service Name	Filters the activity results to include operations only for the specified service instance. You can enter a full or partial service instance name.
Service Type	Filters the activity results to include operations only for instances of the specified service type. The default value is the current cloud service.
Operation	Filters the activity results to include selected types of operations. You can select any subset of the given operations. The default value is `All`.
Search	Searches for activities by applying the filters specified by the Start Time Range, Status, Service Name, Service Type and Operation fields, and displays activity results in the table.
Reset	Clears the Start Time Range and Service Name fields, and returns the Status and Operation fields to their default values.
Results per page	Specifies the number of results you want to view per page. The default value is `10`.
	Displays status messages for the given operation. Clicking on the resulting downward arrow hides the status messages.
Service Name	Shows the name of the service instance and its identity domain: *`service_instance`*`:`*`identity_domain`* You can sort the column in ascending or descending order.
Service Type	Shows the type of cloud service for this instance. You can sort the column in ascending or descending order.
Operation	Shows the type of operation performed on the service instance. You can sort the column in ascending or descending order.

Element	Description
Status	Shows the status of the operation performed on the service instance. You can sort the column in ascending or descending order.
Start Time	Shows the time the operation started. You can sort the column in ascending or descending order.
End Time	Shows the time the operation ended, if the operation is complete. You can sort the column in ascending or descending order.
Initiated By	Shows the user that initiated the operation. The user can be any user in the identity domain who initiated the operation or, for certain operations such as automated backup, System. You can sort the column in ascending or descending order.

SSH Access Page

The SSH Access page enables you to view and add SSH public keys to Oracle Database Cloud Service deployments in your identity domain. You can restrict the list of deployments displayed using search filters.

Topics

- What You Can Do From the Activity Page
- What You See on the Activity Page

What You Can Do From the SSH Access Page

Use the SSH Access page to view and add SSH public keys to Database Cloud Service deployments in your identity domain.

You can use the page's Search section to filter the list of displayed deployments based on deployment name.

In the table of results, you can:

- Click any column heading to sort the table by that column.
- Click the triangle at the start of a deployment's row to see more details.

What You See on the SSH Access Page

The following table describes the key information shown on the SSH Access page.

Element	Description
Service Name	Filters the results to include SSH keys only for the specified deployment. You can enter a full or partial deployment name.
Service Type	Filters the results to include SSH keys only for deployments of the specified service type. The default value is the current cloud service.
Search	Searches for SSH keys by applying the filters specified by the Service Name and Service Type fields, and displays the results in the table.
Results per page	Specifies the number of results you want to view per page. The default value is `10`.

Element	Description
	Displays a description of an item in the results table. Clicking on the resulting downward arrow hides the description.
Service Name	Shows the name of the deployment.
Service Type	Shows the type of cloud service for this deployment.
Last Update	Shows the most recent time the SSH keys for this deployment were updated.
Actions	Click the **Add New Key** button to add a new SSH public key to this deployment. The **Add New Key** overlay is displayed with its **Key value** field displaying the deployment's most recent SSH public key. Specify the new public key using one of the following methods: • Select **Upload a new SSH Public Key value** and click **Choose File** to select a file that contains the public key. • Select **Key value**. Delete the current key value and paste the new public key into the text area. Make sure the value does not contain line breaks or end with a line break.

IP Reservations Page

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The Oracle Database Cloud Service IP Reservations page displays the IP reservations you have created. You use the page to view, manage, and create Oracle Compute Cloud Service security rules.

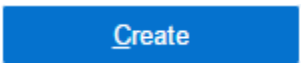
What You Can Do From the Oracle Database Cloud Service IP Reservations Page

Use the IP Reservations page to perform the tasks described in the following topics:

- Creating an IP Reservation
- Deleting an IP Reservation

What You See on the Oracle Database Cloud Service IP Reservations Page

The following table describes the key information shown on the Oracle Database Cloud Service IP Reservations page.

Element	Description
	Launches the IP reservation creation process. Clicking this button will open the Create dialog box wherein you can create an IP reservation and allocate it to a region.
Search	Searches for IP reservations either by name or by region. You can enter a full or partial name and search will return all IP reservations that meet the specified criteria.

Element	Description
IP Reservation list	Lists, by name, all IP reservations created for your service instance and specific details about each reservation. These details are: • Region • Status • IP address • Date the IP reservation was created.
	Clicking this icon deletes the IP Reservation. If the icon is disabled (grayed-out), the reservation is in use and you cannot delete it.

QuickStarts Page

 This topic does not apply to Oracle Cloud Infrastructure.

What You See on the Oracle Database Cloud Service QuickStarts Page

The following table describes the key information shown on the Oracle Database Cloud Service QuickStarts page.

Element	Description
Cancel	Goes to the My Services Dashboard page.
Custom	Goes to the first page of the Create Instance wizard, where you can create a customized database deployment instead of using one of the QuickStart templates.
Instance Name	The name for the new database deployment. The name: • Must start with a letter. • Must not exceed 25 characters in length. • Must contain only letters and numbers. • Must be unique with the identity domain.
Create (below each template)	Displays the Confirmation window, where you download a zip file containing information needed to access the database deployment and then click **Create** to begin creation of the deployment.

Overview Page

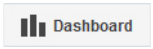
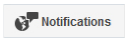
The Oracle Database Cloud Service Overview page displays overview information for an Oracle Database Cloud Service database deployment.

The following tables describe the elements and options available in the various areas of the Overview page:

- What You See in the Banner Area
- What You See in the Tiles Area
- What You See in the Page Content Area

What You See in the Banner Area

The following table describes the elements and options available in the banner area at the top of the page.

Element	Description
 menu	Navigation menu providing access to My Services and other Oracle Cloud services in the identity domain.
	User menu providing access to help, accessibility options, console version information and sign-out.
	Click to go to the My Services Dashboard page.
	Click to go to the My Services Users page.
	Click to go to the My Services Notifications page.
 (next to the “Oracle Database Cloud Service” link)	Click to see details about the database deployment: description, identity domain, subscription type, user who created the deployment, and when the deployment was created.
Oracle Database Cloud Service link	Click to return to the Instances Page.

Element	Description
 (next to the deployment's name)	Deployment menu that provides the following options: • **Open DBaaS Monitor Console** — Open the Oracle Cloud Database Monitor for the deployment. • **Open Application Express Console** — Open the Oracle Application Express home page for the deployment. • **Open EM Console** — Open the database console for the deployment, either Enterprise Manager Database Express 12c or Enterprise Manager 11g Database Control. • **Start** — Start a stopped deployment. See Stopping, Starting and Restarting a Database Deployment. • **Stop** — Stop a deployment. See Stopping, Starting and Restarting a Database Deployment. • **Restart** — Restart a deployment. See Stopping, Starting and Restarting a Database Deployment. • **Switchover** — Start a switchover operation. (Available only for deployments with a Data Guard configuration.) • **Failover** — Start a manual failover operation. (Available only for deployments with a Data Guard configuration.) • **Reinstate** — Start an operation to reinstate a failed primary as the standby. (Available only for deployments with a Data Guard configuration.) • **Scale Up/Down** — Scale the compute shape or storage of the deployment. For information, see Scaling a Database Deployment. (Not available on Oracle Cloud Infrastructure) • **Add Storage** — Add storage to the deployment. For information, see Scaling Up the Storage for a Database Deployment. (Available only on Oracle Cloud Infrastructure) • **Access Rules** — Go to the Access Rules Page to manage the access rules that control network access to the deployment. (Not available on Oracle Cloud Infrastructure) • **SSH Access** — Add an SSH public key to the deployment. See Adding an SSH Public Key. • **Replace Database using Backup** — Replace the database on the deployment using an existing backup stored by Database Backup Cloud Service. See Creating a Database Deployment Using a Cloud Backup. • **View Activity** — Go to the Activity Page to view activities performed on this deployment. **Note:** To ensure security by default, the ports required to access these consoles are initially blocked. To use any of these consoles, you must first enable network access to the console's port or create an SSH tunnel to the console's port. See Accessing Database Cloud Service
	Click to start a stopped deployment.

Element	Description
	Click to stop a running deployment.
	Click to restart a running deployment.
	Click to poll the status of the deployment's compute nodes and display the results on this page.

What You See in the Tiles Area

The following table describes the elements and options available in the tiles area at the side of the page.

Element	Description
Overview tile	The current tile, highlighted to indicate that you are viewing the Overview page.
Administration tile	Click to access these pages for the deployment: • Backup Page • Patching Page • Snapshots Page

What You See in the Page Content Area

The following table describes the elements and options available in the main content area of the page.

Element	Description
	Click to refresh the page.
Service Overview section	Displays a summary box followed by information about the deployment. The summary box shows high-level information about the the deployment: compute nodes, OCPUs, memory, and local storage. Following the summary box is a listing of information about the deployment, including Oracle Database version, Software edition, backup destination, overall status, and so on. Click the **Show more...** link to see even more information about the deployment.
Resources section	Contains an entry for each compute node of the deployment. Each entry displays information about the compute node and provides a menu to perform actions on the compute node.
	(Available only for deployments with a Data Guard configuration.) Click to poll the status of the Data Guard configuration on the deployment's compute nodes and refresh the information on this page.

Element	Description
(for each compute node)	Compute node menu that provides the following options: • **Start** — Start a stopped compute node. See Stopping, Starting and Restarting a Database Deployment . • **Stop** — Stop a compute node. See Stopping, Starting and Restarting a Database Deployment . • **Restart** — Restart a compute node. See Stopping, Starting and Restarting a Database Deployment . • **Switchover** — Start a switchover operation. (Available only for deployments with a Data Guard configuration.) • **Failover** — Start a manual failover operation. (Available only for deployments with a Data Guard configuration.) • **Reinstate** — Start an operation to reinstate a failed primary as the standby. (Available only for deployments with a Data Guard configuration.) • **Scale Up/Down** — Scale the compute shape or storage of a compute node. For information, see Scaling a Database Deployment. (Not available on Oracle Cloud Infrastructure) • **Add Storage** — Add storage to the deployment. For information, see Scaling Up the Storage for a Database Deployment. (Available only on Oracle Cloud Infrastructure) • **Access Rules** — Go to the Access Rules Page to manage the access rules that control network access to the deployment. (Not available on Oracle Cloud Infrastructure)
Data Guard Metrics	(Available only for deployments with a Data Guard configuration.) Displays metrics about the Data Guard configuration.

Access Rules Page

This topic does not apply to Oracle Cloud Infrastructure.

The Oracle Database Cloud Service Access Rules page displays rules used to control network access to Oracle Database Cloud Service deployments. You use the page to view, manage, and create Oracle Compute Cloud Service security rules.

Topics

- What You Can Do From the Oracle Database Cloud Service Access Rules Page
- What You See on the Oracle Database Cloud Service Access Rules Page

What You Can Do From the Oracle Database Cloud Service Access Rules Page

Use the Access Rules page to perform the tasks described in the following topics:

- Enabling Port Access by Enabling an Automatically Created Access Rule
- Enabling or Restricting Port Access by Creating an Access Rule

What You See on the Oracle Database Cloud Service Access Rules Page

The following table describes the key information shown on the Oracle Database Cloud Service Access Rules page.

Element	Description
Results per page	Specifies the number of results you want to view per page. The default value is `10`.
Create Rule	Click to create a new rule. See Enabling or Restricting Port Access by Creating an Access Rule.
Status	Displays an icon that indicates whether a rule is enabled or disabled.
	Indicates the rule is enabled.
	Indicates the rule is disabled.
Rule Name	Name of the rule. When creating a rule, this must start with a letter, followed by letters, numbers, hyphens, or underscores. The name cannot start with `ora_` or `sys_`.
Source	Hosts from which traffic is allowed. Possible values are `DB`, `PUBLIC-INTERNET`, or a custom value in the form of an IP address.
Destination	Security list to which traffic is allowed. This will be `DB`, the `ora_db` security list for the deployment.
Ports	Port or range of ports for the rule.
Description	Description of the rule (optional).
Rule Type	Type of rule. Rule types are: • `DEFAULT`—Rules created automatically when the database deployment was created. Can be enabled or disabled, but not deleted. See Enabling Port Access by Enabling an Automatically Created Access Rule. • `SYSTEM`—Rules created by the system. Cannot be enabled, disabled, or deleted. • `USER`—Rules created by you or another user. Can be enabled, disabled, or deleted.
 (for rule)	Menu that provides the following options: • **Enable**—Enable the rule. • **Disable**—Disable the rule. • **Delete**—Delete the rule (`USER` rules only).

Backup Page

You use the Backup page to manage backup and recovery of a particular database deployment.

What You See on the Oracle Database Cloud Service Backup Page

The following table describes the key information shown on the Oracle Database Cloud Service Backup page.

Element	Description
Backup Now	Click to create a full backup of the database deployment.
Recover	Click to recover the database deployment to the latest backup or to a specific point in time.
Configure Backups	Click to update the credentials for backing up to cloud storage.
(for each available backup)	Menu that provides the **Recover** option. Choose this option to recover to the given backup.
Recovery History	Listing of recovery operations on the database deployment. Click the triangle icon next to the title to view the listing.

Patching Page

You use the Patching page to view available patches, initiate a patching process, and view details of the last patching process for a particular database deployment.

What You See on the Oracle Database Cloud Service Patching Page

The following table describes the key information shown on the Oracle Database Cloud Service Patching page.

Element	Description
Available Patches	A list of patches you can apply to the deployment.
(for each listed patch)	**Menu** icon provides the following options for the patch: • **Precheck** — Check whether the patch can be successfully applied to the deployment. • **Patch** — Apply the patch to the deployment.
Details of Last Patching Activity	Expand to see a description of the actions taken during the last patching operation.
Rollback	Click to roll back the last patching operation. See Rolling Back a Patch or Failed Patch by Using the Oracle Database Cloud Service Console.

Snapshots Page

This topic does not apply to Oracle Cloud Infrastructure.

You use the Snapshots page to create snapshots of the Oracle Compute Cloud Service storage volumes that support a database deployment hosting a single-instance database. You can then use these snapshots to create database deployments quickly because their storage is linked to the snapshot's storage using "copy on write" technology provided by Oracle Compute Cloud Service.

What You See on the Oracle Database Cloud Service Snapshots Page

The following table describes the key information shown on the Oracle Database Cloud Service Snapshots page.

Element	Description
Create Storage Snapshot	Click to create a storage snapshot, which can be used to create a new database deployment called a linked clone.
 (for each available snapshot)	Menu that provides the following options: • **Create Database Clone** — Create a linked-clone deployment. • **Delete** — Delete a snapshot .

Create Instance: Instance Page

Create Instance: Instance is the first page in the wizard you use to create a new database deployment, as described in Creating a Customized Database Deployment.

What You See in the Navigation Area

Element	Description
Cancel	Click to cancel the Create Instance wizard without creating a new database deployment.
Next>	Click to advance to the Create Instance: Instance Details page.

What You See in the Page Content Area

The following table describes the key information shown on the Create Instance: Instance page.

Element	Description
Instance Name	The name for the new database deployment. The name: • Must not exceed 50 characters. • Must start with a letter. • Must contain only letters, numbers, or hyphens. • Must not contain any other special characters. • Must be unique within the identity domain.
Description	(Optional) A description for the new database deployment.
Notification Email	(Optional) An email address where you would like updates about the deployment-creation operation to be sent.
Region	(Available only if your identity domain is enabled for regions.) The region for the database deployment. If you choose a region that supports Oracle Cloud Infrastructure, the **Availability Domain** and **Subnet** fields are displayed, and the deployment will be created on Oracle Cloud Infrastructure. Otherwise, the deployment will be created on Oracle Cloud Infrastructure Classic. Choose **No Preference** to let Database Cloud Service choose an Oracle Cloud Infrastructure Classic region for you.
Availability Domain	(Available only on Oracle Cloud Infrastructure) The availability domain (within the region) where the database deployment will be placed.
Subnet	(Available only on Oracle Cloud Infrastructure) The subnet (within the availability domain) that will determine network access to the database deployment.

Element	Description
IP Network	(Available only if you have selected an Oracle Cloud Infrastructure Classic region and you have defined one or more IP networks created in that region using Oracle Cloud Infrastructure Compute Classic.) Select the IP network where you want the database deployment placed. Choose **No Preference** to use the default shared network provided by Oracle Cloud Infrastructure Compute Classic. For more information about IP networks, see these topics in *Using Oracle Cloud Infrastructure Compute Classic*: • About IP Networks • Creating an IP Network
Assign Public IP	(Available only if you have selected an IP network.) Choose whether to assign public IP addresses to the compute nodes in your database deployment. If you select this check box (default), then any node added during deployment creation, or later added as part of a scaling operation, will have a public IP address assigned to it. You will be able to directly access the nodes from the public Internet. If you deselect this check box, then any node added during deployment creation, or later added as part of a scaling operation, will not have a public IP address assigned to it. You will not be able to directly access the nodes from the public Internet. This selection is for use cases where you intend to access the nodes and the database only from within your IP network or from your on-premises data center over a VPN network.
License Type	(Available only in accounts that use the Universal Credits payment model) Controls how the right to use Oracle Database on the new deployment is established. To use the "Bring Your Own License" (BYOL) feature, which enables you to use an existing perpetual Oracle Database license to establish the right to use Oracle Database on a deployment, select **My organization already owns Oracle Database software licenses. Bring my existing database software license to the Oracle Database Cloud Service.** Your Oracle Cloud account will be charged a lesser amount for the new deployment because the right to use Oracle Database is covered by your perpetual license agreement. To use your Oracle Cloud account, select **Subscribe to a new Oracle Database software license and the Oracle Database Cloud Service.** Your account will be charged for the new deployment according to your Oracle Database Cloud Service agreement.
Service Level	(Available only in accounts that include Oracle Database Exadata Cloud Service or old accounts that predate the Universal Credits payment model.) The service level for the new deployment: • **Oracle Database Cloud Service** is the service level you should choose for Database Cloud Service. • **Oracle Database Cloud Service - Virtual Image** (Not available on Oracle Cloud Infrastructure)

Element	Description
Metering Frequency	(Available only in old accounts that predate the Universal Credits payment model.) The metering frequency for the new deployment: • **Hourly** • **Monthly**
Software Release	The release version of Oracle Database for the new deployment: • **Oracle Database 11g Release 2** • **Oracle Database 12c Release 1** • **Oracle Database 12c Release 2** • **Oracle Database 18c** See Oracle Database Software Release.
Software Edition	The Oracle Database software package for the new deployment: • **Standard Edition** • **Enterprise Edition** • **Enterprise Edition - High Performance** • **Enterprise Edition - Extreme Performance** See Oracle Database Software Edition.

Element	Description
Database Type	The type of deployment to create: • **Single Instance**—A single Oracle Database instance and database data store hosted on one compute node. For more information about this type, see Characteristics of a Single Instance Database Deployment. • **Database Clustering with RAC**—A two-node clustered database using Oracle Real Application Clusters technology; two compute nodes each host an Oracle Database instance, and the two instances access the same shared database data store. For more information about this type, see Characteristics of a Database Clustering with RAC Database Deployment. (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer) • **Single Instance with Data Guard Standby**—Two single-instance databases, one acting as the primary database and one acting as the standby database in an Oracle Data Guard configuration. For more information about this type, see Characteristics of a Single Instance with Data Guard Standby Database Deployment. • **Database Clustering with RAC and Data Guard Standby** —Two two-node Oracle RAC databases, one acting as the primary database and one acting as the standby database in an Oracle Data Guard configuration. For more information about this type, see Characteristics of a Database Clustering with RAC and Data Guard Standby Database Deployment. (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer) • **Data Guard Standby for Hybrid DR** — Single-instance database acting as the standby database in an Oracle Data Guard configuration. The primary database is on your own system. (Not available on Oracle Cloud Infrastructure or on Oracle Cloud at Customer) Not all types are available with all combinations of service level and software edition: • Single Instance is the only type supported by the Standard Edition software edition. • The two types that use Oracle RAC are available only with Enterprise Edition - Extreme Performance software edition.

Create Instance: Instance Details Page

Create Instance: Instance Details is a page in the Create Instance wizard you use to create a new database deployment. For more information, see Creating a Customized Database Deployment.

The following tables describe the key information shown on the Create Instance: Instance Details page:

- What You See in the Navigation Area
- What You See in the Database Configuration Section
- What You See in the Backup and Recovery Configuration Section

- What You See in the Initialize Data From Backup Section
- What You See in the Standby Database Section

What You See in the Navigation Area

Element	Description
<Previous	Click to return to the Create Instance: Instance page.
Cancel	Click to cancel the Create Instance wizard without creating a new database deployment.
Next>	Click to advance to the Create Instance: Confirmation page.

What You See in the Database Configuration Section

Element	Description
DB Name (SID)	The name for the database instance. The name: • Must not exceed 8 characters. • Must start with a letter. • Must contain only letters and numbers.
PDB Name	(Available only for Oracle Database 12c or later.) The name for the default pluggable database (PDB). The name: • Must not exceed 8 characters. • Must start with a letter. • Must contain only letters, numbers, or these symbols: _ (underscore), # (hash sign), or $ (dollar sign). This option is not available if Create Instance from Existing Backup is set to Yes.
Administration Password **Confirm Password**	The password for the following administrative users: • Oracle Database administrative users • Oracle Application Express admin user The password: • Must be 8 to 30 characters in length. • Must contain at least one lowercase letter • Must contain at least one uppercase letter • Must contain at least one number • Must contain at least one of these symbols: _ (underscore), # (hash sign), or $ (dollar sign). • Must not contain the word “oracle”.
Usable Database Storage (GB)	The amount of storage in GB for actual database data. Note that up to 8% of this storage will be used for file system constructs and other overhead.
Total Data File Storage (GB)	The computed amount of storage in GB that will be allocated to the deployment, including space for operating system and product binaries, supporting files, database data and configuration files, and so on.
Compute Shape	The number of Oracle Compute Units (OCPUs) and amount of memory (RAM) for each compute node of the new database deployment. Database Cloud Service offers several OCPU/RAM combinations, as described in Computing Power.

Element	Description
SSH Public Key **Edit**	The SSH public key to be used for authentication when using an SSH client to connect to a compute node that is associated with your database deployment. Click **Edit** to specify the public key. You can upload a file containing the public key value, paste in the value of a public key, or create a system-generated key pair. If you paste in the value, make sure the value does not contain line breaks or end with a line break.
Use High Performance Storage	(Available only if you have a metered subscription and you chose an Oracle Cloud Infrastructure Classic region on the wizard's Instance page.) Controls the device type to be used for database block storage. By default, block storage is allocated on spinning devices. If you select this option, then block storage will be allocated on solid state devices, at an increased cost. For pricing details, refer to the Block Storage information at https://cloud.oracle.com/compute-classic/pricing.
Advanced Settings: Listener Port	The port number for the Oracle Net Listener. The port number must be between 1521 and 5499 (inclusive).
Advanced Settings: Timezone	The time zone for the new database deployment. The default is Coordinated Universal Time (UTC).
Advanced Settings: Character Set	The database character set for the database. The database character set is used for: • Data stored in SQL `CHAR` data types (`CHAR`, `VARCHAR2`, `CLOB`, and `LONG`) • Identifiers such as table names, column names, and PL/SQL variables • Entering and storing SQL and PL/SQL source code This option is not available if Create Instance from Existing Backup is set to Yes.
Advanced Settings: National Character Set	The national character set for the database. The national character set is used for data stored in SQL `NCHAR` data types (`NCHAR`, `NCLOB`, and `NVARCHAR2`). This option is not available if Create Instance from Existing Backup is set to Yes.
Advanced Settings: Enable Oracle GoldenGate	(Not available on Oracle Cloud at Customer) Configures the database for use as the replication database of an Oracle GoldenGate Cloud Service instance. See Using Oracle GoldenGate Cloud Service with Database Cloud Service.
Advanced Settings: Include "Demos" PDB	(Available only for Oracle Database 12c Release 1.) Controls whether the "Demos" PDB is to be included in the database. This PDB contains demos for many new features of Oracle Database 12c such as in-memory and multitenant. Usable Data File Storage must to be at least 25 GB to include this PDB.

Element	Description
Advanced Settings: IP Reservations	(Not available on Oracle Cloud Infrastructure) (Available only if you chose an Oracle Cloud Infrastructure Classic region on the wizard's Instance page and did not deselect the **Assign Public IP** option.) Specifies whether to use an IP reservation for this deployment. If you choose **Assign Automatically**, an IP reservation is not used and Database Cloud Service acquires a new IP address for use by the deployment. Otherwise, Database Cloud Service uses the IP reservation you choose.

What You See in the Backup and Recovery Configuration Section

Element	Description
Backup Destination	Controls how backups for the deployment are to be configured: • **Both Cloud Storage and Local Storage**—backups are configured to be created automatically and stored both on local storage and on cloud storage. If this choice is selected, the Cloud Storage Container, User Name and Password fields are displayed. • **Cloud Storage Only** — backups are configured to be created automatically and stored on cloud storage. If this choice is selected, the Cloud Storage Container, User Name and Password fields are displayed. **Note:** This choice is not currently available for database deployments that use Oracle Real Application Clusters (Oracle RAC). • **None**—Backups are not configured for the deployment. For more information about backups and backup configurations, see About Backing Up Database Deployments on Database Cloud Service.

Element	Description
Cloud Storage Container	The location where backups to cloud storage are to be stored: • For database deployments in Oracle Cloud Infrastructure, enter the URL of an existing Oracle Cloud Infrastructure Object Storage bucket. The URL is of the form: `https://swiftobjectstorage.`*`region`*`.oraclecloud.com/v1/`*`namespace/bucket`* For example: `https://swiftobjectstorage.us-phoenix-1.oraclecloud.com/v1/mycompany/mybucket` You must create this storage bucket before you begin creating the database deployment. See Object Storage API in Oracle Cloud Infrastructure documentation. • For database deployments in Oracle Cloud Infrastructure Classic, enter the location of an Oracle Cloud Infrastructure Object Storage Classic container using this form: `Storage-`*`identity_domain/container`* where *`identity_domain`* is the id of the identity domain, and *`container`* is the name of the container. If this container doesn't exist, use the **Create Cloud Storage Container** checkbox to create it. **Note:** In some Oracle Cloud Infrastructure Classic accounts, you cannot use the above form. If you get an error when you try to use this form, you must instead provide a full URL for the container using this form: *`rest-endpoint-url/container`* To determine the *`rest-endpoint-url`* value for your account, see About REST URLs for Oracle Cloud Infrastructure Object Storage Classic Resources in *Using Oracle Cloud Infrastructure Object Storage Classic*.
Cloud Storage User Name	A user with read/write (and creation, if necessary) access to the location specified in **Cloud Storage Container**: • For database deployments in Oracle Cloud Infrastructure, enter the user name you use to sign in to the Oracle Cloud Infrastructure console. • For database deployments in Oracle Cloud Infrastructure Classic, enter the Oracle Cloud user name of the administrator of the Oracle Cloud Infrastructure Object Storage Classic container specified in **Cloud Storage Container**. Usually, this is your Oracle Cloud user name.
Cloud Storage Password	The password necessary to access the location specified in **Cloud Storage Container**: • For database deployments in Oracle Cloud Infrastructure, enter your Swift password (auth token). • For database deployments in Oracle Cloud Infrastructure Classic, enter the password of the Oracle Cloud user specified in **Cloud Storage User Name**.

Element	Description
Create Cloud Storage Container	(Not available on Oracle Cloud Infrastructure) Create a new Oracle Cloud Infrastructure Object Storage Classic container as part of the database deployment creation. Specify the container name and the Cloud Storage user name and password in the preceding fields.
Total Estimated Monthly Storage	Storage for data files and backups.

What You See in the Initialize Data From Backup Section

Element	Description
Create Instance from Existing Backup	Create a database deployment whose database is derived from a cloud backup created using Oracle Database Backup Cloud Service. The other fields and options in the Initialize Data From Backup section only display if Create Instance from Existing Backup is set to Yes.
On-Premises Backup	Indicates the origin of the source database backup. Select this option if the source database backup is not from another Database Cloud Service database deployment in the same identify domain. In this case, the following fields and options are displayed except for Source Service Name. Deselect this option if the source database backup is from another Database Cloud Service database deployment in the same identify domain. In this case, only the Source Instance Name and Backup Tag fields are displayed.
Database ID	The database identifier of the database from which the existing backup was created. You can get this value by using the following SQL query: `SQL> SELECT dbid FROM v$database;`
Decryption Method **Edit**	Specifies the information necessary to decrypt the source database backup. Click **Edit** to specify the necessary information. In the resulting dialog: • For a backup that uses Transparent Database Encryption (TDE), select **Upload Wallet File** then click **Browse** and specify a zip file containing the source database's TDE wallet directory, and the contents of that directory. **Note:** If the source database is from another Database Cloud Service database deployment, its TDE wallet directory is `/u01/app/oracle/admin/`*`dbname`*`/tde_wallet`. • For a backup that uses password encryption, select **Paste RMAN Key Value** and paste the password (key value) used to encrypt the backup.

Element	Description
Cloud Storage Container	The URL where the existing backup is stored: • The URL of an Oracle Cloud Infrastructure Object Storage bucket. The URL is of the form: `https://swiftobjectstorage.`*`region`*`.oraclecloud.com/v1/`*`namespace/bucket`* For example: `https://swiftobjectstorage.us-phoenix-1.oraclecloud.com/v1/mycompany/mybucket` • The URL of an Oracle Cloud Infrastructure Object Storage Classic container. The URL is of the general form: *`rest-endpoint-url/container`*
Username	The Oracle Cloud user name of the administrator of the Oracle Cloud Infrastructure Object Storage Classic container specified in **Cloud Storage Container**.
Password	The password of the user specified in **Username**.
Source Instance Name	From the list of possible alternatives, specify the database deployment whose source database backup you want to use.
Backup Tag	A list of backups available for the specified database deployment. The latest backup is selected by default, but you can choose an earlier backup.

What You See in the Standby Database Section

Element	Description
Standby Database Configuration	Controls where the standby database is placed in relation to the primary database: • **High Availability**—The standby database is placed in a different availability domain from the primary database, thus providing isolation at the infrastructure level. • **Disaster Recovery**—The standby database is placed in a different data center from the primary database, thus providing isolation at the infrastructure level and geographical separation to support availability despite catastrophic events. See Using Oracle Data Guard in Database Cloud Service for more information. If you choose this option, the Enable Oracle GoldenGate option is disabled.

Create Instance: Confirmation Page

Create Instance: Confirmation is the final page in the Create Instance wizard you use to create a new database deployment. For more information, see Creating a Customized Database Deployment.

What You See on the Create Instance: Confirmation Page

The Create Instance: Confirmation page presents a summary list of all the choices you made on the preceding pages of the Create Instance wizard. In addition, it provides the controls described in the following table.

Element	Description
<Previous	Click to return to the Create Instance: Instance Details page.
Cancel	Click to cancel the Create Instance wizard without creating a new deployment.
Create>	Click to begin the process of creating a Database Cloud Service deployment. The Create Instance wizard closes and the Oracle Database Cloud Service console is displayed, showing the new deployment with a status of In progress.

C

The oracle-dbcs-cli Utility

You can use the `oracle-dbcs-cli` utility on your Linux computer to connect to Oracle Cloud and perform a variety of life-cycle and administration operations on Oracle Database Cloud Service deployments.

> **Note:**
>
> The `oracle-dbcs-cli` utility does not currently support database deployments that use Oracle Real Application Clusters.

Topics

Downloading and Installing the oracle-dbcs-cli Utility

Before you can use the `oracle-dbcs-cli` utility, you must download it to your Linux computer and install it.

1. On your Linux computer, create a directory to contain the installed `oracle-dbcs-cli` utility.
2. Download the installation zip file to the directory you created.

   The installation zip file is available at the Oracle Technology Network on the Public Database Cloud Downloads page.
3. In a terminal window, navigate to the directory you created and where you downloaded the installation zip file.
4. Expand the installation zip file:

   ```
   $ tar -vxzf ./oracle-dbcs-cli.tar.gz
   ```

> **Note:**
>
> For the `oracle-dbcs-cli` utility to operate correctly, make sure:
>
> - You are using Java 1.7 on your computer. Enter `java -version` in a terminal window to discover what version you are using.
> - You run the `oracle-dbcs-cli` utility from the installation directory, or set your `PATH` environment variable to include the `dborch.jar` file in the installation directory.

Running the oracle-dbcs-cli Utility

The `oracle-dbcs-cli` utility provides a number of subcommands to perform various life-cycle and administration operations on Oracle Database Cloud Service database deployments.

To run the `oracle-dbcs-cli` utility, first navigate to the directory where you installed the utility. Then, run a command of the form:

```
./oracle-dbcs-cli -help | -ver | subcommand subcommmand-options
```

Use the `-help` option to display help information for the `oracle-dbcs-cli` utility.

Use the `-ver` option to display version information for the `oracle-dbcs-cli` utility.

Use *`subcommand subcommmand-options`* to perform one of the life-cycle or administration operations that for the `oracle-dbcs-cli` utility supports:

- create: create a new deployment.
- delete: delete a deployment.
- list: list deployments in an identity domain or list detailed information for one deployment.
- patch apply: apply a patch to a deployment .
- patch check: check whether a deployment meets the prerequisites of a patch.
- patch rollback: roll back a patch on a deployment.
- patch status: display the status of a patch's application to a deployment.
- patch list: list the patches available for a deployment.
- scaleup: scale up the Compute shape of a deployment's compute node.

The create Subcommand

The `create` subcommand creates a new Database Cloud Service deployment.

```
oracle-dbcs-cli create -dat data-file
```

Options of this subcommand are as follows.

Option	Description
`-dat` *`data-file`*	The name of the data file containing information to connect to Oracle Cloud and information describing how to configure the new deployment. For information about the content of the data file, see The Data File for the oracle-dbcs-cli create Subcommand.

The delete Subcommand

The `delete` subcommand deletes an existing Database Cloud Service deployment.

```
oracle-dbcs-cli delete -cfg config-file -vmname node-name
```

Options of this subcommand are as follows.

Option	Description
`-cfg` *`config-file`*	The name of the configuration file containing information to connect to Oracle Cloud. For information about the content of the configuration file, see The Configuration File for oracle-dbcs-cli Subcommands.
`-vmname` *`node-name`*	The name of the deployment's compute node virtual machine to delete.

The list Subcommand

The `list` subcommand displays a list of deployments in an identity domain or displays detailed information for one deployment, depending on whether you specify the `-vmname` option.

```
oracle-dbcs-cli list -cfg config-file [-vmname node-name]
```

Options of this subcommand are as follows.

Option	Description
`-cfg` *`config-file`*	The name of the configuration file containing information to connect to Oracle Cloud. For information about the content of the configuration file, see The Configuration File for oracle-dbcs-cli Subcommands.
`-vmname` *`node-name`*	The name of the deployment's compute node virtual machine about which to display detailed information. When omitted, a list of deployments in the identity domain is displayed.

The patch apply Subcommand

The `patch apply` subcommand applies a patch to the virtual machine of a Database Cloud Service deployment.

```
oracle-dbcs-cli patch apply -key ssh-private-key-file -host node-ip-address -id
patch-id [-secret ssh-passphrase] [-force]
```

Options of this subcommand are as follows.

Option	Description
`-key` *`ssh-private-key-file`*	The name of the file containing the SSH private key to connect to the deployment's virtual machine.
`-host` *`node-ip-address`*	The public IP address of the compute node associated with the deployment.
`-id` *`patch-id`*	The patch id of the patch to apply. To determine the patch id, use the `oracle-dbcs-cli patch list` command.
`-secret` *`ssh-passphrase`*	The passphrase of the SSH private key, if it was created with a passphrase. If the passphrase includes spaces, enclose it in single quotes; for example: `-secret 'This is a passphrase'`
`-force`	Forces the patching operation to ignore any errors that occur.

The patch check Subcommand

The `patch check` subcommand checks whether the virtual machine of a Database Cloud Service deployment meets the prerequisites of a specified patch.

`oracle-dbcs-cli patch check -key` *`ssh-private-key-file`* `-host` *`node-ip-address`* `-id` *`patch-id`* `[-secret` *`ssh-passphrase`*`] [-force]`

Options of this subcommand are as follows.

Option	Description
`-key` *`ssh-private-key-file`*	The name of the file containing the SSH private key to connect to the deployment's virtual machine.
`-host` *`node-ip-address`*	The public IP address of the compute node associated with the deployment.
`-id` *`patch-id`*	The patch id of the patch to check. To determine the patch id, use the `oracle-dbcs-cli patch list` command.
`-secret` *`ssh-passphrase`*	The passphrase of the SSH private key, if it was created with a passphrase. If the passphrase includes spaces, enclose it in single quotes; for example: `-secret 'This is a passphrase'`
`-force`	Forces the patch-checking operation to ignore any errors that occur.

The patch rollback Subcommand

The `patch rollback` subcommand rolls back a patch that was successfully or unsuccessfully applied to the virtual machine of a Database Cloud Service deployment.

`oracle-dbcs-cli patch rollback -key` *`ssh-private-key-file`* `-host` *`node-ip-address`* `-id` *`patch-id`* `[-secret` *`ssh-passphrase`*`]`

Options of this subcommand are as follows.

Option	Description
`-key` *`ssh-private-key-file`*	The name of the file containing the SSH private key to connect to the deployment's virtual machine.
`-host` *`node-ip-address`*	The public IP address of the compute node associated with the deployment.
`-id` *`patch-id`*	The patch id of the patch to roll back. To determine the patch id, use the `oracle-dbcs-cli patch list` command.
`-secret` *`ssh-passphrase`*	The passphrase of the SSH private key, if it was created with a passphrase. If the passphrase includes spaces, enclose it in single quotes; for example: `-secret 'This is a passphrase'`

The patch status Subcommand

The `patch status` subcommand display the status of a patching operation on the virtual machine of a Database Cloud Service deployment.

`oracle-dbcs-cli patch status -key` *`ssh-private-key-file`* `-host` *`node-ip-address`* `-id` *`transaction-id`* `[-secret` *`ssh-passphrase`*`]`

Options of this subcommand are as follows.

Option	Description
`-key` *`ssh-private-key-file`*	The name of the file containing the SSH private key to connect to the deployment's virtual machine.
`-host` *`node-ip-address`*	The public IP address of the compute node associated with the deployment.
`-id` *`transaction-id`*	The transaction id of the patching operation. To determine the transaction id, use the `oracle-dbcs-cli patch list` command.
`-secret` *`ssh-passphrase`*	The passphrase of the SSH private key, if it was created with a passphrase. If the passphrase includes spaces, enclose it in single quotes; for example: `-secret 'This is a passphrase'`

The patch list Subcommand

The `patch list` subcommand displays a list of the patches available for the virtual machine of a Database Cloud Service deployment.

`oracle-dbcs-cli patch list -key` *`ssh-private-key-file`* `-host` *`node-ip-address`* `[-secret` *`ssh-passphrase`*`]`

Options of this subcommand are as follows.

Option	Description
`-key` *`ssh-private-key-file`*	The name of the file containing the SSH private key to connect to the deployment's virtual machine.

Option	Description
`-host node-ip-address`	The public IP address of the compute node associated with the deployment.
`-secret ssh-passphrase`	The passphrase of the SSH private key, if it was created with a passphrase. If the passphrase includes spaces, enclose it in single quotes; for example: `-secret 'This is a passphrase'`

The scaleup Subcommand

The `scaleup` subcommand scales up the Compute shape of the compute node associated with a Database Cloud Service deployment.

```
oracle-dbcs-cli scaleup -cfg config-file -vmname node-name -shape shape-name
```

Options of this subcommand are as follows.

Option	Description
`-cfg config-file`	The name of the configuration file containing information to connect to Oracle Cloud. For information about the content of the configuration file, see The Configuration File for oracle-dbcs-cli Subcommands.
`-vmname node-name`	The name of the deployment's compute node virtual machine.
`-shape shape-name`	The shape to scale the virtual machine up to. Specify one of these values for *shape-name*: • oc3 — 1 OCPU with 7.5 GB RAM • oc4 — 2 OCPUs with 15 GB RAM • oc5 — 4 OCPUs with 30 GB RAM • oc6 — 8 OCPUs with 60 GB RAM • oc7 — 16 OCPUs with 120 GB RAM • oc1m — 1 OCPU with 15 GB RAM • oc2m — 2 OCPUs with 30 GB RAM • oc3m — 4 OCPUs with 60 GB RAM • oc4m — 8 OCPUs with 120 GB RAM • oc5m — 16 OCPUs with 240 GB RAM

The Configuration File for oracle-dbcs-cli Subcommands

Many of the subcommands that the `oracle-dbcs-cli` utility supports require a configuration file that specifies information to connect to the correct Oracle Database Cloud Service identity domain.

Contents of the Configuration File

The configuration file you specify in many `oracle-dbcs-cli` subcommands is a text file containing lines of the format:

```
name=value
```

where *name* is the predefined name for a piece of data and *value* is the value of that piece of data in your context. The following table shows the names and values required in the configuration file.

Name	Description
`sm_url`	The url to the Database Cloud Service REST interface: `https://dbaas.oraclecloud.com/paas/service/dbcs/api/v1.1/instances`
`identity_domain`	Name of the identity domain to connect to.
`user_name`	User name of an Oracle Cloud user authorized as an administrator in the given identity domain.
`password`	Password of the given Oracle Cloud user.

Example of a Configuration File

Here is an example of a configuration file for use with `oracle-dbcs-cli` subcommands.

```
sm_url=https://dbaas.oraclecloud.com/paas/service/dbcs/api/v1.1/instances
identity_domain=usoracle99999
user_name=dbcsadmin
password=Pa55_WoRd
```

The Data File for the oracle-dbcs-cli create Subcommand

The `create` subcommand of the `oracle-dbcs-cli` utility requires a data file that specifies information to connect to the correct Oracle Database Cloud Service identity domain and information about how to configure the new database deployment.

Template Data File

Included in the download for the `oracle-dbcs-cli` utility is a template data file you can use as a starting point for creating a data file of your own. This template data file is named `dboplan.dat.external.tmpl` and is located in the same directory as the `oracle-dbcs-cli` utility.

Contents of the Data File

The data file you specify in the `oracle-dbcs-cli create` subcommand is a text file containing lines of the format:

name=*value*

where *name* is the predefined name for a piece of data and *value* is the value of that piece of data in your context. The following table shows the names and values required in the configuration file.

Name	Description
`sm_url`	The url to the Database Cloud Service REST interface. Enter this value: `https://dbaas.oraclecloud.com/paas/service/dbcs/api/v1.1/instances` Value in template data file: Blank
`user_name`	User name of an Oracle Cloud user authorized as an administrator in the given identity domain. Value in template data file: Blank
`password`	Password of the given Oracle Cloud user. Value in template data file: Blank
`identity_domain`	Name of the identity domain to connect to. Value in template data file: Blank
`subscriptionType`	The billing period for the deployment. Enter `monthly` or `hourly`. Value in template data file: Blank
`vm_name`	The name for the deployment. The service name: • Must not exceed 50 characters. • Must start with a letter. • Must contain only letters, numbers, or hyphens. • Must not contain any other special characters. • Must be unique within the identity domain. Value in template data file: Blank
`vm_shape`	The Oracle Compute Cloud Service shape of the compute node virtual machine. Enter one of the following: • `oc3` — 1 OCPU with 7.5 GB RAM • `oc4` — 2 OCPUs with 15 GB RAM • `oc5` — 4 OCPUs with 30 GB RAM • `oc6` — 8 OCPUs with 60 GB RAM • `oc7` — 16 OCPUs with 120 GB RAM • `oc1m` — 1 OCPU with 15 GB RAM • `oc2m` — 2 OCPUs with 30 GB RAM • `oc3m` — 4 OCPUs with 60 GB RAM • `oc4m` — 8 OCPUs with 120 GB RAM • `oc5m` — 16 OCPUs with 240 GB RAM Value in template data file: `ot1` (an invalid value)
`vm_seclist`	The name of the Oracle Compute Cloud Service security list to be created for the new deployment. Leave this value blank to use a default name. Value in template data file: Blank

Name	Description
vm_dbsecrules	A comma-separated list of entries that describe the Oracle Compute Cloud Service security rules and security applications to be created for the new deployment. Each entry has the form: *ip-list*:*application*:*port*:*status* For each entry, an Oracle Compute Cloud Service security application is created using the *application* and *port*, and an Oracle Compute Cloud Service security rule is created, linking the provided *ip-list* to the deployment's security list (specified by the vm_seclist value). The security rule is set to the provided *status*. The components of an entry's form are as follows: • *ip-list* specifies the source group of the security rule to be created. Its value must be public or site, which correspond respectively to the **public-internet** and **site** predefined Oracle Compute Cloud Service security IP lists. • *application* specifies software on the deployment that will service incoming requests: – dbconsole — Enterprise Manager 11g Database Control. – dbexpress — Enterprise Manager Database Express 12c. – gfish — web server providing HTTP access. – listener — The Oracle Net Listener. – ssh — The SSH daemon. • *port* specifies the port of the security application to be created. Default ports for the security applications are as follows: – 22 for ssh – 80 for gfish – 1158 for dbconsole – 1521 for listener – 5500 for dbexpress • *status* specifies whether the security rule is to be created as enabled or disabled. Its value must be enabled or disabled. Value in template data file: public:ssh:22:enabled, public:listener:1521:disabled, public:gfish:80:disabled, public:dbconsole:1158:disabled, public:dbexpress: 5500:disabled
vm_nat	The NAT IP pool from which the new compute node public IP address is drawn. Value in template data file: Blank
vm_ha	Currently unsupported. Use the value "monitor". Value in template data file: "monitor"

Name	Description
`vm_boot`	Type of boot volume to create for the compute node. Use the value `nds`. Value in template data file: `nds`
`vm_boot_size`	Size in GB of the boot volume to create for the compute node. Value in template data file: `21gb`
`vm_sshkeys`	A string containing the text of an SSH public key. This key is added to Oracle Compute Cloud Service and associated with the deployment as part of the creation operation. Value in template data file: Blank
`db_version`	Version of Oracle Database to install on the compute node. Enter `12102` or `11204`. Value in template data file: `12102`
`db_edition`	Edition of Oracle Database to install on the compute node. Enter `enterprise` or `standard`. If you specify `standard`, a Standard Edition 2 database is created if you specify `12102` for `db_version` and a Standard Edition database is created if you specify `11204` for `db_version`. Value in template data file: `enterprise`
`db_bundle`	Level of Oracle Database Enterprise Edition to install on the compute node; valid on if you entered `enterprise` for `db_edition`. Enter `"basic"` or `"high-perf"`. Value in template data file: `"high-perf"`
`db_lvm`	Controls whether the database storage uses Linux LVM (logical volume manager). Enter `yes` or `no`. Value in template data file: `no`
`db_nat_ip`	The fully qualified name of an existing Oracle Compute Cloud Service IP reservation to use as the public IP address for the service. Leave this value blank to have Database Cloud Service create an IP reservation as part of the creation process. The name you provide must have this form: `/Compute-`*`domain`*`/user/reservation` where *`domain`* name the identity domain, *`user`* is the name of the user who created the IP reservation, and *`user`* is the name of the IP reservation; for example: `/Compute-usoracle99999/dbaasadmin/custom-ip` Value in template data file: Blank

Name	Description
db_timezone	The time zone to use when configuring the operating system. Enter one of the following values: Africa/Cairo, Africa/Casablanca, Africa/Harare, Africa/Monrovia, Africa/Nairobi, Africa/Tripoli, Africa/Windhoek, America/Araguaina, America/Asuncion, America/Bogota, America/Caracas, America/Chihuahua, America/Cuiaba, America/Denver, America/Fortaleza, America/Guatemala, America/Halifax, America/Manaus, America/Matamoros, America/Monterrey, America/Montevideo, America/Phoenix, America/Santiago, America/Tijuana, Asia/Amman, Asia/Ashgabat, Asia/Baghdad, Asia/Baku, Asia/Bangkok, Asia/Beirut, Asia/Calcutta, Asia/Damascus, Asia/Dhaka, Asia/Irkutsk, Asia/Jerusalem, Asia/Kabul, Asia/Karachi, Asia/Kathmandu, Asia/Krasnoyarsk, Asia/Magadan, Asia/Muscat, Asia/Novosibirsk, Asia/Riyadh, Asia/Seoul, Asia/Shanghai, Asia/Singapore, Asia/Taipei, Asia/Tehran, Asia/Tokyo, Asia/Ulaanbaatar, Asia/Vladivostok, Asia/Yakutsk, Asia/Yerevan, Atlantic/Azores, Australia/Adelaide, Australia/Brisbane, Australia/Darwin, Australia/Hobart, Australia/Perth, Australia/Sydney, Brazil/East, Canada/Newfoundland, Canada/Saskatchewan, Europe/Amsterdam, Europe/Athens, Europe/Dublin, Europe/Helsinki, Europe/Istanbul, Europe/Kaliningrad, Europe/Moscow, Europe/Paris, Europe/Prague, Europe/Sarajevo, Pacific/Auckland, Pacific/Fiji, Pacific/Guam, Pacific/Honolulu, Pacific/Samoa, US/Alaska, US/Central, US/Eastern, US/East-Indiana, US/Pacific, UTC Value in template data file: UTC

Name	Description
`charset`	The database character set for the database. Enter one of the following values: `AL32UTF8`, `AR8ADOS710`, `AR8ADOS720`, `AR8APTEC715`, `AR8ARABICMACS`, `AR8ASMO8X`, `AR8ISO8859P6`, `AR8MSWIN1256`, `AR8MUSSAD768`, `AR8NAFITHA711`, `AR8NAFITHA721`, `AR8SAKHR706`, `AR8SAKHR707`, `AZ8ISO8859P9E`, `BG8MSWIN`, `BG8PC437S`, `BLT8CP921`, `BLT8ISO8859P13`, `BLT8MSWIN1257`, `BLT8PC775`, `BN8BSCII`, `CDN8PC863`, `CEL8ISO8859P14`, `CL8ISO8859P5`, `CL8ISOIR111`, `CL8KOI8R`, `CL8KOI8U`, `CL8MACCYRILLICS`, `CL8MSWIN1251`, `EE8ISO8859P2`, `EE8MACCES`, `EE8MACCROATIANS`, `EE8MSWIN1250`, `EE8PC852`, `EL8DEC`, `EL8ISO8859P7`, `EL8MACGREEKS`, `EL8MSWIN1253`, `EL8PC437S`, `EL8PC851`, `EL8PC869`, `ET8MSWIN923`, `HU8ABMOD`, `HU8CWI2`, `IN8ISCII`, `IS8PC861`, `IW8ISO8859P8`, `IW8MACHEBREWS`, `IW8MSWIN1255`, `IW8PC1507`, `JA16EUC`, `JA16EUCTILDE`, `JA16SJIS`, `JA16SJISTILDE`, `JA16VMS`, `KO16KSC5601`, `KO16KSCCS`, `KO16MSWIN949`, `LA8ISO6937`, `LA8PASSPORT`, `LT8MSWIN921`, `LT8PC772`, `LT8PC774`, `LV8PC1117`, `LV8PC8LR`, `LV8RST104090`, `N8PC865`, `NE8ISO8859P10`, `NEE8ISO8859P4`, `RU8BESTA`, `RU8PC855`, `RU8PC866`, `SE8ISO8859P3`, `TH8MACTHAIS`, `TH8TISASCII`, `TR8DEC`, `TR8MACTURKISHS`, `TR8MSWIN1254`, `TR8PC857`, `US7ASCII`, `US8PC437`, `UTF8`, `VN8MSWIN1258`, `VN8VN3`, `WE8DEC`, `WE8DG`, `WE8ISO8859P1`, `WE8ISO8859P15`, `WE8ISO8859P9`, `WE8MACROMAN8S`, `WE8MSWIN1252`, `WE8NCR4970`, `WE8NEXTSTEP`, `WE8PC850`, `WE8PC858`, `WE8PC860`, `WE8ROMAN8`, `ZHS16CGB231280`, `ZHS16GBK`, `ZHT16BIG5`, `ZHT16CCDC`, `ZHT16DBT`, `ZHT16HKSCS`, `ZHT16MSWIN950`, `ZHT32EUC`, `ZHT32SOPS`, `ZHT32TRIS` Value in template data file: `AL32UTF8`
`ncharset`	The national character set for the database. Enter one of the following values: `AL16UTF16`, `UTF8` Value in template data file: `AL16UTF16`
`db_vols`	A space-separated, parenthesized list of the sizes of the four Oracle Compute Cloud Service storage volumes in addition to the boot volume to be created for the compute node. Each of the four entries in the list has the form: *`name`*`:`*`size`* where: • *`name`* is the name of one of the four volumes: `bits`, `data`, `fra` or `redo`. • *`size`* is the size in GB for the named volume; for example, `25gb`. Value in template data file: `(bits:30gb data:10gb fra:10gb redo:10gb)`

Name	Description
db_redo_log_size	The size in MB for each of the three redo logs created in the database. Use the abbreviation M instead of MB or mb; for example, 150M. Value in template data file: 100M
db_sid	The name for the database. The name your enter: • Must not exceed 8 characters. • Must start with a letter. • Must contain only letters, numbers, or these symbols: _ (underscore), # (hash sign), or $ (dollar sign). Value in template data file: orcl
db_passwd	The password for the following administrative users: • Oracle Database administrative users SYS and SYSTEM • Oracle Application Express ADMIN user • Oracle DBaaS Monitor dbaas_monitor user The password you enter: • Must be 8 to 30 characters in length. • Must contain at least one lowercase letter • Must contain at least one uppercase letter • Must contain at least one number • Must contain at least one of these symbols: _ (underscore), # (hash sign), or $ (dollar sign). Value in template data file: Blank
db_automem	Controls whether the database created on the deployment is configured for automatic memory management. Enter yes or no. Value in template data file: yes
db_cdb	Value in template data file: yes
db_pdb_name	(Applicable only for Oracle Database 12c Release 1.) The name for the default PDB (Pluggable Database). The name you enter: • Must not exceed 8 characters. • Must start with a letter. • Must contain only letters, numbers, or these symbols: _ (underscore), # (hash sign), or $ (dollar sign). Value in template data file: pdb1
db_em	Controls whether the Enterprise Manager tool (Enterprise Manager Database Express 12c for Oracle Database 12c or Enterprise Manager 11g Database Control for Oracle Database 11g) is configured. Enter yes or no. Value in template data file: yes

Name	Description
db_archlog	Controls whether archive logs are enabled. Enter yes or no. Value in template data file: yes **Note:** If backups are configured (db_bkup_disk is set to yes), archive logs are enabled regardless of the value you specify for db_archlog.
db_flashback	Controls whether flashback logs are enabled. Enter yes or no. Value in template data file: yes
db_flashback_days	The minimum time in days to retain flashback logs in the recovery area. Value in template data file: "1"
db_bkup_disk	Controls whether backups to local storage on the compute node are configured. Enter yes or no. When taken together, the values of db_bkup_disk and db_bkup_oss determine the backup destination: • db_bkup_disk=**yes** and db_bkup_oss=**yes** — the Both Cloud Storage and Local Storage destination • db_bkup_disk=**no** and db_bkup_oss=**yes** — the Cloud Storage Only destination • db_bkup_disk=**no** and db_bkup_oss=**no** — the None destination Value in template data file: yes
db_bkup_cron_entry	Controls whether an entry is added to the /etc/crontab file to enable daily backups. Enter yes or no. Value in template data file: yes
db_bkup_daily_time	Specifies the time (using 24-hour, HH:MM format) when daily backups are to occur. For example, 02:45 is 2:45 AM, and 14:45 is 2:45 PM. Leave this value blank to have Database Cloud Service pick a random time from 11 PM (23:00) to 3 AM (03:00). Value in template data file: Blank
db_bkup_disk_recovery_window	The number of days for which backups and archived redo logs on local storage are maintained. The interval always ends with the current time and extends back in time for the number of days specified. Value in template data file: 7

Name	Description
db_bkup_oss	Controls whether backups to an Oracle Cloud Infrastructure Object Storage Classic container using Oracle Database Backup Cloud Service are to be configured. Enter yes or no. When taken together, the values of db_bkup_disk and db_bkup_oss determine the backup destination: • db_bkup_disk=**yes** and db_bkup_oss=**yes** — the Both Cloud Storage and Local Storage destination • db_bkup_disk=**no** and db_bkup_oss=**yes** — the Cloud Storage Only destination • db_bkup_disk=**no** and db_bkup_oss=**no** — the None destination Value in template data file: no
db_bkup_oss_url	The REST endpoint of the Oracle Cloud Infrastructure Object Storage Classic container to use for backups. For information on the value to provide, see See "About REST URLs for Oracle Storage Cloud Service Resources" in *Using Oracle Cloud Infrastructure Object Storage Classic*. Value in template data file: Blank
db_bkup_oss_user	The user name of an Oracle Cloud user who has read/write access to the container specified in db_bkup_oss_url. Value in template data file: Blank
db_bkup_oss_passwd	The password of the user specified in db_bkup_oss_user. Value in template data file: Blank
db_bkup_oss_recovery_window	The number of days for which backups on cloud storage are maintained. The interval always ends with the current time and extends back in time for the number of days specified. Value in template data file: 14
db_bkup_cfg_files	Controls whether backups are to include the files listed in /home/oracle/bkup/dbcfg.spec and /home/oracle/bkup/oscfg.spec. Enter yes or no. Value in template data file: yes
db_bkup_cfg_recovery_window	The number of days for which backups of configuration files are maintained. The interval always ends with the current time and extends back in time for the number of days specified. Value in template data file: Blank
db_tde_action	Controls whether TDE is configured. Enter config or none. Value in template data file: none

Name	Description
db_tde_ks_login	Controls how the database is to access keys in the keystore wallet. Enter one of these values: • manual—Every time the database is started the administrator must log into the database and open the keystore wallet manually. • local—Every time the database is started the Local Auto Login TDE feature is used to open the keystore wallet automatically. • auto—Every time the database is started the Auto Login TDE feature is used to open the keystore wallet automatically. Value in template data file: auto
db_net_security_enable	Controls whether Oracle Net Services data encryption and integrity are configured. Enter yes or no. Value in template data file: no
db_net_security_encryption_enable	Controls whether Oracle Net Services data encryption is configured. Enter yes or no. Value in template data file: yes
db_net_security_encryption_target	Specifies whether the Oracle Net Services data encryption configuration is for the server or the client. Use the value server. Value in template data file: server

Name	Description
db_net_security_encryption_type	Specifies how Oracle Net Services data encryption is negotiated with clients. Enter one of these values: • rejected—Enter this value if you do not elect to enable data encryption, even if required by the client. In this scenario, this side of the connection specifies that data encryption is not permitted. If the client side is set to required, the connection terminates with error message ORA-12650. If the client side is set to requested, accepted or rejected, the connection continues without error and without data encryption enabled. • accepted—Select this value to enable data encryption if required or requested by the client. In this scenario, this side of the connection does not require data encryption, but it is enabled if the client side is set to required or requested. If the client side is set to required or requested, and an encryption algorithm match is found, the connection continues without error and with data encryption enabled. If the client side is set to required and no algorithm match is found, the connection terminates with error message ORA-12650. If the client side is set to requested and no algorithm match is found, or if the client side is set to accepted or rejected, the connection continues without error and without data encryption enabled. • requested—Select this value to enable data encryption if the client permits it. In this scenario, this side of the connection specifies that data encryption is desired but not required. Data encryption is enabled if the client side specifies accepted, requested, or required. There must be a matching algorithm available, otherwise data encryption is not enabled. If the client side specifies required and there is no matching algorithm, the connection fails. • required—Select this value to enable data encryption or preclude the connection. In this scenario, this side of the connection specifies that data encryption must be enabled. The connection fails if the client side specifies rejected or if there is no compatible algorithm. Value in template data file: required
db_net_security_encryption_methods	Specifies a list of algorithms that can be used for data encryption. Separate the algorithms with commas and do not include spaces. Here is a list of valid encryption algorithms: AES128 AES192 AES256 Value in template data file: AES256,AES192,AES128
db_net_security_integrity_enable	Controls whether Oracle Net Services data integrity is configured. Enter yes or no. Value in template data file: yes

Name	Description
db_net_security_integrity_target	Specifies whether the Oracle Net Services data integrity configuration is for the server or the client. Use the value server. Value in template data file: server
db_net_security_integrity_checksum_level	Specifies how Oracle Net Services data integrity is negotiated with clients. Enter one of these values: • rejected—Enter this value if you do not elect to enable data integrity, even if required by the client. In this scenario, this side of the connection specifies that data integrity is not permitted. If the client side is set to required, the connection terminates with error message ORA-12650. If the client side is set to requested, accepted or rejected, the connection continues without error and without data integrity enabled. • accepted—Select this value to enable data integrity if required or requested by the client. In this scenario, this side of the connection does not require data integrity, but it is enabled if the client side is set to required or requested. If the client side is set to required or requested, and an integrity algorithm match is found, the connection continues without error and with data integrity enabled. If the client side is set to required and no algorithm match is found, the connection terminates with error message ORA-12650. If the client side is set to requested and no algorithm match is found, or if the client side is set to accepted or rejected, the connection continues without error and without data integrity enabled. • requested—Select this value to enable data integrity if the client permits it. In this scenario, this side of the connection specifies that data integrity is desired but not required. Data integrity is enabled if the client side specifies accepted, requested, or required. There must be a matching algorithm available, otherwise data integrity is not enabled. If the client side specifies required and there is no matching algorithm, the connection fails. • required—Select this value to enable data integrity or preclude the connection. In this scenario, this side of the connection specifies that data integrity must be enabled. The connection fails if the client side specifies rejected or if there is no compatible algorithm. Value in template data file: required
db_net_security_integrity_methods	Specifies a list of algorithms that can be used for data integrity. Separate the algorithms with commas and do not include spaces. Here is a list of valid integrity algorithms: SHA1 SHA256 SHA384 SHA512 Note that of these four algorithms, SHA1 is the only one supported by Oracle Database 11g. Value in template data file: SHA1

D

The dbaascli Utility

You can use the `dbaascli` utility to perform a variety of life-cycle and administration operations on Oracle Database Cloud Service database deployments.

Using the `dbaascli` utility, you can:

- Change the password of a database user.
- Start and stop a database.
- Start and stop the Oracle Net listener
- Check the status of the Oracle Data Guard configuration.
- Perform switchover and failover in an Oracle Data Guard configuration.
- Patch the database deployment.
- Perform database recovery.
- Rotate the master encryption key.

To use the `dbaascli` utility:

1. Connect to a compute node associated with the Database Cloud Service deployment.

   Commands using the `patch, dbpatchm` or `orec` subcommands must be run with `root` administrator privileges. In this case, first connect as the `opc` user and then start a root-user command shell by executing the `sudo –s` command.

   Otherwise, connect as the `oracle` user.

   For instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Run the `dbaascli` utility using a command of the form:

   ```
   # dbaascli subcommand subcommand-options
   ```

The `dbaascli` utility supports these subcommands:

Subcommand	Subcommand Options
`database`	`bounce` – shuts down and then restarts the database instance. `changepassword` – changes the password of the specified user. `start`– starts the database instance and opens the database. `status` – displays the open mode of the database and additional information about the database deployment. `stop` – shuts down the database instance.
`dataguard`	`failover` – performs a manual failover. `reinstate` – reinstates a failed primary database. `status` – checks the status of the configuration. `switchover` – performs a switchover operation.

Subcommand	Subcommand Options
dbpatchm	apply – applies the patch. clonedb – applies a patch to a test deployment. list_patches – displays a list of available patches. list_tools – checks whether any cloud tooling updates are available. prereq – checks the prerequisites of a patch. rollback – rolls back the last deployment patch. switchback – restores database software to a prior state. toolsinst – downloads and applies the patch containing the cloud tooling update.
dv	off – disables Oracle Database Vault. on– enables Oracle Database Vault.
gg	setup – configures the database as a valid replication database. status – indicates whether the database has been configured as a valid replication database.
listener	bounce – stops and restarts the listener. start – starts the listener. status – displays the status of the listener, including a summary of listener configuration settings, listening protocol addresses, and a summary of services registered with the listener. stop – stops the listener.
netsec	config – configures network encryption and network integrity. config encryption– configures network encryption. config integrity – configures network integrity. status – displays network encryption and network integrity configuration information.
orec	duplicate – updates the standby database in a Data Guard configuration. keep list – lists the available long-term backups. keep tag – restores a specific long-term backup and performs recovery. latest – restores the most recent backup and performs complete recovery. list – lists the available normal backups. pitr – restores a specific normal backup and performs recovery. scn – restores the most recent backup and performs recovery through the specified SCN.

Subcommand	Subcommand Options
patch	db apply – applies the database patch. db cleanup – removes temporary files created during database patching operations. db list – displays a list of available database patches. db prereq – checks whether any database updates are available. db switchback – rolls back the last database patch. os apply – applies the OS patch. os list – displays a list of available OS patches. tools apply – downloads and applies the patch containing the cloud tooling update. tools auto disable – disables automatic cloud tooling updates. tools auto enable – enables automatic cloud tooling updates. tools auto execute – downloads and applies the patch containing the latest cloud tooling update. tools auto status – checks whether automatic cloud tooling updates are enabled or disabled. tools list – checks whether any cloud tooling updates are available.
tde	rotate masterkey – changes (rotates) the master encryption key. status – displays information about the software keystore, including the type and status.

dbaascli database bounce

The `database bounce` subcommand of the `dbaascli` utility can be used to shut down and restart the database.

Execute this command as the `oracle` user.

```
dbaascli database bounce
```

When this subcommand is executed the database is shut down in immediate mode. The database instance is then restarted and the database is opened. In Oracle Database 12c or later, all PDBs are opened.

dbaascli database changepassword

The `database changepassword` subcommand of the `dbaascli` utility is used to change the password of a database user.

Execute this command as the `oracle` user.

```
dbaascli database changepassword
```

Enter the user name and new password when prompted.

dbaascli database start

The `database start` subcommand of the `dbaascli` utility can be used to start the database instance and open the database.

Execute this command as the `oracle` user.

```
dbaascli database start
```

When this subcommand is executed the database instance is started and the database is opened. In Oracle Database 12c or later, all PDBs are opened.

dbaascli database status

The `database status` subcommand of the `dbaascli` utility can be used to check the status of the database in your database deployment.

Execute this command as the `oracle` user.

```
dbaascli database status
```

Output from the command includes the open mode of the database, the software release and edition of the database deployment, and release version of other software components.

dbaascli database stop

The `database stop` subcommand of the `dbaascli` utility can be used to shut down the database.

Execute this command as the `oracle` user.

```
dbaascli database stop
```

When this subcommand is executed the database is shut down in immediate mode. No new connections or new transactions are permitted. Active transactions are rolled back and all connected users are disconnected.

dbaascli dataguard failover

The `dataguard failover` subcommand of the `dbaascli` utility is used to perform a manual failover to the standby database in your Oracle Data Guard configuration.

Execute this command as the `oracle` user.

```
dbaascli dataguard failover [--force yes|no] [--password password]
```

Options of this subcommand are as follows.

Option	Description
`--force yes|no`	The `force` option can be used if the `dataguard status` subcommand shows that the Data Guard configuration is in a warning or error state.

Option	Description
`--password` *password*	The `password` option is used to supply the SYS user password if it has changed since the Data Guard configuration was created.

dbaascli dataguard reinstate

The `dataguard reinstate` subcommand of the `dbaascli` utility is used to reinstate a failed primary database as a standby database after a failover.

Execute this command as the `oracle` user.

```
dbaascli dataguard reinstate [--password password]
```

Options of this subcommand are as follows.

Option	Description
`--password` *password*	The `password` option is used to supply the SYS user password if it has changed since the Data Guard configuration was created.

dbaascli dataguard status

The `dataguard status` subcommand of the `dbaascli` utility is used to check the status of the Oracle Data Guard configuration.

Execute this command as the `oracle` user.

```
dbaascli dataguard status [--details yes|no]
```

Options of this subcommand are as follows.

Option	Description
`--details yes|no`	The `details` option is used to request a more detailed listing of the status of the Data Guard configuration.

dbaascli dataguard switchover

The `dataguard switchover` subcommand of the `dbaascli` utility is used to perform a switchover to the standby database in your Oracle Data Guard configuration.

Execute this command as the `oracle` user.

```
dbaascli dataguard switchover [--password password]
```

Options of this subcommand are as follows.

Option	Description
`–password` *password*	The `password` option is used to supply the SYS user password if it has changed since the Data Guard configuration was created.

dbaascli dbpatchm apply

> **Note:**
>
> This command is deprecated as of release 18.3.2 in July 2018. It is replaced by the dbaascli patch db apply command. You should discontinue using this deprecated command as it will be removed in a future release.

The `dbpatchm apply` subcommand of the `dbaascli` utility is used to apply a patch.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -apply
```

Before executing the `apply` subcommand you must edit the `/var/opt/oracle/patch/dbpatchm.cfg` patching configuration file, setting the keys for the desired patch. For more information about this file and its keys, see The dbpatchm.cfg Configuration File.

The `dbpatchm apply` subcommand displays progress as the patch is applied.

dbaascli dbpatchm clonedb

The `dbpatchm clonedb` subcommand of the `dbaascli` utility is used to apply a patch to a test deployment before you apply it to a live, production database deployment.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -clonedb
```

The `dbpatchm clonedb` subcommand displays progress information as it copies information from the live deployment to the test deployment and then applies the patch to the test deployment.

dbaascli dbpatchm list_patches

> **Note:**
>
> This command is deprecated as of release 18.3.2 in July 2018. It is replaced by the dbaascli patch db list command. You should discontinue using this deprecated command as it will be removed in a future release.

The `dbpatchm list_patches` subcommand of the `dbaascli` utility is used to check whether any patches are available.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -list_patches
```

A patch update is available if the command response includes the "INFO: images available for patching" message. The patch ID will be displayed as part of the command response. This patch ID can be used to download and apply the patch. See dbaascli dbpatchm apply for detail on applying a patch.

dbaascli dbpatchm list_tools

> **Note:**
>
> This command is deprecated as of release 18.3.2 in July 2018. It is replaced by the dbaascli patch tools list command. You should discontinue using this deprecated command as it will be removed in a future release.

The `dbpatchm list_tools` subcommand of the `dbaascli` utility is used to check whether any cloud tooling updates are available.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -list_tools
```

A cloud tooling update is available if the command response includes the "INFO: tools images available for patching" message. The patch ID of the cloud tooling update will be displayed as part of the command response. This patch ID can be used to download and apply the patch.

dbaascli dbpatchm prereq

> **Note:**
>
> This command is deprecated as of release 18.3.2 in July 2018. It is replaced by the dbaascli patch db prereq command. You should discontinue using this deprecated command as it will be removed in a future release.

The `dbpatchm prereq` subcommand of the `dbaascli` utility is used to check the prerequisites of a patch.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -prereq
```

Before executing the `prereq` subcommand you must edit the `/var/opt/oracle/patch/dbpatchm.cfg` patching configuration file, setting the keys for the desired patch. For more information about this file and its keys, see The dbpatchm.cfg Configuration File.

dbaascli dbpatchm rollback

> **Note:**
>
> This command is deprecated as of release 18.3.2 in July 2018. It is replaced by the dbaascli patch db switchback command. You should discontinue using this deprecated command as it will be removed in a future release.

The `dbpatchm rollback` subcommand of the `dbaascli` utility is used to roll back the last patch applied to a database deployment.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -rollback
```

dbaascli dbpatchm switchback

The `dbpatchm switchback` subcommand of the `dbaascli` utility is used to revert back to the state of the database deployment before a patch was applied.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -switchback
```

dbaascli dbpatchm toolsinst

> **Note:**
>
> This command is deprecated as of release 18.3.2 in July 2018. It is replaced by the dbaascli patch tools apply command. You should discontinue using this deprecated command as it will be removed in a future release.

The `dbpatchm toolsinst` subcommand of the `dbaascli` utility is used to download and apply a patch containing the cloud tooling update.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli dbpatchm --run -toolsinst -rpmversion=patch-id|LATEST
```

Options of this subcommand are as follows.

Option	Description
`-rpmversion=patch-id|LATEST`	The `rpmversion` option is used to specify the patch to be downloaded and applied. Specify the ID of the patch, or specify `LATEST` to download and apply the latest available patch.

dbaascli dv off

The `dv off` subcommand of the `dbaascli` utility is used to disable Oracle Database Vault in a database deployment.

Note: this command shuts down and then restarts the database. Therefore, you should make sure all database connections are closed before you use this command.

```
dbaascli dv off [cdb|pdb]
```

The utility prompts the user for the following:

- Database Vault Owner user name
- Database Vault Owner password

If no options are specified, the utility will disable Database Vault on the root container (CDB) and all existing pluggable databases (PDBs) in a database deployment using Oracle Database 12c or later. The utility assumes that Database Vault credentials are the same in the CDB and all of the PDBs.

When you install Oracle Database Vault, it revokes a set of privileges from several Oracle Database-supplied users and roles. Be aware that if you disable Oracle Database Vault, these privileges remain revoked. See "Privileges That Are Revoked from Existing Users and Roles" in *Oracle Database Vault Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2 for additional information.

Options of this subcommand are as follows.

Option	Description
`cdb`	The `cdb` option is used to disable Database Vault on the root container (CDB) only in a database deployment using Oracle Database 12c or later.
`pdb`	The `pdb` option is used to disable Database Vault on a specific PDB in a database deployment using Oracle Database 12c or later. When this option is used, the utility also prompts the user for the PDB name.

dbaascli dv on

The `dv on` subcommand of the `dbaascli` utility is used to configure and enable Oracle Database Vault in a database deployment.

```
dbaascli dv on [cdb|pdb]
```

The utility prompts the user for the following:

- Database Vault Owner user name

- Database Vault Owner password
- Database Vault Account Manager user name
- Database Vault Account Manager password
- PDB name, if the `pdb` option is specified

If no options are specified, the utility will configure and enable Database Vault on the root container (CDB) and all existing PDBs of a database deployment using Oracle Database 12c or later. It will set the same Database Vault credentials in the root container and the PDBs.

If you want to enable Database Vault on PDBs separately, you must first enable Database Vault on the CDB.

In a database deployment using Oracle Database 12c or later, the Database Vault Owner and Account Manager user names must begin with `c##`.

Review "What to Expect After You Enable Oracle Database Vault" in *Oracle Database Vault Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2 to gain an understanding of the impact of enabling and configuring Database Vault.

Options of this subcommand are as follows.

Option	Description
`cdb`	The `cdb` option is used to configure and enable Database Vault on the root container (CDB) only in a database deployment using Oracle Database 12c or later.
`pdb`	The `pdb` option is used to configure and enable Database Vault on a specific PDB in a database deployment using Oracle Database 12c or later. When this option is used, the utility also prompts the user for the PDB name. Database Vault must be enabled on the CDB before you can enable it on any PDBs. Provide the same credentials for the PDB as you provided for the CDB.

dbaascli gg setup

The `gg setup` subcommand of the `dbaascli` utility is used to configure a deployment's database for Oracle GoldenGate Cloud Service replication.

Execute this command as the `oracle` user.

```
dbaascli gg setup
```

Enter the Oracle GoldenGate Cloud Service database user name when prompted for the GoldenGate admin username:

- For Oracle Database 11g, specify **`ggadm`**.
- For Oracle Database 12c or later, specify **`c##ggadm`**.

Enter the password specified during the database deployment creation process when prompted for the GoldenGate admin password.

dbaascli gg status

The `gg status` subcommand of the `dbaascli` utility is used to check whether a deployment's database is configured for Oracle GoldenGate Cloud Service replication.

Execute this command as the `oracle` user.

```
dbaascli gg status
```

A status of `enabled` indicates that the database is a valid replication database. If it has not been configured as a replication database, the status is `disabled`.

dbaascli listener bounce

The `listener bounce` subcommand of the `dbaascli` utility is used to stop and restart the listener.

Execute this command as the `oracle` user.

```
dbaascli listener bounce
```

This command causes the listener to be stopped and then restarted.

dbaascli listener start

The `listener start` subcommand of the `dbaascli` utility is used to start the listener.

Execute this command as the `oracle` user.

```
dbaascli listener start
```

dbaascli listener status

The `listener status` subcommand of the `dbaascli` utility is used to obtain information about the status of the listener.

Execute this command as the `oracle` user.

```
dbaascli listener status
```

Basic status information about the listener, including a summary of listener configuration settings, listening protocol addresses, and a summary of services registered with the listener is displayed.

dbaascli listener stop

The `listener stop` subcommand of the `dbaascli` utility is used to stop the listener.

Execute this command as the `oracle` user.

```
dbaascli listener stop
```

dbaascli netsec config

The `netsec config` subcommand of the `dbaascli` utility is used to configure Oracle Net encryption and integrity settings.

By default, database deployments on Database Cloud Service are configured to enable native Oracle Net encryption and integrity. You can use the `netsec config` subcommand to change Oracle Net encryption and integrity settings. For detailed information on Oracle Net encryption and integrity, see "Configuring Oracle Database Network Encryption and Data Integrity" in *Oracle Database Security Guide* for Release 18, 12.2 or 12.1 or "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients" in *Database Advanced Security Administrator's Guide* for Release 11.2.

If you only want to configure encryption settings, see dbaascli netsec config encryption. If you only to configure integrity settings, see dbaascli netsec config integrity.

Execute this command as the `oracle` user.

```
dbaascli netsec config
  --encryption_methods algorithm[,algorithm]...
  --encryption_target client|server
  --encryption_type accepted|rejected|requested|required
  --integrity_clevel accepted|rejected|requested|required
  --integrity_methods algorithm[,algorithm]...
  --integrity_target client|server
```

Options of this subcommand are as follows.

Option	Description
`encryption_methods algorithm[,algorithm]...`	The `encryption_methods` option is used to specify the encryption algorithm(s). Valid values are: AES128, AES192, and AES256.
`encryption_target client|server`	The `encryption_target` option is used to specify whether the encryption setting applies to the client or server. Use `server`.

Option	Description
`encryption_type accepted|rejected|requested|required`	The `encryption_type` option is used to specify the action to take when negotiating encryption. • `rejected`—Enter this value if you do not elect to enable data encryption, even if required by the client. In this scenario, this side of the connection specifies that data encryption is not permitted. If the client side is set to `required`, the connection terminates with error message `ORA-12650`. If the client side is set to `requested`, `accepted` or `rejected`, the connection continues without error and without data encryption enabled. • `accepted`—Select this value to enable data encryption if required or requested by the client. In this scenario, this side of the connection does not require data encryption, but it is enabled if the client side is set to `required` or `requested`. If the client side is set to `required` or `requested`, and an encryption algorithm match is found, the connection continues without error and with data encryption enabled. If the client side is set to `required` and no algorithm match is found, the connection terminates with error message `ORA-12650`. If the client side is set to `requested` and no algorithm match is found, or if the client side is set to `accepted` or `rejected`, the connection continues without error and without data encryption enabled. • `requested`—Select this value to enable data encryption if the client permits it. In this scenario, this side of the connection specifies that data encryption is desired but not required. Data encryption is enabled if the client side specifies `accepted`, `requested`, or `required`. There must be a matching algorithm available, otherwise data encryption is not enabled. If the client side specifies `required` and there is no matching algorithm, the connection fails. • `required`—Select this value to enable data encryption or preclude the connection. In this scenario, this side of the connection specifies that data encryption must be enabled. The connection fails if the client side specifies `rejected` or if there is no compatible algorithm.

Option	Description
`integrity_clevel accepted|rejected| requested|required`	The `integrity_clevel` option is used to specify the checksum level. • `rejected`—Enter this value if you do not elect to enable data integrity, even if required by the client. In this scenario, this side of the connection specifies that data integrity is not permitted. If the client side is set to `required`, the connection terminates with error message `ORA-12650`. If the client side is set to `requested`, `accepted` or `rejected`, the connection continues without error and without data integrity enabled. • `accepted`—Select this value to enable data integrity if required or requested by the client. In this scenario, this side of the connection does not require data integrity, but it is enabled if the client side is set to `required` or `requested`. If the client side is set to `required` or `requested`, and an integrity algorithm match is found, the connection continues without error and with data integrity enabled. If the client side is set to `required` and no algorithm match is found, the connection terminates with error message `ORA-12650`. If the client side is set to `requested` and no algorithm match is found, or if the client side is set to `accepted` or `rejected`, the connection continues without error and without data integrity enabled. • `requested`—Select this value to enable data integrity if the client permits it. In this scenario, this side of the connection specifies that data integrity is desired but not required. Data integrity is enabled if the client side specifies `accepted`, `requested`, or `required`. There must be a matching algorithm available, otherwise data integrity is not enabled. If the client side specifies `required` and there is no matching algorithm, the connection fails. • `required`—Select this value to enable data integrity or preclude the connection. In this scenario, this side of the connection specifies that data integrity must be enabled. The connection fails if the client side specifies `rejected` or if there is no compatible algorithm.
`integrity_methods` *`algorithm`*`[,`*`algorithm`*`] ...`	The `integrity_methods` option is used to specify the integrity algorithm. Valid values are: `SHA1`, `SHA256`, `SHA384`, and `SHA512`.`SHA1` is the only algorithm supported by Oracle Database 11*g*.
`integrity_target client|server`	The `integrity_target` option is used to specify whether the integrity setting applies to the client or server. Use `server`.

dbaascli netsec config encryption

The `netsec config encryption` subcommand of the `dbaascli` utility is used to configure Oracle Net encryption settings.

By default, database deployments on Database Cloud Service are configured to enable native Oracle Net encryption and integrity. You can use the `netsec config encryption` subcommand to change Oracle Net encryption settings. See "Configuring Oracle Database Network Encryption and Data Integrity" in *Oracle Database Security Guide* for Release 18, 12.2 or 12.1 or "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients" in *Database Advanced Security Administrator's Guide* for Release 11.2 for detailed information on Oracle Net encryption.

Execute this command as the `oracle` user.

```
dbaascli netsec config encryption
  --methods algorithm[,algorithm]...
  --target client|server
  --type accepted|rejected|requested|required
```

Options of this subcommand are as follows.

Option	Description
`methods` *`algorithm`*`[,`*`algorithm`*`]...`	The `methods` option is used to specify the encryption algorithm(s). Valid values are: `AES128`, `AES192`, and `AES256`.
`target client|server`	The `target` option is used to specify whether the encryption setting applies to the client or server. Use `server`.

Option	Description
`type accepted|rejected|requested|required`	The `type` option is used to specify the action to take when negotiating encryption. • `rejected`—Enter this value if you do not elect to enable data encryption, even if required by the client. In this scenario, this side of the connection specifies that data encryption is not permitted. If the client side is set to `required`, the connection terminates with error message `ORA-12650`. If the client side is set to `requested`, `accepted` or `rejected`, the connection continues without error and without data encryption enabled. • `accepted`—Select this value to enable data encryption if required or requested by the client. In this scenario, this side of the connection does not require data encryption, but it is enabled if the client side is set to `required` or `requested`. If the client side is set to `required` or `requested`, and an encryption algorithm match is found, the connection continues without error and with data encryption enabled. If the client side is set to `required` and no algorithm match is found, the connection terminates with error message `ORA-12650`. If the client side is set to `requested` and no algorithm match is found, or if the client side is set to `accepted` or `rejected`, the connection continues without error and without data encryption enabled. • `requested`—Select this value to enable data encryption if the client permits it. In this scenario, this side of the connection specifies that data encryption is desired but not required. Data encryption is enabled if the client side specifies `accepted`, `requested`, or `required`. There must be a matching algorithm available, otherwise data encryption is not enabled. If the client side specifies `required` and there is no matching algorithm, the connection fails. • `required`—Select this value to enable data encryption or preclude the connection. In this scenario, this side of the connection specifies that data encryption must be enabled. The connection fails if the client side specifies `rejected` or if there is no compatible algorithm.

dbaascli netsec config integrity

The `netsec config integrity` subcommand of the `dbaascli` utility is used to configure Oracle Net integrity settings.

By default, database deployments on Database Cloud Service are configured to enable native Oracle Net integrity. You can use the `netsec config integrity` subcommand to change Oracle Net integrity settings. See "Configuring Oracle Database Network Encryption and Data Integrity" in *Oracle Database Security Guide* for Release 18, 12.2 or 12.1 or "Configuring Network Data Encryption and Integrity for

Oracle Servers and Clients" in *Database Advanced Security Administrator's Guide* for Release 11.2 for detailed information on Oracle Net encryption and integrity.

Execute this command as the `oracle` user.

```
dbaascli netsec config integrity
  --clevel accepted|rejected|requested|required
  --methods algorithm[,algorithm]...
  --target client|server
```

Options of this subcommand are as follows.

Option	Description
`clevel accepted|rejected|requested|required`	The `clevel` option is used to specify the checksum level. • `rejected`—Enter this value if you do not elect to enable data integrity, even if required by the client. In this scenario, this side of the connection specifies that data integrity is not permitted. If the client side is set to `required`, the connection terminates with error message `ORA-12650`. If the client side is set to `requested`, `accepted` or `rejected`, the connection continues without error and without data integrity enabled. • `accepted`—Select this value to enable data integrity if required or requested by the client. In this scenario, this side of the connection does not require data integrity, but it is enabled if the client side is set to `required` or `requested`. If the client side is set to `required` or `requested`, and an integrity algorithm match is found, the connection continues without error and with data integrity enabled. If the client side is set to `required` and no algorithm match is found, the connection terminates with error message `ORA-12650`. If the client side is set to `requested` and no algorithm match is found, or if the client side is set to `accepted` or `rejected`, the connection continues without error and without data integrity enabled. • `requested`—Select this value to enable data integrity if the client permits it. In this scenario, this side of the connection specifies that data integrity is desired but not required. Data integrity is enabled if the client side specifies `accepted`, `requested`, or `required`. There must be a matching algorithm available, otherwise data integrity is not enabled. If the client side specifies `required` and there is no matching algorithm, the connection fails. • `required`—Select this value to enable data integrity or preclude the connection. In this scenario, this side of the connection specifies that data integrity must be enabled. The connection fails if the client side specifies `rejected` or if there is no compatible algorithm.
`methods algorithm[,algorithm]...`	The `methods` option is used to specify the integrity algorithm. Valid values are: `SHA1`, `SHA256`, `SHA384`, and `SHA512`.`SHA1` is the only algorithm supported by Oracle Database 11*g*.

Option	Description
`target client|server`	The `target` option is used to specify whether the integrity setting applies to the client or server. Use `server`.

dbaascli netsec status

The `netsec status` subcommand of the `dbaascli` utility is used to display information about network encryption and network integrity configuration.

Execute this command as the `oracle` user.

```
dbaascli netsec status [encryption|integrity]
```

If no options are specified, the utility will display information about network encryption and network integrity configuration.

Options of this subcommand are as follows.

Option	Description
`encryption`	The `encryption` option is used to display information about network encryption configuration.
`integrity`	The `integrity` option is used to display information about network integrity configuration.

dbaascli patch db apply

This command is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm apply command.

The `patch db apply` subcommand of the `dbaascli` utility is used to apply a database patch.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch db apply --patchid {patch-id|LATEST}
```

The `patch db apply` subcommand displays progress as the patch is applied.

Options of this subcommand are as follows.

Option	Description
`--patchid {patch-id|LATEST}`	The `patchid` option is used to specify the database patch to be downloaded and applied. Specify the ID of the patch, or specify LATEST to download and apply the latest available database patch.

dbaascli patch db cleanup

 This command is not yet available on Oracle Cloud at Customer.

The `patch db cleanup` subcommand of the `dbaascli` utility is used to remove temporary files that may have been created when performing database patch operations. You can run this command after using the `patch db apply` or `patch db switchback` subcommands of the `dbaascli` utility.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch db cleanup
```

dbaascli patch db list

 This command is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm list_patches command.

The `patch db list` subcommand of the `dbaascli` utility is used to check whether any database patches are available.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch db list
```

A patch update is available if the command response includes the "INFO: images available for patching" message. The patch ID will be displayed as part of the command response. This patch ID can be used to download and apply the patch. See dbaascli patch db apply for detail on applying a patch.

dbaascli patch db prereq

 This command is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm prereq command.

The `dbpatchm prereq` subcommand of the `dbaascli` utility is used to check the prerequisites of a database patch.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch db prereq
```

dbaascli patch db switchback

 This command is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm rollback command.

The `patch db switchback` subcommand of the `dbaascli` utility is used to revert back to the state of the database deployment before a database patch was applied.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch db switchback
```

dbaascli patch os apply

The `patch os apply` subcommand of the `dbaascli` utility is used to apply the latest Linux OS patches that are available.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch os apply
```

dbaascli patch os list

The `patch os list` subcommand of the `dbaascli` utility is used to check whether any Linux OS patches are available.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch os list
```

A Linux OS patch is available if the command response includes the "INFO: tools images available for patching" message.

dbaascli patch tools apply

 This command is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm toolsinst command.

The `patch tools apply` subcommand of the `dbaascli` utility is used to download and apply a patch containing the cloud tooling update.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch tools apply --patchid {patch-id|LATEST}
```

Options of this subcommand are as follows.

Option	Description
`--patchid {patch-id|LATEST}`	The `patchid` option is used to specify the patch to be downloaded and applied. Specify the ID of the patch, or specify LATEST to download and apply the latest available patch.

dbaascli patch tools auto disable


This command is not yet available on Oracle Cloud at Customer.

The `patch tools auto disable` subcommand of the `dbaascli` utility is used to disable automatic cloud tooling updates for a compute node.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch tools auto disable
```

The utility disables automatic cloud tooling updates by removing the `/etc/crontab` file entry that was previously created by the dbaascli patch tools auto enable subcommand.

dbaascli patch tools auto enable


This command is not yet available on Oracle Cloud at Customer.

The `patch tools auto enable` subcommand of the `dbaascli` utility is used to enable automatic cloud tooling updates for a compute node.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch tools auto enable
```

An entry is added to the `/etc/crontab` file that causes the system to regularly check for cloud tooling updates and apply new updates to the compute node when they become available.

dbaascli patch tools auto execute


This command is not yet available on Oracle Cloud at Customer.

The `patch tools auto execute` subcommand of the `dbaascli` utility is used to apply the latest cloud tooling update to a compute node.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch tools auto execute
```

Automatic cloud tooling updates must be enabled for the compute node. See dbaascli patch tools auto enable. This command enables you to immediately check for cloud tooling updates and apply the latest update to the compute node, rather than wait for the next automatic check.

dbaascli patch tools auto status


This command is not yet available on Oracle Cloud at Customer.

The `patch tools auto status` subcommand of the `dbaascli` utility is used to check whether automatic cloud tooling updates are enabled or disabled for the compute node.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch tools auto status
```

Automatic cloud tooling updates are enabled if the command response includes the message "INFO: auto rpm update is enabled". If automatic updates are disabled, the command response includes the message "INFO: auto rpm update is disabled".

dbaascli patch tools list


This command is not yet available on Oracle Cloud at Customer. Instead, you must use the dbaascli dbpatchm list_tools command.

The `patch tools list` subcommand of the `dbaascli` utility is used to check whether any cloud tooling updates are available.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli patch tools list
```

A cloud tooling update is available if the command response includes the "INFO: tools images available for patching" message. The patch ID of the cloud tooling update will be displayed as part of the command response. This patch ID can be used to download and apply the patch.

dbaascli orec duplicate

The `orec duplicate` subcommand of the `dbaascli` utility is used to xxx in an Oracle Database Guard configuration.

You must execute this command on the standby instance's compute node as the `root` user.

```
dbaascli orec --args -duplicate
```

dbaascli orec keep list

The `orec latest` subcommand of the `dbaascli` utility is used to list the available long-term backups.

You must execute this command as the `root` user.

```
dbaascli orec --args -keep -list
```

dbaascli orec keep tag

The `orec keep tag` subcommand of the `dbaascli` utility is used to restore a specific long-term backup and perform recovery.

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli orec --args -keep -tag backup-tag
```

Options of this subcommand are as follows.

Option	Description
`–tag` *`backup-tag`*	The `tag` option is used to supply the backup tag of the long-term backup that should be restored for the recovery operation.

dbaascli orec latest

The `orec latest` subcommand of the `dbaascli` utility is used to restore the most recent backup and perform complete recovery.

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

You must execute this command as the `root` user.

```
dbaascli orec --args -latest -cfgfiles
```

dbaascli orec list

The `orec latest` subcommand of the `dbaascli` utility is used to list the available normal backups.

You must execute this command as the `root` user.

```
dbaascli orec --args -list
```

dbaascli orec pitr

The `orec pitr` subcommand of the `dbaascli` utility is used to restore a specific normal backup and perform recovery.

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli orec --args -pitr backup-tag -cfgfiles
```

Options of this subcommand are as follows.

Option	Description
`-pitr` *`backup-tag`*	The `pitr` option is used to supply the backup tag of the backup that should be restored for the recovery operation.

dbaascli orec scn

The `orec scn` subcommand of the `dbaascli` utility is used to restore the most recent backup and perform recovery through the specified system change number (SCN).

> **Note:**
>
> Beginning in version 18.4.6, the default behavior of the `dbaascli orec` subcommand changed. The command now downloads only the backup data. Use the new `-cfgfiles` option to also download the configuration files.

Connect to the compute node as the `opc` user and execute this command as the `root` user.

```
dbaascli orec --args -scn SCN -cfgfiles
```

Options of this subcommand are as follows.

Option	Description
`-scn` *`SCN`*	The `scn` option is used to supply the system change number (SCN) for the end point of the recovery operation.

dbaascli tde rotate masterkey

The `tde rotate masterkey` subcommand of the `dbaascli` utility is used to change (rotate) the master encryption key.

Execute this command as the `oracle` user.

```
dbaascli tde rotate masterkey
```

Enter the password specified during the database deployment creation process when prompted for the keystore password.

dbaascli tde status

The `tde status` subcommand of the `dbaascli` utility is used to view information about the software keystore used in tablespace encryption.

Execute this command as the `oracle` user.

```
dbaascli tde status
```

Output from the command includes the type of keystore and the status of the keystore.

E

The raccli Utility

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `raccli` utility is provided on Oracle Database Cloud Service deployments that use Oracle Real Application Clusters (RAC) to perform a variety of life-cycle and administration operations.

Using the `raccli` utility, you can perform operations like:

- Backing up the database
- Recovering the database from a backup
- Changing configuration of automatic backups
- Patching the Oracle Database, Grid Infrastructure and cloud tooling software
- Changing the configuration of security features
- Tracking the progress and completion of long-running operations performed as asynchronous jobs

To use the `raccli` utility:

1. Connect as the **opc** user to a compute node associated with the Database Cloud Service deployment.

   For instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Run the `raccli` utility using a command of the form:

   ```
   $ raccli subcommand subcommand-options
   ```

3. When you are finished using the `raccli` utility, disconnect from the compute node:

   ```
   $ exit
   ```

The `raccli` utility supports these subcommands:

`apply patch`: applies a patch to the Grid Infrastructure or Oracle Database home.

`clean backup`: deletes any unnecessary backups or backup pieces as determined by the backup configuration's recovery windows for local and cloud backups.

`create backup`: creates a backup of the configuration files, inventory configuration files, Grid Infrastructure and Oracle Database homes, database, and tooling metadata.

`create recovery`: initiates database recovery.

`describe job`: provides information about the progress and status of a long-running operation.

`describe system`: provides information about the installation of Grid Infrastructure, Oracle Database, and the RDK cloud tooling.

`failover dataguard`: performs a manual failover to the standby database in your Data Guard configuration.

`list backup`: provides a list and status of all the backup jobs that have been submitted through the `raccli` utility.

`list backupconfig`: provides a list of the backup configuration settings.

`list jobs`: provides a list of all the jobs that have been submitted through the `raccli` utility.

`list recovery`: provides information about recovery jobs.

`reinstate dataguard`: reinstates a failed primary database as a standby database after a failover.

`status dataguard`: displays status information about the Data Guard configuration.

`switchover dataguard`: performs a switchover to the standby database in your Data Guard configuration.

`update backupconfig`: updates the backup configuration.

`update databasepassword`: updates the password in the keystore (wallet) and optionally updates the password for the `SYS` and `SYSTEM` users.

`update netsec`: updates the Oracle Net security configuration.

`update rdk`: updates the cloud tooling on the Database Cloud Service deployment.

`update server`: checks for and corrects any configuration issues related to the Linux OS on a compute node of the deployment.

`update tde`: enables transparent data encryption (TDE) and rotates the TDE key.

raccli apply patch

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `apply patch` subcommand of the `raccli` utility is used to apply a patch to the Oracle Database and Grid Infrastructure home on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli apply patch  -db -tag tag-name  [-prechecks] [-dg]
```

This subcommand runs asynchronously. That is, it creates a job to apply the patch, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`-db`	Causes `raccli` to update the Oracle Database home and the Grid Infrastructure home. The Oracle Database home and Grid Infrastructure home are updated on both compute nodes of the deployment. The node from which you run the command is taken offline, patched, and then brought back online. Then the second node is taken offline, patched, and brought back online.
`-tag` *`tag-name`*	The name of the patch to apply. To find out the tag name for the latest available patch, see *What's New for Oracle Database Cloud Service*.
`-prechecks`	Causes `raccli` to precheck application of the patch instead of actually applying the patch. To perform the precheck, `raccli` uses the `opatchauto` utility's `apply -analyze` subcommand.
`-dg`	Causes `raccli` to perform the operation on the nodes of both the databases in a database deployment of type **Database Clustering with RAC and Data Guard Standby**. The nodes of the standby database are patched first, followed by the nodes of the primary database. **Note:** On older database deployments the node from which you run the command may not have the ssh key necessary to access the other database's nodes. In such cases, the `apply patch` subcommand displays instructions to copy ssh key files to the node.

raccli clean backup

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `clean backup` subcommand of the `raccli` utility deletes any unnecessary backups or backup pieces as determined by the backup configuration's recovery windows for local and cloud backups.

```
raccli clean backup  [-force]
```

Options of this subcommand are as follows.

Option	Description
`-force`	Causes the `raccli clean backup` command to ignore the recovery windows for local and cloud backups. Thus, all backups are deleted if you use this option.

raccli create backup

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `create backup` subcommand of the `raccli` creates a backup of the configuration files, inventory configuration files, Grid Infrastructure and Oracle Database home, database, and tooling metadata of an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli create backup  [-tag tag-name]
```

This subcommand runs asynchronously. That is, it creates a job to perform the backup, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`-tag tag-name`	The name of the backup job. This is not an RMAN tag.

Example

Here is an example of the `create backup` subcommand. The name of the backup in this example is `backuptest`. You can view details of the backup job by using the raccli describe job subcommand. You can view details of the backup by using the raccli list backup subcommand.

```
[opc@example1 ~]$ raccli create backup -tag backuptest

{
  "joburi" : "http://localhost:7070/dcs/7/responses",
  "requestStatus" : "SUCCESS",
  "jobid" : "7"
}
[opc@example1 ~]$
```

raccli create recovery


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `create recovery` subcommand of the `raccli` initiates database recovery of an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli create recovery  [-tag tag-name][-latest]|[-pitr][-scn SCN]|[-timestamp time]
```

This subcommand runs asynchronously. That is, it creates a job to perform the recovery, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`-tag` *`tag-name`*	The name of the recovery job. This is not an RMAN tag. If this option is omitted, a default name of “auto” is given.
`-latest`	Indicates that a complete database recovery should be performed.
`-pitr`	Indicates that a database point-in-time (incomplete) recovery should be performed. The recovery end point is specified by the `–scn` or `–timestamp` option.
`-scn` *`SCN`*	The system change number (SCN) for the end point of the recovery.
`-timestamp` *`time`*	The time for the end point of the recovery. The format is `MM/DD/YYYY HH24:MI:SS`.

Example

Here is an example of the `create recovery` subcommand. The name of the recovery job in this example is `recovertest`. This is an example of a request for a complete database recovery operation. You can use the job ID displayed in the output as input to the raccli describe job command to track the job’s progress.

```
[opc@example1 ~]$ raccli create recovery -tag recovertest -latest

{
  "joburi" : "http://localhost:7070/dcs/8/responses",
  "requestStatus" : "SUCCESS",
  "jobid" : "8"
}
[opc@example1 ~]$
```

raccli describe job

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `describe job` subcommand of the `raccli` utility provides information about the progress and status of a long-running operation.

```
raccli describe job  job-id
```

The output of this subcommand depends on the kind of operation that the given job is.

Options of this subcommand are as follows.

Option	Description
`job-id`	The ID of the job you want to view detailed information about.

Example

Here is an example of the `describe job` subcommand. This example shows details about the backup job in the raccli create backup example.

```
opc@example1 ~]$ raccli describe job 7

{
  "requestStatus" : "SUCCESS",
  "jobStatus" : "SUCCESS",
  "message" : null,
  "response" : [ {
    "startTime" : "Thu Oct 15 18:37:38 UTC 2015",
    "endTime" : "Thu Oct 15 18:37:39 UTC 2015",
    "status" : "SUCCESS",
    "taskId" : "TaskZJsonRpcExt_6939",
    "taskResult" : "Resource {  id: 1444934258987, name: null, description: null }",
    "taskName" : "DB Config files backup",
    "taskDescription" : null
  }, {
    "startTime" : "Thu Oct 15 18:37:39 UTC 2015",
    "endTime" : "Thu Oct 15 18:38:10 UTC 2015",
    "status" : "SUCCESS",
    "taskId" : "TaskZJsonRpcExt_6941",
    "taskResult" : "Resource {  id: 1444934260034, name: null, description: null }",
    "taskName" : "Database Backup",
    "taskDescription" : null
  }, {
    "startTime" : "Thu Oct 15 18:38:11 UTC 2015",
    "endTime" : "Thu Oct 15 18:38:11 UTC 2015",
    "status" : "SUCCESS",
    "taskId" : "TaskZJsonRpcExt_6943",
    "taskResult" : "Resource {  id: 1444934291029, name: null, description: null }",
    "taskName" : "Persisting Backup metadata",
    "taskDescription" : null
  } ]
}
[opc@example1 ~]$
```

raccli describe system


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `describe system` subcommand of the `raccli` utility provides information about the installation of Grid Infrastructure, Oracle Database, and the RDK cloud tooling on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli describe system
```

Example

Here is an example of the `describe system` subcommand.

```
[opc@example1 ~]$ raccli describe system

"serviceName": "example"
{
        "NodeName": "example1"
        {
                "componentType": "GridHome"
                "componentName": "OraGrid12102"
                "componentVersion": "12.1.0.2.0(21297657, 20299018)"
```

```
        }
        {
                "componentType": "DatabaseHome"
                "componentName": "OraDB12102_home1"
                "componentVersion": "12.1.0.2.10 (21125181)"
        }
        {
                "componentType": "RDK"
                "componentName": "RDK"
                "componentVersion": "15.4.1.0.0"
        }
}
{
        "NodeName": "example2"
        {
                "componentType": "GridHome"
                "componentName": "OraGrid12102"
                "componentVersion": "12.1.0.2.0(21297657, 20299018)"
        }
        {
                "componentType": "DatabaseHome"
                "componentName": "OraDB12102_home1"
                "componentVersion": "12.1.0.2.10 (21125181)"
        }
        {
                "componentType": "RDK"
                "componentName": "RDK"
                "componentVersion": "15.4.1.0.0"
        }
}

[opc@example1 ~]$
```

raccli failover dataguard


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `failover dataguard` subcommand of the `raccli` utility is used to perform a manual failover to the standby database in your Data Guard configuration. Run this subcommand on a compute node of the standby database you are failing over to.

```
raccli failover dataguard [-dbname database-name]  -passwd database-password
```

When the failover operation completes successfully, the once-primary database is no longer a member of the Data Guard configuration, and the once-standby database has the role of primary database.

This subcommand runs asynchronously. That is, it creates a job to perform the failover operation, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

> **Note:**
>
> You must manually configure backups after the failover operation completes. For instructions, see Manual backup configuration required after switchover or failover on Oracle RAC plus Data Guard deployments in *Known Issues for Oracle Database Cloud Service*.

Options of this subcommand are as follows.

Option	Description
`-dbname` *`database-name`*	(Optional) The database unique name of the standby database that you are failing over to.
`-passwd` *`database-password`*	The password of the database's SYS user.

raccli list backup


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `list backup` subcommand of the `raccli` utility provides a list and status of all the backup jobs on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli list backup
```

Example

Here is an example of the `list backup` subcommand. The output in this example shows the detail of the backup taken in the raccli create backup example.

```
[opc@example1 ~]$ raccli list backup

[ {
  "updatedTimestamp" : "2015-10-11, 01:03:01",
  "id" : 1444525202870,
  "name" : "rdbaas_backup",
  "description" : null,
...
      "location" : "DISK",
      "info" : "Location /u03/app/oracle"
    } ]
  } ],
  "jobId" : "6"
}, {
  "updatedTimestamp" : "2015-10-15, 18:38:11",
  "id" : 1444934257459,
  "name" : "rdbaas_backup",
  "description" : null,
  "createTimeStamp" : "2015-10-15, 18:37:37",
  "type" : "Backup",
  "tagName" : "backuptest",
```

```
  "endTime" : null,
  "backupComponents" : [ {
    "updatedTimestamp" : "2015-10-15, 18:37:39",
    "id" : 1444934258987,
    "name" : null,
    "description" : null,
    "createTimeStamp" : "2015-10-15, 18:37:38",
    "type" : null,
    "component_name" : "DBConfig",
    "backupLocations" : [ {
      "updatedTimestamp" : "2015-10-15, 18:37:39",
      "id" : 1444934259945,
      "name" : null,
      "description" : null,
      "createTimeStamp" : "2015-10-15, 18:37:39",
      "type" : null,
      "location" : "DISK",
      "info" : "Location /u03/app/oracle"
    } ]
  }, {
    "updatedTimestamp" : "2015-10-15, 18:38:11",
    "id" : 1444934260034,
    "name" : null,
    "description" : null,
    "createTimeStamp" : "2015-10-15, 18:37:40",
    "type" : null,
    "component_name" : "Database",
    "backupLocations" : [ {
      "updatedTimestamp" : "2015-10-15, 18:38:10",
      "id" : 1444934290955,
      "name" : null,
      "description" : null,
      "createTimeStamp" : "2015-10-15, 18:38:10",
      "type" : null,
      "location" : "DISK",
      "info" : "Location /u03/app/oracle"
    } ]
  } ],
  "jobId" : "7"
} ]
[opc@example1 ~]$
```

raccli list backupconfig


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `list backupconfig` subcommand of the `raccli` utility provides a list of the backup configuration settings for an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli list backupconfig
```

Example

Here is an example of the `list backupconfig` subcommand.

```
[opc@example1 ~]$ raccli list backupconfig
```

```
{
  "updatedTimestamp" : "2015-10-10, 01:41:20",
  "id" : 1444441280176,
  "name" : "rdbaas_backup_config",
  "description" : null,
  "createTimeStamp" : "2015-10-10, 01:41:20",
  "type" : "BackupConfig",
  "dbHomeBackup" : false,
  "giHomeBackup" : false,
  "backupOsConfigFiles" : true,
  "backupGiConfigFiles" : true,
  "diskEnabled" : true,
  "ossEnabled" : false,
  "diskRecoveryWindow" : 7,
  "ossRecoveryWindow" : 30,
  "cronDate" : "01:00",
  "backupConfigFiles" : true,
  "osConfigFilesRef" : "/opt/oracle/dcs/rdbaas/config/oscfg.spec",
  "giConfigFilesRef" : "/opt/oracle/dcs/rdbaas/config/gicfg.spec",
  "dbConfigFilesRef" : "/opt/oracle/dcs/rdbaas/config/dbcfg.spec",
  "cloudStorageServiceName" : null,
  "cloudStorageIdentityDomain" : null,
  "cloudStorageUser" : null,
  "cloudStorageHost" : null,
  "cloudStoreContainer" : null
}
[opc@example1 ~]$
```

raccli list jobs


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `list jobs` subcommand of the `raccli` utility provides a list of all the jobs on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli list jobs
```

Example

Here is an example of the `list jobs` subcommand.

```
[opc@example1 ~]$ raccli list jobs

Job Id    Job Name                    Status             Message
1          service creation           Success            null
2         backup creation             Success            null
3         backup creation             Success            null
4         backup creation             Success            null
5         backup creation             Success            null
6         backup creation             Success            null
7         backup creation             Success            null
8         Create recovery             Running            null

[opc@example1 ~]$
```

raccli list recovery

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `list recovery` subcommand of the `raccli` utility provides information about recovery jobs.

`raccli list recovery [-id` *`recovery-id`*`][-tag` *`recovery-tagname`*`]`

Options of this subcommand are as follows.

Option	Description
`-id` *`recovery-id`*	The system-generated ID of the recovery job. Use this option to display information about a specific recovery job.
`-tag` *`recovery-tagname`*	The name supplied when the recovery job was submitted. Use this option to display information about a specific recovery job.

Example

Here is an example of the `list recovery` subcommand. The output in this example shows the detail of the recovery performed in the raccli create recovery example.

```
[opc@example1 ~]$ raccli list recovery

[ {
  "updatedTimestamp" : "2015-10-15, 18:47:17",
  "id" : 1444934529920,
  "name" : "rdbaas_recovery",
  "description" : null,
  "createTimeStamp" : "2015-10-15, 18:42:09",
  "type" : "Recovery",
  "recoveryComponents" : [ {
    "updatedTimestamp" : "2015-10-15, 18:47:17",
    "id" : 1444934530186,
    "name" : "rdbaas_recovery_database",
    "description" : null,
    "createTimeStamp" : "2015-10-15, 18:42:10",
    "type" : null,
    "component_type" : "Database",
    "recoveryParams" : [ {
      "updatedTimestamp" : "2015-10-15, 18:42:10",
      "id" : 1444934530485,
      "name" : "rdbaas_recovery_database_parameters",
      "description" : null,
      "createTimeStamp" : "2015-10-15, 18:42:10",
      "type" : null,
      "parameter" : "latest",
      "value" : "true"
    } ]
  } ],
  "recoveryTag" : "backuptest",
  "endTime" : null,
  "jobId" : "8"
```

```
} ]
[opc@example1 ~]$
```

raccli reinstate dataguard


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `reinstate dataguard` subcommand of the `raccli` utility is used to reinstate a failed primary database as a standby database after a failover. Run this subcommand on a compute node of the current primary database.

```
raccli reinstate dataguard [-dbname database-name]  -passwd database-password
```

When the reinstate operation completes successfully, the once-primary database is reinstated as a member of the Data Guard configuration in the role of standby database.

> **Note:**
>
> If your database deployment is running Oracle Database 12*c* Release 2 (12.2) or later, then **before** you run the `raccli reinstate dataguard` command you must make sure the database you are reinstating is stopped or is started in `MOUNT` mode. To start the database in `MOUNT` mode, enter these commands:
>
> ```
> srvctl stop database -d db-unique-name -o abort
> srvctl start database -d db-unique-name -o mount
> ```

The `raccli reinstate dataguard` subcommand runs asynchronously. That is, it creates a job to perform the reinstate operation, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`-dbname` *`database-name`*	(Optional) The database unique name of the database that you are reinstating.
`-passwd` *`database-password`*	The password of the database's SYS user.

raccli status dataguard

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `status dataguard` subcommand of the `raccli` utility displays status information about the Data Guard configuration. You can run this subcommand on a compute node of either the primary or standby database.

```
raccli status dataguard  -passwd database-password
```

This subcommand provides information about the Data Guard configuration, including its primary and standby databases and the Oracle RAC nodes hosting each of the databases.

Options of this subcommand are as follows.

Option	Description
`-passwd` *`database-password`*	The password of the database's SYS user.

raccli switchover dataguard

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `switchover dataguard` subcommand of the `raccli` utility is used to perform a switchover to the standby database in your Data Guard configuration. You can run this subcommand on a compute node of either the primary or standby database.

```
raccli switchover dataguard [-dbname database-name]  -passwd database-password
```

When the switchover operation completes successfully, the roles of the two databases are reversed: what was the primary database is now the standby database, and what was the standby database is now the primary database. Both remain as members of the Data Guard configuration.

This subcommand runs asynchronously. That is, it creates a job to perform the switchover, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

> **Note:**
>
> You must manually configure backups after the switchover operation completes. For instructions, see Manual backup configuration required after switchover or failover on Oracle RAC plus Data Guard deployments in *Known Issues for Oracle Database Cloud Service*.

Options of this subcommand are as follows.

Option	Description
`-dbname` *`database-name`*	(Optional) The database unique name of the standby database that will be the primary database after the switchover completes.
`-passwd` *`database-password`*	The password of the database's SYS user.

raccli update backupconfig


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `update backupconfig` subcommand of the `raccli` utility updates the backup configuration on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

> **Note:**
>
> If you have used the `update rdk` subcommand of the `raccli` utility to update the cloud tooling to 16.4.5 or later, you must manually update the opc installer for the Oracle Database Cloud Backup Module before you use the `update backupconfig` subcommand. For instructions, see in Updating the cloud tooling on a deployment hosting Oracle RAC requires manual update of the Oracle Database Cloud Backup Module in *Known Issues for Oracle Database Cloud Service*.

```
raccli update backupconfig  -params parameter-string
```

This subcommand runs synchronously.

Options of this subcommand are as follows.

Option	Description
`-params` *`parameter-string`*	The parameters to update.

Parameters

The parameter string is a single JSON object that specifies the backup configuration settings to be updated.

The following table describes the parameters that can be specified in this JSON string.

Parameter	Description
`backupDerby`	Controls whether a backup of the metadata store for each RAC compute node is taken. Enter `true` (default) or `false`.
`backupGiConfigFiles`	Controls whether a backup of the Grid Infrastructure configuration files is taken. Enter `true` (default) or `false` .
`backupOsConfigFiles`	Controls whether a backup of operating system configuration files is taken. Enter `true` (default) or `false` .
`dbHomeBackup`	Controls whether a backup of the Oracle Database home is taken. Enter `true` or `false` (default).
`cloudStorageContainerUrl`	The URL of the Oracle Storage Cloud Service container.
`cloudStoragePwd`	The password of the user specified in `cloudStorageUser`. You must also provide the user name in `cloudStorageUser`.
`cloudStorageUser`	The user name of an Oracle Cloud user who has read/write access to the container. You must also provide the password of the user in `cloudStoragePwd`.
`cronDate`	Specifies the time (using 24-hour, HH:MM format) when daily backups are to occur. For example, 02:45 is 2:45 AM, and 14:45 is 2:45 PM. The default value is `01:00`.
`diskEnabled`	Controls whether backups to local storage on the compute node are configured. Enter `true` or `false`. You must enter the same value in `ossEnabled`.
`diskRecoveryWindow`	The number of days for which backups and archived redo logs on local storage are maintained. The interval always ends with the current time and extends back in time for the number of days specified. The default value is 7.
`giHomeBackup`	Controls whether a backup of the Grid Infrastructure home is taken. Enter `true` or `false` (default).
`ossEnabled`	Controls whether backups to an Oracle Storage Cloud Service container using Oracle Database Backup Cloud Service are to be configured. Enter `true` or `false`. You must enter the same value in `diskEnabled`.
`ossRecoveryWindow`	The number of days for which backups on cloud storage are maintained. The interval always ends with the current time and extends back in time for the number of days specified. The default value is `30`.

Example

Here is an example of the `update backupconfig` subcommand. You can use the raccli list backupconfig subcommand to view your changes.

```
[opc@example1 ~]$ raccli update backupconfig -params '{"cronDate" : "02:45"}'

{
  "requestStatus" : "SUCCESS",
  "jobStatus" : "SUCCESS",
  "message" : null,
  "response" : [ ]
}
  [opc@example1 ~]$
```

raccli update databasepassword

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `update databasepassword` subcommand of the `raccli` utility updates the password in the keystore (wallet) and optionally updates the password of the `SYS` and `SYSTEM` users.

> **Note:**
>
> This command is only applicable if your database deployment uses Oracle Real Application Clusters (RAC) and Oracle Data Guard together.
>
> You must execute this command if you have used the `update rdk` command to update the cloud tooling from release 17.2.1 or an earlier release.

```
raccli update databasepassword -password  password [-saveonly]
```

If you are executing this command following an update to the cloud tooling from release 17.2.1 or earlier, you must execute the command with the `–saveonly` option on both the primary and standby database to update the password in the keystore.

If you are executing this command to update the password in the keystore and to update the password for the `SYS` and `SYSTEM` users, you must also copy the password file from the primary database to the standby database after executing the command on the primary database for Oracle Database 11*g* and Oracle Database 12*c* Release 1. You do not need to copy the password file for Oracle Database 12*c* Release 2 or later releases.

This subcommand runs synchronously.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`–password` *password*	The password specified when the database deployment was created, unless you have changed the wallet password since the deployment was created.

Option	Description
–saveonly	This option indicates the password should be updated only in the keystore (wallet). If you omit this option when executing the command on the primary database, the password is updated in the keystore (wallet) and for the SYS and SYSTEM users. If you omit this option when executing the command on the standby database, the password is updated only in the keystore (wallet).

raccli update netsec

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `update netsec` subcommand of the `raccli` utility updates the Oracle Net security configuration on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli update netsec {-encryption|-integrity} {-server|-client}  -type type  -
algorithm algorithm
```

This subcommand runs synchronously.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
-encryption	Controls whether Oracle Net Services data encryption is configured.
-integrity	Controls whether Oracle Net Services data integrity is configured.
-server	Specifies that the Oracle Net Services data encryption or data integrity configuration is for the server. Use this value.
-client	Specifies that the Oracle Net Services data encryption or data integrity configuration is for the client.

Option	Description
`-type` *type*	Specifies how Oracle Net Services data encryption or data integrity is negotiated with clients. For data encryption enter one of these values: • `rejected`—Enter this value if you do not elect to enable data encryption, even if required by the client. In this scenario, this side of the connection specifies that data encryption is not permitted. If the client side is set to `required`, the connection terminates with error message `ORA-12650`. If the client side is set to `requested`, `accepted` or `rejected`, the connection continues without error and without data encryption enabled. • `accepted`—Select this value to enable data encryption if required or requested by the client. In this scenario, this side of the connection does not require data encryption, but it is enabled if the client side is set to `required` or `requested`. If the client side is set to `required` or `requested`, and an encryption algorithm match is found, the connection continues without error and with data encryption enabled. If the client side is set to `required` and no algorithm match is found, the connection terminates with error message `ORA-12650`. If the client side is set to `requested` and no algorithm match is found, or if the client side is set to `accepted` or `rejected`, the connection continues without error and without data encryption enabled. • `requested`—Select this value to enable data encryption if the client permits it. In this scenario, this side of the connection specifies that data encryption is desired but not required. Data encryption is enabled if the client side specifies `accepted`, `requested`, or `required`. There must be a matching algorithm available, otherwise data encryption is not enabled. If the client side specifies `required` and there is no matching algorithm, the connection fails. • `required`—Select this value to enable data encryption or preclude the connection. In this scenario, this side of the connection specifies that data encryption must be enabled. The connection fails if the client side specifies `rejected` or if there is no compatible algorithm. For data integrity enter one of these values: • `rejected`—Enter this value if you do not elect to enable data integrity, even if required by the client. In this scenario, this side of the connection specifies that data integrity is not permitted. If the client side is set to `required`, the connection terminates with error message `ORA-12650`. If the client side is set to `requested`, `accepted` or `rejected`, the connection continues without error and without data integrity enabled. • `accepted`—Select this value to enable data integrity if required or requested by the client. In this scenario, this side of the connection does not require data integrity, but it is enabled if the client side is set to

Option	Description
	`required` or `requested`. If the client side is set to `required` or `requested`, and an integrity algorithm match is found, the connection continues without error and with data integrity enabled. If the client side is set to `required` and no algorithm match is found, the connection terminates with error message `ORA-12650`. If the client side is set to `requested` and no algorithm match is found, or if the client side is set to `accepted` or `rejected`, the connection continues without error and without data integrity enabled. • `requested`—Select this value to enable data integrity if the client permits it. In this scenario, this side of the connection specifies that data integrity is desired but not required. Data integrity is enabled if the client side specifies `accepted`, `requested`, or `required`. There must be a matching algorithm available, otherwise data integrity is not enabled. If the client side specifies `required` and there is no matching algorithm, the connection fails. • `required`—Select this value to enable data integrity or preclude the connection. In this scenario, this side of the connection specifies that data integrity must be enabled. The connection fails if the client side specifies `rejected` or if there is no compatible algorithm.
`-algorithm` *`algorithm`*	The algorithm to be used for data encryption or data integrity. For `encryption`, the choices are `AES128`, `AES192`, and `AE256`. For `integrity` with Oracle Database 12c and later releases, the choices are `SHA1`, `SHA512`, `SHA384`, and `SHA25`. For `integrity` with Oracle Database 11g, the only accepted value is `SHA1`.

raccli update rdk

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `update rdk` subcommand of the `raccli` utility updates the cloud tooling on an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

> **Note:**
>
> If you are updating a database deployment that also uses Oracle Data Guard and are updating from release 17.2.1 or earlier, you must also execute the `update databasepassword` command to store the password in the keystore (wallet).

```
raccli update rdk -tag tag-number
```

> **Caution:**
>
> After executing the `raccli update rdk` command, wait three minutes before executing commands using either raccli or the user interface to allow time for the server to restart.

This subcommand runs asynchronously. That is, it creates a job to update the cloud tooling, reports the job ID of the created job, and then exits. To track the progress of the job to its completion, use the `raccli describe job` command.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`-tag tag-number`	The tag of the cloud tooling to update to. For *`tag-number`* enter the version of tooling you want to update to without the dots in the version number. For example, to update to 17.2.3 tooling you would enter **1723**. To find out the tag number for the latest available tooling update, see *What's New for Oracle Database Cloud Service*.

Example

Here is an example of the `update rdk` subcommand. You can use the raccli describe job subcommand to view details about the job that is started when you execute this subcommand.

```
[opc@example1 ~]$ raccli update rdk -tag 1723

{
  "jobId" : "10",
  "requestStatus" : "SUCCESS"
}
  [opc@example1 ~]$
```

raccli update server

 This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `update server` subcommand of the `raccli` utility checks for and corrects any configuration issues related to the Linux OS running on a compute node of an Oracle Database Cloud Service database deployment that uses Oracle Real Application Clusters (RAC).

```
raccli update server -tag tag-number [-no-reboot] [-allnodes]
```

Options of this subcommand are as follows.

Option	Description
`-tag tag-number`	The tag of the server update to run. In general, you specify the value of your current RDK version, as reported by the raccli describe system command, but without the dots or trailing zeros. For example, if `raccli describe system` reports an RDK version of `18.1.1.0.0`, you would specify **1811** as the tag number in a `raccli update server` command.
`-no-reboot`	When specified, the `update server` subcommand does not reboot the compute node after correcting any configuration issues. You must manually reboot it, as described in Rebooting a Compute Node. If `-allnodes` is also specified, neither compute node is rebooted and you must manually reboot each of them.
`-allnodes`	When specified, the `update server` subcommand checks for and corrects any configuration issues on the current compute node and then performs the same operation on the other compute node of the cluster.

raccli update tde


This topic does not apply to Oracle Cloud Infrastructure or to Oracle Cloud at Customer.

The `update tde` subcommand of the `raccli` utility provides a way to enable transparent data encryption (TDE) and rotate the TDE key.

```
raccli update tde {-enable|-rotatekey} -passwd  password
```

This subcommand runs synchronously.

If the subcommand fails, it reports a FAILURE status and provides a message describing the reason for the failure.

Options of this subcommand are as follows.

Option	Description
`–enable`	Used to enable the TDE configuration.
`–rotatekey`	Used to rotate the key.
`–passwd password`	The password specified when the database deployment was created, unless you have changed the wallet password since the deployment was created.

Example

Here is an example of the `update tde` subcommand. This example rotates (changes) the value of the master key.

```
[opc@example1 ~]$ raccli update tde -rotatekey -passwd Welcome_1

{
  "requestStatus" : "SUCCESS",
  "jobStatus" : "SUCCESS",
```

```
  "message" : null,
  "response" : [ ]
}
[opc@example1 ~]$
```

F

The dbpatchmdg Utility

The `dbpatchmdg` utility is provided on Oracle Database Cloud Service deployments that are configured with Oracle Data Guard to perform a variety of patching operations.

Using the `dbpatchmdg` utility, you can perform operations like:

- List available patches
- Check if nodes are ready for patching
- Apply patches
- Roll back patches

Topics

Running the dbpatchmdg Utility

The `dbpatchmdg` utility is provided on Oracle Database Cloud Service deployments that are configured with Oracle Data Guard to perform a variety of patching operations.

To use the `dbpatchmdg` utility:

1. Connect as the **opc** user to a compute node associated with the Database Cloud Service deployment.

   For instructions, see Connecting to a Compute Node Through Secure Shell (SSH).

2. Run the `dbpatchmdg` utility using a command of the form:

   ```
   # dbpatchmdg -option option-options
   ```

3. When you are finished using the `dbpatchmdg` utility, exit the root-user command shell and disconnect from the compute node:

   ```
   # exit
   $ exit
   ```

dbpatchmdg apply_async

> **Note:**
>
> This command is deprecated as of release 18.3.6 in September 2018. It is replaced by the dbaascli patch db apply command. You should discontinue using this deprecated command as it will be removed in a future release.

The `apply_async` option of the `dbpatchmdg` utility is used to apply a patch to the Oracle Database home on an Oracle Database Cloud Service database deployment hosting an Oracle Data Guard configuration.

```
dbpatchmdg -apply_async patch_id  [-sshkey=/root/sshkey] [-txn_fl=/var/opt/
oracle/log/dbpatchmdg/file_name] [-exclude_node=ip address] [-no_switchover]
```

The `apply_async` option of the `dbpatchmdg` utility applies the patch on all nodes in the Oracle Data Guard configuration, one after another. The node hosting the standby database is patched first. A switchover operation is performed after patching is complete on the standby and then the patch is applied to the node hosting the new standby.

Options of `dbpatchmdg apply_async` are as follows.

Option	Description
`patch_id`	The ID of the patch to be applied.
`-sshkey=/root/sshkey`	The SSH key. Provide this if the SSH key has changed.
`-txn_fl=/var/opt/oracle/log/dbpatchmdg/file_name`	The file that the master transaction record will be written to.
`-exclude_node=ip address`	The IP address of a node that should be excluded from patching. If you want to specify this option, you must patch the standby database before patching the primary database.
`-no_switchover`	Indicates that no switchover should take place during patching. With this option, the primary database will incur downtime. If you want to specify this option, you must patch the standby database before patching the primary database.

dbpatchmdg precheck_async

> **Note:**
>
> This command is deprecated as of release 18.3.6 in September 2018. It is replaced by the dbaascli patch db prereq command. You should discontinue using this deprecated command as it will be removed in a future release.

The `precheck_async` option of the `dbpatchmdg` utility is used to check the prerequisites of a patch before you apply it to an Oracle Database Cloud Service database deployment hosting an Oracle Data Guard configuration.

```
dbpatchmdg -precheck_async patch_id  [-sshkey=ssh_file] [-txn_fl=/var/opt/oracle/log/
dbpatchmdg/file_name] [-exclude_node=ip address]
```

Options of `dbpatchmdg precheck_async` are as follows.

Option	Description
`patch_id`	The ID of the patch to be checked.
`-sshkey=ssh_file`	The SSH key file name. Provide this if the SSH key has changed.
`-txn_fl=/var/opt/oracle/log/dbpatchmdg/file_name`	The file that the master transaction record will be written to.
`-exclude_node=ip address`	The IP address of a node that should be excluded from the check.

dbpatchmdg rollback_async

> **Note:**
>
> This command is deprecated as of release 18.3.6 in September 2018. It is replaced by the dbaascli patch db switchback command. You should discontinue using this deprecated command as it will be removed in a future release.

The `rollback_async` option of the `dbpatchmdg` utility is used to roll back a patch that has been applied on a Oracle Database Cloud Service database deployment hosting an Oracle Data Guard configuration.

```
dbpatchmdg -rollback_async patch_id  [-sshkey=ssh_file] [-txn_fl=/var/opt/oracle/log/
dbpatchmdg/file_name] [-exclude_node=ip address] [-no_switchover]
```

Options of `dbpatchmdg rollback_async` are as follows.

Option	Description
`patch_id`	The ID of the patch to be rolled back.
`-sshkey=ssh_file`	The SSH key file name. Provide this if the SSH key has changed.
`-txn_fl=/var/opt/oracle/log/dbpatchmdg/file_name`	The file that the master transaction record will be written to.
`-exclude_node=ip address`	The IP address of a node that should be excluded from the roll back.
`-no_switchover`	Indicates that no switchover should take place during the operation. With this option, the primary database will incur downtime.

G

Using Oracle DBaaS Monitor

> **Note:**
>
> As of May 2018, Oracle DBaaS Monitor is decommissioned in favor of Oracle SQL Developer Web. Beginning May 2018, new Oracle Database Cloud Service database deployments of single-instances databases include SQL Developer Web instead of DBaaS Monitor. If you have an older database deployment, you should replace DBaaS Monitor with SQL Developer Web as soon as is feasible. For instructions, see Updating the Cloud Tooling on Database Cloud Service. For information about SQL Developer Web, see Using Oracle SQL Developer Web in Database Cloud Service.

Oracle DBaaS Monitor provides monitoring and management of the Oracle database and listener on Oracle Database Cloud Service.

Topics

About Oracle DBaaS Monitor

Oracle DBaaS Monitor provides monitoring and management of the Oracle database and listener on Oracle Database Cloud Service.

DBaaS Monitor provides quick and easy access to a variety of information about the database instance running on a database deployment:

- Overall, how much storage is allocated to tablespaces, and how much of that storage is used

- For each tablespace, how much storage is allocated and how much of that storage is used, with additional drill-down capabilities to view segments
- A real-time graph showing wait events across several selectable categories
- The alert log, with log searching capabilities
- A list of open user sessions, with drill-down capabilities to view session details such as the last SQL statement, explain plan, waits, contention, and so on
- A list of initialization parameters, with the ability to change parameter values, both in memory and in the SPFILE.
- Indication of whether certain database options are enabled
- Monitoring of current and past SQL Developer PDB uploads
- A list of the SQL statements that are being monitored in the database, with real time display of details such as the status, duration, degree of parallelism, and so on

You can use DBaaS Monitor to view information about the compute node:

- CPU utilization information in an interactive table format, with automatic refresh intervals
- OS process information, with filtering and automatic refresh capabilities

DBaaS Monitor also provides the following management capabilities:

- Start up and shut down the database instance
- Open and close a pluggable database
- Create and drop a pluggable database
- Plug in and unplug a pluggable database
- Clone a pluggable database
- Start and stop the listener

Accessing Oracle DBaaS Monitor

Database deployments of single-instance databases on Oracle Database Cloud Service include Oracle DBaaS Monitor, a built-in monitor that provides a wide spectrum of information about Oracle Database and operating system status and resource usage.

You can access Oracle DBaaS Monitor in the following ways:

- Using the “Open DBaaS Monitor Console” menu item
- Using a direct URL
- Using an SSH tunnel

Using the "Open DBaaS Monitor Console" Menu Item to Access Oracle DBaaS Monitor

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access Oracle DBaaS Monitor is blocked by default. To use the **Open DBaaS Monitor Console** menu item, you must unblock port 443, either by enabling the deployment's **ora_p2_httpssl** predefined access rule or by creating your own access rule that opens port 443. For instructions, see Enabling Access to a Compute Node Port.

1. Open the Services page of the Oracle Database Cloud Service console.

   For detailed instructions, see Accessing the Oracle Database Cloud Service Console.

2. From the menu for the deployment, select **Open DBaaS Monitor Console**.
3. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.

4. When prompted for a user name and password, enter **dbaas_monitor** as the user name and the password specified during the database deployment creation process, and then click **OK**.

   Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

The **Open DBaaS Monitor Console** menu item is also available on the Overview page in the menu next to the deployment's name.

Using a Direct URL to Access Oracle DBaaS Monitor

> **Note:**
>
> For database deployments built on Oracle Cloud Infrastructure Classic, the network port to access Oracle DBaaS Monitor is blocked by default. To use a direct URL, you must unblock port 443, either by enabling the deployment's **ora_p2_httpssl** predefined access rule or by creating your own access rule that opens port 443. For instructions, see Enabling Access to a Compute Node Port.

1. In your web browser, go to the following URL:

   `https://node-ip-address/dbaas_monitor`

where *node-ip-address* is the IP address of the deployment's compute node as listed on the deployment's Overview page.

2. If your browser displays a warning that your connection is not secure or not private, use the browser's advanced option to continue.

   You get this warning because Database Cloud Service database deployments use a self-signed certificate to provide HTTPS (secure HTTP) connectivity, and such certificates are considered suspicious by many web browsers.

3. When prompted for a user name and password, enter **dbaas_monitor** as the user name and the password specified during the database deployment creation process, and then click **OK**.

   Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

Using an SSH Tunnel to Access Oracle DBaaS Monitor

1. Create an SSH tunnel to port 443 on the compute node hosting Oracle DBaaS Monitor. For information about creating an SSH tunnel, see Creating an SSH Tunnel to a Compute Node Port.
2. After creating the SSH tunnel, direct your browser to the URL `https://localhost/dbaas_monitor`.
3. When prompted for a user name and password, enter **dbaas_monitor** as the user name and the password specified during the database deployment creation process, and then click **OK**.

   Note: if the database deployment was created using a QuickStart template, the password is available in the zip file downloaded when the deployment was created.

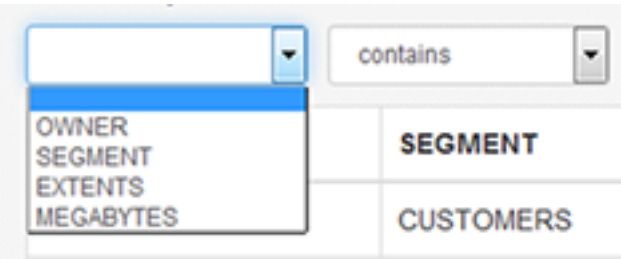
Filtering the Display on DBaaS Monitor Pages

On DBaaS Monitor pages, the PDB selector enables you to limit the display of information in Oracle Database 12c and later releases. You can also filter what items display in tabular lists.

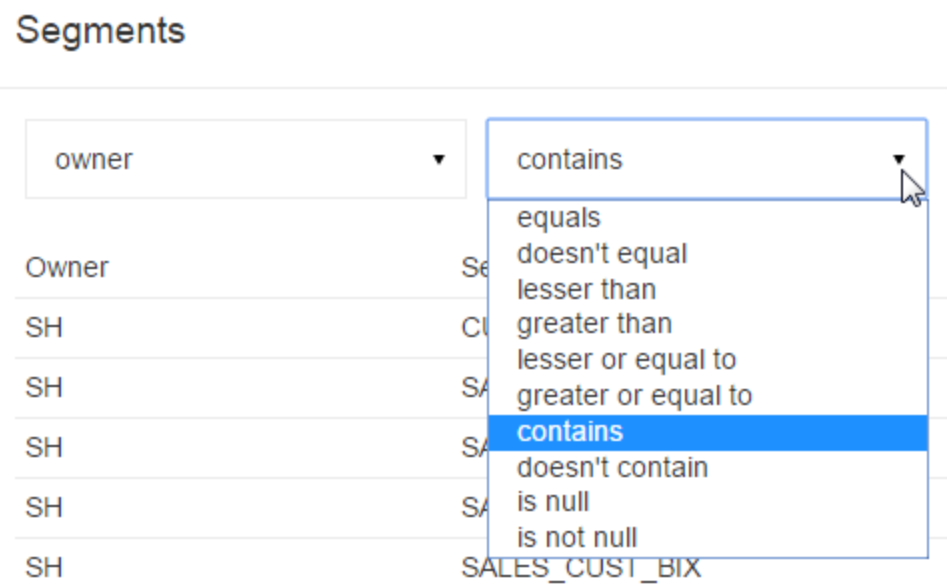
For Oracle Database 12c and later releases, you can use the PDB selector to limit displayed information. The default selection, Overall, results in no filtering; information for the container database and all PDBs is shown. The CDB$ROOT selection shows only tablespace information for the tablespaces in the container database root. The PDB$SEED selection shows only tablespace information for the tablespaces in the pluggable database seed. The PDB selector also lets you limit displayed information to a specific pluggable database.

On DBaaS Monitor pages that show tabular lists, you can filter what items the table displays.

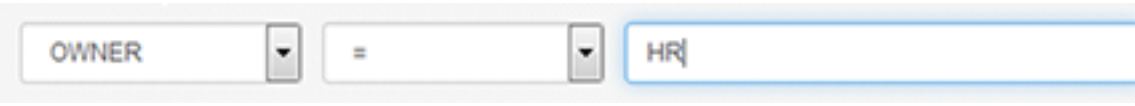
1. Expand the first drop-down menu and select the column you want to use as the filter.



2. In the second drop-down menu, select the operator.



3. In the box, enter the value and click the search icon.



Administering the Listener

You can use DBaaS Monitor to administer the listener, including troubleshooting unknown service name errors.

Topics

- Viewing Listener Status Information
- Starting the Listener
- Stopping the Listener
- Verifying that the Listener Knows of a Service

Viewing Listener Status Information

The Oracle Net Listener (the listener) is a process that resides on the server. It listens for incoming client connection requests and manages traffic to the server.

To review listener status information:

- On the DBaaS Monitor home page, click **Listener**. Or in the Database drop-down menu, click **Listener**.

  The RDBMS Listener page displays.

The RDBMS Listener page shows the following information:

- Status of the listener, including start time
- Protocol addresses the listener is configured to listen on
- Summary of the database services registered with the listener and the service handlers allocated to each service
- Registered database services (service name), database instance associated with the service, and connection status

Starting the Listener

By default, an Oracle Net listener is automatically started in your database deployment. If you have shut it down or it has crashed, you can use DBaaS Monitor to start the listener by performing the following steps:

1. On the DBaaS Monitor home page, click **Listener**. Or in the Database drop-down menu, click **Listener**.
2. In the Listener page menu, click **Turn on** to start the listener.

Stopping the Listener

To stop the listener:

1. On the DBaaS Monitor home page, click **Listener**. Or in the Database drop-down menu, click **Listener**.
2. Click **Turn off** to stop the listener.

Verifying that the Listener Knows of a Service

An "ORA-12514: Listener Does Not Currently Know of Service Requested in Connect Descriptor" error indicates that the specified service name is unknown by the Oracle Net Listener process. You can verify that the specified service name is configured with the listener by performing the following steps:

1. On the DBaaS Monitor home page, click **Listener**. Or in the Database drop-down menu, click **Listener**.

   The RDBMS Listener page displays.
2. Review the information displayed to determine whether the requested service name is listed.

Starting and Stopping the Database Instance

You can use DBaaS Monitor to start up and shut down your database instance.

Topics

Starting the Database Instance

You can use DBaaS Monitor to start the database instance in OPEN mode. In this mode the instance is started, the database is mounted and then opened.

If you need to start the database instance in any other mode (NOMOUNT, MOUNT, or FORCE), use SQL*Plus instead of DBaaS Monitor. For more information, see "Starting Up and Shutting Down" in *Oracle Database Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2.

To start the database instance in OPEN mode:

1. On the DBaaS Monitor home page, click Database Status. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your database, click **Start database**.

   The database instance is started.

Stopping the Database Instance

You can use DBaaS Monitor to shut down the database instance in IMMEDIATE mode. In this mode, no new connections are allowed. No new transactions are allowed to be started and any uncommitted transactions are rolled back.

If you need to shut down the database instance in any other mode (ABORT, NORMAL, or TRANSACTIONAL), use SQL*Plus instead of DBaaS Monitor. For more information, see "Starting Up and Shutting Down" in *Oracle Database Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2.

To shut down the database instance in IMMEDIATE mode:

1. On the DBaaS Monitor home page, click Database Status. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your database, click **Stop database**.

   The database instance is shut down.

Viewing and Modifying Initialization Parameters

Initialization parameters configure the database instance, including memory structures, and define locations for database files.

Values for initialization parameters are stored in a text-based initialization parameter file (PFILE) or binary server parameter file (SPFILE). The initialization parameter file is

read at database instance startup. For more information, see "Initialization Parameters" in *Oracle Database Reference* for Release 18, 12.2, 12.1 or 11.2.

To view and modify the value of an initialization parameter:

1. From any DBaaS Monitor page, click Parameters from the Database drop-down menu.
2. On the Parameters page, use the arrow buttons in the lower right corner to navigate the multiple pages of parameters, or use the search fields at the top of the page.
3. To perform a search, enter values in the search criteria columns and click the search icon to locate the initialization parameter.
4. Enter the new value in the Value field.
5. Expand the Save changes menu and select the appropriate value:
   - To Memory: Updates the value for the existing database instance, but does not save it to the SPFILE.
   - To SPFILE: Updates the value in the SPFILE, but does not change it in the existing instance. The new value will take effect when the instance is restarted.
   - To both: Updates the values for the existing database instance and updates the value in the SPFILE.
6. Click **Yes** to confirm your change.
7. Click **OK** to close the Results page.

Viewing User Account and Expiring Password Information

You can use DBaaS Monitor to view information about the status of user accounts and passwords.

You can use the DBaaS Monitor to view how many user accounts are in the following states:

- Open—This status indicates that the user's account is unlocked and access to the database is enabled.
- Locked—This status indicates that the user's account is locked and access to the database is disabled. The account must be unlocked to enable access to the database.
- Expired—This status indicates that the user's password has expired and must be changed before the user can log in to the database.

You can view this information for the entire database or for a specific PDB.

In the Expiring Accounts Password box on the DBaaS Monitor home page, you can see a list of user accounts and whether a user account password has expired or the number of days before it will expire. You can view this information for the entire database or for a specific PDB.

Viewing Tablespace and Segment Space Usage

You can use DBaaS Monitor to view tablespace and segment space usage.

A tablespace is a database storage unit that groups related logical structures together. A tablespace is comprised of datafiles. A segment is a set of extents allocated from a tablespace for a specific database object such as a table or index.

To view space usage information

1. On the DBaaS Monitor home page, click **Online Database Storage**. Alternatively, from any DBaaS Monitor page, click **Storage** from the Database drop-down menu.

   The Storage page displays. If the Oracle database is version 12c or later, the Storage page shows the used and allocated storage space for all tablespaces in the container in the root, and the used and allocated storage space for tablespaces in any pluggable databases. If the Oracle database is version 11g, the Storage page shows the used and allocated space for the entire database.
2. For an Oracle database version 12c or later, you can click **show tablespaces** for the container database, or click **show tablespaces** for each pluggable database.
3. When you click **show tablespaces**, a list of tablespaces appears. You can click a tablespace to view its storage information. An interactive report appears, showing the segments that exist within the tablespace. Most segments are user objects, and they include tables, LOBs, and indexes.
4. On the Segments page, you can refine the list of segments shown by using the filter feature.

   For example, you can search for all the segments for a specific owner (schema) by selecting OWNER from the first drop-down list, entering the owner (schema) name in the box, and clicking the search icon.

Changing the TDE Keystore Password

You can use DBaaS Monitor to change the password of the TDE keystore.

When the database deployment is created on Database Cloud Service, a local autologin software keystore is created. The keystore is local to the compute node and is protected by a system-generated password. The keystore is part of the key-based architecture that is used to transparently encrypt (and decrypt) tablespaces.

> **Note:**
>
> This feature is only available on database deployments using Oracle Database 12*c* or later.

To change the TDE keystore password:

1. On the DBaaS Monitor home page, click Database Status. Or, in the Database pulldown menu, click **Manage**.
2. In the menu for your database, click **Change TDE Keystore Password**.

3. In the Change TDE Keystore Password dialog box, enter the current keystore password and the new keystore password.
4. Click **OK**.
5. Click **OK** on the confirmation message.

Viewing Alert Log Entries and Checking for Errors

You can use DBaaS Monitor to review the alert log periodically to verify that your database system is operating normally.

The alert log is a chronological log of messages including the following:

- Nondefault initialization parameters used at startup
- Administrative operations, such as STARTUP, SHUTDOWN, ARCHIVE LOG, RECOVER, and CREATE/ALTER/ DROP DATABASE/TABLESPACE
- Messages and errors relating to the functions of certain background processes, such as LGWR
- Internal errors (ORA-600), block corruption errors (ORA-1578), and deadlock errors (ORA-60)

To view alert log entries:

1. On any DBaaS Monitor page, click **Alerts** from the Database drop-down menu.
2. Search for a specific value by selecting the display column in the first drop-down list, selecting the condition in the second drop-down list, and entering your search value in the box. Then, click the search icon.

The Alerts box on the DBaaS Monitor Home page has two sections, each of which can be clicked for further detail. One section indicates the total number of all messages. The other section indicates how many errors have been recorded in the alert log in the last 7 days. To view errors in the alert log:

1. On the DBaaS Monitor home page, click **ERRORS** in the Alerts box.
2. On the Alerts page select **Type** from the first drop-down list and enter your search criteria in the third field. Then, click the search icon.

   Errors recorded in the alert log are displayed.

Viewing Real Time SQL Monitor

This page shows, in real time, the SQL statements that are being monitored in the database.

To review the SQL statements being monitored:

- On the DBaaS Monitor home page, in the Database drop-down menu, click **Real Time SQL Monitor**.

  The Real Time SQL Monitor page displays.

This tool helps identify run-time issues for SQL statements and monitor their behavior, by providing two major functions:

- General view of monitored statements

- View of SQL execution details

General View of Monitored Statements

The page contains a table of SQL statements currently running. This table shows the following information:

- **Status** — Current state of the SQL statement execution. For example, a SQL statement that has already finished its execution will show a status of "DONE".
- **Duration** — This is the amount of time a SQL statement is taking, or has taken, to execute.
- **SQL ID** — SQL identifier of the statement being monitored.
- **Session ID** — Session identifier that is executing, or has executed, the SQL statement.
- **Session Serial Number** — Used to uniquely identify a session's objects.
- **Instance Degree of Parallelism** — This Degree of Parallelism (DOP) column shows how many instances and parallel execution servers are allocated. It is shown in the form of “number of instances” | “number of parallel servers”
- **CPU Time** — This is the CPU time consumed by the execution of the query.
- **I/O Time** — This is the I/O time consumed by the execution of the query.
- **Start Time** — This is the time in which the execution of the SQL statement started.
- **SQL Statement** — This is the SQL statement being monitored.

For more information, see "Monitoring the Database" in *Oracle Database Administrator's Guide* for Release 18, 12.2, 12.1 or 11.2.

View of SQL Execution Details

When a SQL statement is drilled down from the main monitor table, a detailed view is shown. The SQL ID, Start Time and the SQL Execution ID represent the execution key that uniquely identify this SQL statement. A detail view consists of the general characteristics that integrate the execution of a SQL statement.

General information about the query execution is provided:

- **Execution Plan** — Degree of Parallelism of the SQL statement
- **Execution Started** — Time that the SQL statement execution started
- **Last Refresh Time:** — Last update time of the SQL monitor registry for the SQL statement
- **Execution ID** — Execution identifier
- **User** — User in the format USER@CONTAINER
- **Fetch Calls** — Number of fetch calls done by the SQL statement

General statistics of the SQL statement are provided: total duration of execution, the number of buffered gets, number of Input/Output requests and bytes.

A duration breakdown shows the percentage of the total duration of the execution of the SQL statement, divided into two types of times:

- **Database Time** — Any time of execution related to the database needs; for example, CPU Time
- **Wait Time** — The waiting time that the statement goes through to complete an execution

Each duration time can be drilled down by clicking on the "Show Detail" element, where a gauge graph shows the percentage of time that integrates each type of duration time.

PLSQL and Java Time are provided. These time measurements are outside of the duration of the SQL statement.

Detailed information of the statement — This space holds the information corresponding to the explain plan, parallel behaviour and CPU activity involved in the execution of the statement:

- **SQL Text** — SQL statement that was or is being executed.
- **Plan Statistics** — Explain plan of the execution of the SQL statement in the form of a table. Each row is a different operation involved in the execution of the SQL statement and it shows hierarchy dependency by adding a space at the beginning of the text in the Operation column.
  - Operation, Name, Estimated Rows, Cost, Actual Rows, Memory, Temp(Max), IO Requests, IO Bytes
- **Graphic view of the Plan Statistics** — The plan statistics table in a graphic representation; the hierarchy is presented as a collapsible tree map in which each node represent an operation.
- **Parallelism Details for the SQL statement** — Each execution consists of a parallel coordinator and one or more parallel sets. Each set can have one or more processes. When a row has dependents, each of its columns will be the sum of the values of its dependants. When this happens, a sigma symbol will appear to show that a value consists of the sum of others. The columns shown are the following:
  - Process Name, Buffer Gets, CPU Time, Elapsed Time, Other Wait Time, Server Set, Read Requests, Read Bytes
- **Activity Line Chart for the CPU Usage** — Line chart showing the number of different types of CPU activities registered in the execution of the SQL statement. The Y axis represents the number of CPU activities and the X axis represents the time registered for that activity. Each activity is represented by a custom set of colors depending on the activity.

Administering Pluggable Databases

You can use DBaaS Monitor to create and drop a pluggable database, open and close a pluggable database, plug in and unplug a pluggable database, and clone a pluggable database.

Topics

Cloning a Pluggable Database

You can use DBaaS Monitor to clone a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your pluggable database, click **Clone**.
3. Complete the Clone PDB dialog and click **OK**.
    - **New PDB Name**—Provide a name for the new PDB.
    - **Source PDB**—This field is pre-filled with the name of the PDB you chose as the source for cloning.
    - **Sparse Clone**—(Not available on Oracle Cloud at Customer) By default, a full clone is created. Select this option to instead create a sparse clone (also called a snap clone).
    - **File Destination**—If Sparse Clone is selected, the File Destination field is made visible. Specify the base file system directory for the cloned PDB's files.
    - **File Name Conversions**— You can optionally provide custom names and expressions for the PDB datafiles.
    - **Unlimited Storage** and **Reuse Temp File**— By default, unlimited storage and reuse temp file are selected for the PDB.
    - **Clone TDE Key**—TDE Key cloning is selected by default.
    - **Keystore Password**—If Clone TDE Key is selected, the Keystore Password field is made visible. Use the password that was specified during the database deployment creation process.

Closing a Pluggable Database

You can use DBaaS Monitor to close a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your pluggable database, click **Modify state**.
3. The Modify PDB dialog appears. In the State field, click **CLOSE**.
4. In the Option field, the default is IMMEDIATE. Choose **IMMEDIATE** or **NORMAL**.
5. Click **OK**.

Creating a Pluggable Database

You can use DBaaS Monitor to create a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. Click **Create PDB**.
3. Complete the Create PDB dialog and click **OK**.
   - **New PDB Name**—Provide a name for the new pluggable database.
   - **Admin Username** and **Admin Password**—Provide a user name and password for the PDB administrator. You need not use the same admin password that you used during the database deployment creation process, unless you wish.
   - **File Name Conversions**—You can optionally provide custom names and expressions for the PDB datafiles.
   - **Unlimited Storage** and **Reuse Temp File**—By default, unlimited storage and reuse temp file are selected for the PDB.
   - **Create TDE Key**—TDE Key creation is selected by default.
   - **Keystore Password**—If Create TDE Key is selected, the Keystore Password field is made visible. Use the password that was specified during the database deployment creation process.

Dropping a Pluggable Database

You can use DBaaS Monitor to drop a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your PDB, click **Drop**.
3. In the Drop PDB dialog, choose whether you want to keep or delete the PDB's datafiles. Then click **OK**.

Opening a Pluggable Database

You can use DBaaS Monitor to open a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your pluggable database, click **Modify state**.
3. The Modify PDB dialog appears. In the State field, click **OPEN**.
4. In the Option field, the default is READ WRITE. Choose **READ WRITE**, **READ ONLY** or **RESTRICTED**.
5. Click **OK**.

Plugging In a Pluggable Database

You can use DBaaS Monitor to plug in a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. Click **Plug in PDB**.

3. Complete the Plug PDB dialog and click **OK**.
    - **New PDB Name**—Provide a name for the pluggable database.
    - **Plug as Clone** —This field is selected by default. If you are plugging in a PDB as a clone, fill the XML Filename field.
    - **Source File Name Conversion**—You can optionally provide custom expressions for the PDB datafiles.
    - **Copy Action**—The default value is Don't Copy. Choose **Don't Copy**, **Copy**, or **Move**.
    - **Unlimited Storage** and **Reuse Temp File**—By default, unlimited storage and reuse temp file are selected for the PDB.
    - **Import TDE Key**—TDE Key import is selected by default.
    - **Keystore Password, Import From, Key Secret**—If Import TDE Key is selected, the Keystore Password, Import From, and Key Secret fields are made visible. For Keystore Password, use the password that was specified during the database deployment creation process.

Unplugging a Pluggable Database

You can use DBaaS Monitor to unplug a pluggable database.

1. On the DBaaS Monitor home page, click **Database Status**. Or, in the Database pull-down menu, click **Manage**.
2. In the menu for your pluggable database, click **Unplug**.
3. Complete the Unplug PDB dialog and click **OK**.
    - **PDB** —This field is pre-filled with the name of the pluggable database you selected.
    - **XML Filename** —This field is pre-filled with the path to the PDB's XML file.
    - **Export TDE Key**—This field isselected by default.
    - **Keystore Password, Export To, Key Secret**—If Export TDE Key is selected, the Keystore Password, Export To, and Key Secret fields are made visible. For Keystore Password, use the password that was specified during the database deployment creation process.