Oracle® Database

Migrating Non-CDBs to New Hardware with the Same Operating System and for a New Release

18c
E97282-06
May 2019

Oracle Database Migrating Non-CDBs to New Hardware with the Same Operating System and for a New Release, 18c

E97282-06

Primary Authors: Sunil Surabhi, Nirmal Kumar

Contributing Authors: Lance Ashdown, Padmaja Potineni, Rajesh Bhatiya, Prakash Jashnani, Douglas Williams, Mark Bauer

Contributors: Roy Swonger, Byron Motta, Hector Vieyra Farfan, Carol Tagliaferri, Mike Dietrich, Marcus Doeringer, Umesh Aswathnarayana Rao, Rae Burns, Subrahmanyam Kodavaluru, Cindy Lim, Amar Mbaye, Akash Pathak, Thomas Zhang, Zhihai Zhang

Contents

9 Upgrading Oracle Database

10 Post-Upgrade Tasks for Oracle Database

Preface

This guide provides a compilation of topics from the Oracle Database user assistance documentation that are collected to help you complete a specific use case scenario.

Use Case Scenario for this Document

Use this scenario document to assist you to migrate Oracle Database to new hardware by duplicating an active database to a remote server using RMAN. After duplicating the active database, you can then upgrade the database.

While you duplicate the database on your physical standby Oracle Database instance, you do not need to shut down your primary Oracle Database instance. The database remains fully accessible to users while you are performing duplication.

Prerequisites for this Scenario

- Ensure that the operating system and the file system are identical on both the source and destination servers.
- Ensure that you have installed the new release Oracle Database software on the destination server.
- Ensure that the source and duplicate database files use the same directory structure.

Outline for this Scenario

1. **Migrating and Upgrading Oracle Database**. Review migration methods using active database duplication with backup sets.
2. **Preparing the Servers and Network for Database Duplication**. Use RMAN to prepare the servers and network for duplication.
3. **Preparing the Primary Instance on the Source Host**. Use RMAN to prepare your primary database.
4. **Preparing the Auxiliary Instance on the Destination Host**. Complete these procedures to prepare your auxiliary instance for switchover.
5. **Duplicating the Primary Oracle Database Instance**. Use `RMAN DUPLICATE`.
6. **Refresh and switchover to the Physical Standby Oracle Database**. Complete these procedures to switch over to your auxiliary instance.
7. **Upgrading Oracle Database**. Carry out the upgrade on the primary instance.

8. **Post-Upgrade Tasks for Oracle Database**. Complete the post-upgrade tasks on the upgraded Oracle Database instance, including testing the upgraded instance in preparation for switching back.

These steps correspond to the chapters in this document.

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.

1

Migrating and Upgrading Oracle Database

Create a standby Oracle Database on a remote host by duplicating an active Oracle Database.

- Overview of Active Database Duplication
  Active database duplication does not require backups of the source database. It duplicates the live source database to the destination host by copying the database files over the network to the auxiliary instance. RMAN can copy the required files as image copies or backup sets.
- Active Database Duplication Using Backup Sets
  Using backup sets to perform active database duplication is also known as the pull-based method of active database duplication.

Overview of Active Database Duplication

Active database duplication does not require backups of the source database. It duplicates the live source database to the destination host by copying the database files over the network to the auxiliary instance. RMAN can copy the required files as image copies or backup sets.

For active database duplication, the duplication technique used determines which channel performs the principal work. When active database duplication is performed using backup sets, the principal work of duplication is performed by the auxiliary channels. When image copies are used, the primary work is performed by the target channels.

To perform active database duplication, a connection to the target database is required. Oracle recommends that you use active database duplication in general, unless network bandwidth between the source host and the destination host is a constraint. Active database duplication requires minimal setup and is simpler to perform.

> **Note:**
>
> For active database duplication, the source database must use a server parameter file.

Some of the scenarios in which active database duplication using backup sets may be preferred over using image copies are:

- You want to use multisection backups, compression, or encryption while duplicating your database.
- The source database does not have sufficient network resources to transfer the required database files to the duplicate database.
- You want to minimize the resources used by the duplication process.

Active database duplication with backup sets uses minimal resources on the source database.

Active Database Duplication Using Backup Sets

Using backup sets to perform active database duplication is also known as the pull-based method of active database duplication.

In this method, RMAN connects as TARGET to the source database and as AUXILIARY to the auxiliary instance. The auxiliary instance then connects to the source database through Oracle Net Services and retrieves the required database files, over the network, from the source database.

Using backup sets for active database duplication provides the following advantages:

- RMAN can use unused block compression, thus reducing the size of backups that must be transported over the network.
- Backup sets can be created in parallel, on the source database, by using multisection backups.
- Backup sets created on the source database can be encrypted.

2

Preparing Servers and Network for Database Duplication

Create a standby database by duplicating the active database. RMAN copies the datafiles directly from the primary database to the standby database.

You must mount or open the primary database before running the `RMAN DUPLICATE FROM ACTIVE DATABASE` command.

- Prerequisites Specific to Active Database Duplication
  When you execute `DUPLICATE` with `FROM ACTIVE DATABASE`, at least one normal target channel and at least one `AUXILIARY` channel are required.
- Checklist for Preparing Active Database Duplication
  Ensure that you prepare source and target databases before using RMAN to carry out active database duplication.
- Configuring RMAN Channels for Use in Oracle Database Duplication
  RMAN channels perform the primary job of database duplication
- Configuring Automatic Channels Across File Systems
  Configure a set of persistent, automatic channels for use in all RMAN sessions.
- Configuring Channels for Active Database Duplication
  With active database duplication, you need not change your source database channel configuration or configure auxiliary channels. However, you may want to increase the parallelism setting of the source database disk channels so that RMAN copies files over the network in parallel.

Prerequisites Specific to Active Database Duplication

When you execute `DUPLICATE` with `FROM ACTIVE DATABASE`, at least one normal target channel and at least one `AUXILIARY` channel are required.

If you do not configure or preallocate channels, RMAN allocates the necessary channels by default. If you configure or manually allocate channels for active duplication with backup sets, ensure that the number of auxiliary channels is greater than or equal to the number of target channels.

When you connect RMAN to the source database as `TARGET`, you must specify a user name and password, even if RMAN uses operating system authentication. The connection to the auxiliary instance must use the same user name and password as the source database connection. The source database must be mounted or open. If the source database is open, then archiving must be enabled. If the source database is not open, then it must have been shut down consistently.

When you connect RMAN to the auxiliary instance, the following rules apply:

- When running RMAN on the same host as the auxiliary instance, you can connect locally without a net service name, provided that you connect using a user name and password, and provided that your `DUPLICATE` command does not include the

`PASSWORD FILE` clause. The connecting user must have the `SYSDBA` or `SYSBACKUP` privilege.

- When connecting remotely, or when using the `PASSWORD FILE` clause in the `DUPLICATE` command, you must connect using a net service name. You must first create a password file for the auxiliary instance.

The source database and auxiliary instances must use the same `SYS` and `SYSBACKUP` password, which means that both instances must have password files. The password file must contain at least two passwords, for the `SYS` and `SYSBACKUP` users. You can start the auxiliary instance and enable the source database to connect to it.

The `DUPLICATE` behavior for password files varies depending on whether your duplicate database will act as a standby database. If you create a duplicate database that is not a standby database, then RMAN does not copy the password file by default. You can specify the `PASSWORD FILE` option to indicate that RMAN can overwrite the existing password file on the auxiliary instance. If you create a standby database, then RMAN copies the password file to the standby host by default, overwriting the existing password file. In this case, the `PASSWORD FILE` clause is not necessary.

You cannot use the `UNTIL` clause when performing active database duplication. RMAN chooses a time based on when the online data files have been completely copied, so that the data files can be recovered to a consistent point in time.

Checklist for Preparing Active Database Duplication

Ensure that you prepare source and target databases before using RMAN to carry out active database duplication.

Source Oracle Database:

- To migrate the source database, you need the database name, database unique name, listener port, service name, database home patch level, and the password for SYS.
- If you have configured source database with Transparent Data Encryption (TDE), then you need a backup of the wallet and the wallet password to allow database duplication with encrypted data.
- The source database can be either in the open or in the mount state.
  - If the source database is open, then it must be in archivelog mode.

    The source database remains fully accessible to users while you are performing the database duplication. Be prepared to take a slight hit on CPU usage and network bandwidth consumption during datafile duplication.
  - If the source database is in the mount state, then shut it down cleanly before bringing it up to the mount state.

> **Note:**
>
> If you choose to maintain the source database in mount state, then the users cannot access the database.

Target Oracle Database:

- A target database system that supports the same database edition as the source database edition.
- Ensure that you have the target database name, database unique name, auxiliary service name, and applied current database home patch level.
- A free TCP port in the target database to setup the auxiliary instance.

Configuring RMAN Channels for Use in Oracle Database Duplication

RMAN channels perform the primary job of database duplication

Each channel corresponds to an Oracle Database server session that performs the duplication tasks. Depending on the duplication technique, RMAN uses either auxiliary channels or target channels.

Use one of the following methods to configure channels:

- Automatically allocate channels by using the `CONFIGURE` command
- Manually allocate channels by using the `ALLOCATE` command

If no automatic channels are configured, then you can manually allocate at least one channel before you begin the duplication. The `ALLOCATE` command that allocates channels must be in the same `RUN` block as the `DUPLICATE` command.

RMAN can use the same channel configurations on the source database for duplication on the destination host even if the source database channels do not specify the AUXILIARY option.

Configuring Automatic Channels Across File Systems

Configure a set of persistent, automatic channels for use in all RMAN sessions.

This example configures automatic disk channels across two file systems:

```
CONFIGURE DEVICE TYPE DISK PARALLELISM 2;
CONFIGURE CHANNEL 1 DEVICE TYPE DISK FORMAT '/disk1/%U';
CONFIGURE CHANNEL 2 DEVICE TYPE DISK FORMAT '/disk2/%U';
```

Because `PARALLELISM` is set to 2, the following command divides the backup pieces between two file systems:

```
BACKUP DEVICE TYPE DISK
  DATABASE PLUS ARCHIVELOG;
```

The following `LIST` command shows how the data file backup was parallelized:

```
RMAN> LIST BACKUPSET 2031, 2032;

List of Backup Sets
===================
```

```
BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
2031    Full    401.99M    DISK        00:00:57     19-JAN-07
        BP Key: 2038   Status: AVAILABLE  Compressed: NO  Tag:
TAG20070119T100532
        Piece Name: /disk1/24i7ssnc_1_1
  List of Datafiles in backup set 2031
  File LV Type Ckp SCN    Ckp Time  Name
  ---- -- ---- ---------- --------- ----
  1       Full 973497     19-JAN-07 /disk3/oracle/dbs/t_db1.f
  5       Full 973497     19-JAN-07 /disk3/oracle/dbs/tbs_112.f

BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
2032    Full    133.29M    DISK        00:00:57     19-JAN-07
        BP Key: 2039   Status: AVAILABLE  Compressed: NO  Tag:
TAG20070119T100532
        Piece Name: /disk2/25i7ssnc_1_1
  List of Datafiles in backup set 2032
  File LV Type Ckp SCN    Ckp Time  Name
  ---- -- ---- ---------- --------- ----
  2       Full 973501     19-JAN-07 /disk3/oracle/dbs/t_ax1.f
  3       Full 973501     19-JAN-07 /disk3/oracle/dbs/t_undo1.f
  4       Full 973501     19-JAN-07 /disk3/oracle/dbs/tbs_111.f
```

Configuring Channels for Active Database Duplication

With active database duplication, you need not change your source database channel configuration or configure auxiliary channels. However, you may want to increase the parallelism setting of the source database disk channels so that RMAN copies files over the network in parallel.

The type of active database duplication technique used determines which channels perform the principal work of duplication. When image copies are used to perform active database duplication, the primary work is performed by the target channels. Configure multiple target channels on the source database to improve the duplication performance. When active database duplication is performed by using backup sets, the principal work of duplication is performed by the auxiliary channels. Therefore, it is recommended that you allocate additional auxiliary channels. The number of auxiliary channels must be greater than or equal to the number of target channels. Using backup sets for active duplication also enables parallelism, which can improve the speed of the duplication process.

3

Preparing the Primary Instance on the Source Host

Prepare the primary Oracle Database instance by configuring net services.

- Configuring SQL*Net to Prepare the Primary Instance on the Source Host
  Add entries for primary and standby databases in the `tnsnames.ora` file, and then save the file.

Configuring SQL*Net to Prepare the Primary Instance on the Source Host

Add entries for primary and standby databases in the `tnsnames.ora` file, and then save the file.

Example 3-1 Adding Net Services

```
[oracle @ ora12c-prm ~] $ cd $ ORACLE_HOME / network / admin
[oracle @ ora12c-prm admin] $ cat tnsnames.ora
 # tnsnames.ora Network Configuration File: /u01/app/oracle/product/12.2.0/
dbhome/network/admin/tnsnames.ora
 # Generated by Oracle configuration tools.
 DUPDB =
 (DESCRIPTION =
 (ADDRESS_LIST =
 (ADDRESS = (PROTOCOL = TCP) (HOST = ora12c-dup) (PORT = 1521))
 )
 (CONNECT_DATA =
 (SERVICE_NAME = dupdb )
 )
 )
 PRMDB =
 (DESCRIPTION =
 (ADDRESS = (PROTOCOL = TCP) (HOST = ora12c-prm.localdomain) (PORT =
1521))
 (CONNECT_DATA =
 (SERVER = DEDICATED)
 (SERVICE_NAME = prmdb )
 )
 )
```

4

Preparing the Auxiliary Instance on the Destination Host

RMAN uses an auxiliary instance to create the duplicate database. You must prepare the auxiliary instance before you begin the duplication.

- Installing the Oracle Database Software on the Destination Host
  When the source and destination host are different, you must install the Oracle Database software on the destination host, so that the auxiliary instance can be created.
- Steps to Create an Initialization Parameter File for the Auxiliary Instance
  The initialization parameter file for the auxiliary instance must contain at least the `DB_NAME` and `DB_DOMAIN` initialization parameters. Additional parameters may be specified, if required. Ensure that the initialization parameter file is on the same host as the RMAN client that performs the duplication.
- Copying the Server Parameter File from the Source Database
  If the source database uses a server parameter file, then including the `SPFILE` option in the `DUPLICATE` command directs RMAN to use the server parameter file from the source database for the auxiliary instance.
- Creating a Password File for the Auxiliary Instance
  Connections to the auxiliary instance can be established by using operating system authentication or password file authentication. For backup-based duplication, you can either create a password file or use operating system authentication to connect to the auxiliary instance. For active database duplication, you must use password file authentication.
- Creating Directories for the Duplicate Database
  On the destination host, you must create the directories that RMAN uses to store the duplicate database files on the destination host.
- Using the Same Names for Database Files in the Source Database and Duplicate Database
  Certain conditions must be met to use the same names for files in the source and duplicate database.
- Establishing Oracle Net Connectivity Between the Source Database and Auxiliary Instance
  You must be able to establish a connection between the source database and auxiliary instance for certain forms of duplication.
- Configuring the Network Between Source and Target Oracle Databases
  In preparation for duplicating your Oracle Database, configure the network between your source and target Oracle Database instances.
- Starting the Auxiliary Instance
  The initialization parameter file that you create is used to start the auxiliary instance.

- Making the Oracle Keystore Available to the Destination Host
  If transparent encryption is configured on the source database, then you must ensure that the Oracle software keystore from the source database is available to the auxiliary instance. Manually copy the keystore from the source database to the destination host.
- Starting RMAN and Connecting to Databases
  You must start the RMAN client and connect to the database instances as required by the chosen duplication technique. The RMAN client can be located on any host so long as it can connect to the necessary databases over the network.

Installing the Oracle Database Software on the Destination Host

When the source and destination host are different, you must install the Oracle Database software on the destination host, so that the auxiliary instance can be created.

> **Note:**
>
> Ensure that you install the same release of Oracle Database software, with the same patch level, on both the source and destination host.

Use one of the following techniques to install the software:

- Perform a normal installation with Oracle Universal Installer (OUI).

  Install an Oracle Database whose release number is the same as that of the source database. Do not create a database; install the software only. Apply any required patches.
- Clone the source Oracle home.

  Use OUI to clone the source Oracle home. This ensures that all patches applied to the source database are present in the duplicate database.

Steps to Create an Initialization Parameter File for the Auxiliary Instance

The initialization parameter file for the auxiliary instance must contain at least the `DB_NAME` and `DB_DOMAIN` initialization parameters. Additional parameters may be specified, if required. Ensure that the initialization parameter file is on the same host as the RMAN client that performs the duplication.

To create the initialization parameter file for the auxiliary instance:

1. Do one of the following:
   - Copy the initialization parameter file from the source host to the destination host, placing it in the operating system-specific default location, and then modify the `DB_NAME` and `DB_DOMAIN` initialization parameters.

If you are duplicating a CDB, ensure that the `ENABLE_PLUGGABLE_DATABASE` parameter is present and set to `TRUE`.

See Copying the Server Parameter File from the Source Database.

- Complete these steps:

  a. Using a text editor, create an empty file for use as a text-based initialization parameter file, and save it in the operating system-specific default location.

  b. In the parameter file, set the `DB_NAME` and `DB_DOMAIN` initialization parameters. These are the only required parameters.

     Setting the `DB_DOMAIN` parameter enables you to connect to the default database service when you connect with a net service name.

  c. If the auxiliary instance is to be a CDB, then set the following parameter:

     ```
     ENABLE_PLUGGABLE_DATABASE=TRUE
     ```

2. Set the various location parameters such as `CONTROL_FILES` and `DB_RECOVERY_FILE_DEST`.
3. If necessary, set other initialization parameters like those needed for Oracle Real Application Clusters.
4. Set the required environment variables, such as `ORACLE_HOME` and `ORACLE_SID`.
5. (Optional) Set initialization parameters that specify the location of the duplicate database files if one of the following conditions is satisfied:
   - The source host and the destination host are the same (duplication to the local host).
   - The duplicate database uses a directory structure that is different from that of the source host to store database files.

   Depending on the technique used to specify alternate names for duplicate database files, include one or more of the following parameters in the initialization parameter file: `CONTROL_FILES`, `DB_FILE_NAME_CONVERT`, `LOG_FILE_NAME_CONVERT`, `DB_CREATE_FILE_DEST`, `DB_CREATE_ONLINE_FILE_DEST_`*n*, and `RECOVERY_FILE_DEST`.

   > **Note:**
   >
   > It is recommended that you verify that all paths specified are accessible to the destination host and to the server session of the auxiliary instance.

   See *"Methods of Generating Database File Names for the Duplicate Database"* in *Oracle Database Backup and Recovery User's Guide*.
6. Start SQL*Plus and connect to the auxiliary instance as a user with `SYSDBA` or `SYSBACKUP` privileges. Start the auxiliary instance in `NOMOUNT` mode. If the file is in the default location, no `PFILE` parameter is required on the `STARTUP` command.

   ```
   SQL> STARTUP NOMOUNT;
   ```

Example 4-1 Sample Initialization Parameter File for the Auxiliary Instance

```
DB_NAME=dupdb
CONTROL_FILES=(/dup/oracle/oradata/prod/control01.ctl,
               dup/oracle/oradata/prod/control02.ctl)
DB_FILE_NAME_CONVERT=(/oracle/oradata/prod/,/dup/oracle/oradata/prod/)
LOG_FILE_NAME_CONVERT=(/oracle/oradata/prod/redo,/dup/oracle/oradata/prod/redo)
```

Copying the Server Parameter File from the Source Database

If the source database uses a server parameter file, then including the `SPFILE` option in the `DUPLICATE` command directs RMAN to use the server parameter file from the source database for the auxiliary instance.

For backup-based duplication, the server parameter file is restored from backups. For active database duplication, the server parameter file is copied from the source database to the auxiliary instance.

When the source database uses a text-based initialization parameter file, use the `PFILE` clause in the `DUPLICATE` command to copy the source database's initialization parameter file to the auxiliary instance.

You can modify the values that were copied or restored from the server parameter file of the source database by using the `PARAMETER_VALUE_CONVERT` option of `SPFILE` or the `SET` clause of the `DUPLICATE`. For example, you can use the `SET` clause to change the value of the `DB_FILE_NAME_CONVERT` parameter in the auxiliary instance's server parameter file.

If the source database does not use a server parameter file or RMAN cannot restore a backup of the server parameter file, then you must manually create a text-based initialization parameter file, as described in Steps to Create an Initialization Parameter File for the Auxiliary Instance.

Creating a Password File for the Auxiliary Instance

Connections to the auxiliary instance can be established by using operating system authentication or password file authentication. For backup-based duplication, you can either create a password file or use operating system authentication to connect to the auxiliary instance. For active database duplication, you must use password file authentication.

To connect to a database by using password file authentication, you must create a password file for the database. When duplicating to a remote host, setting up a password file is mandatory. The default location for the password file is *`$ORACLE_BASE`*`\database` on Windows and *`$ORACLE_BASE`*`/dbs` on Linux and UNIX.

> **Note:**
>
> When you create a standby database by using RMAN duplication, password files are always copied. In all other cases, password files are copied only if you specify the `PASSWORD FILE` option in the `DUPLICATE` command.

Use one of the following options to create a password file for the auxiliary instance on the destination host:

- Use operating system-specific utilities to copy the source database password file to the destination host and then rename it to match the auxiliary instance name. This is applicable only if the source and destination hosts are on the same platform.
- Create the password file manually. Ensure that the password for the `SYSDBA` and `SYSBACKUP` users are the same in the source database and auxiliary instance.
- Create the password file with the `orapwd` utility. The `SYSBACKUP` option creates a `SYSBACKUP` entry in the new password file.

  The following example creates a password file in the 12.2 format names `orapworcl` that is located in the default location in an operating system file:

  ```
  orapwd FILE='/u01/oracle/dbs/orapworcl' FORMAT=12.2
  ```

- Specify the `PASSWORD FILE` option on the `DUPLICATE... FROM ACTIVE DATABASE` command.

  RMAN copies the source database password file to the destination host and overwrites any existing password file for the auxiliary instance. This technique is useful if the source database password file has multiple passwords to make available on the duplicate database.

  When you use active database duplication, the password file must contain at least two passwords, for the `SYS` user and the `SYSBACKUP` user. These passwords must match the passwords in the source database.

> **Note:**
>
> If you create a standby database with the `FROM ACTIVE DATABASE` option, then RMAN always copies the password file to the standby host.

> **See Also:**
>
> *Oracle Database Administrator's Guide*

Creating Directories for the Duplicate Database

On the destination host, you must create the directories that RMAN uses to store the duplicate database files on the destination host.

This includes the directories that store the data files, control files, online redo log files, and temp files.

Use the `NOFILENAMECHECK` clause to indicate that RMAN must not display an error when the names of the database files are the same in the source and duplicate database.

Using the Same Names for Database Files in the Source Database and Duplicate Database

Certain conditions must be met to use the same names for files in the source and duplicate database.

The simplest duplication strategy is to configure the duplicate database to use the same directory structure and file names as the source database. You can use the same directory structure and names only when duplicating to a remote host.

Using the same directory structure and file names means that your environment meets the following requirements:

- If the source database uses ASM disk groups, then the duplicate database must use ASM disk groups with the same names.
- If the source database files are Oracle Managed Files, then the auxiliary instance must set the `DB_CREATE_FILE_DEST` parameter to the same directory location as the source database. Although the directories are the same on the source and destination hosts, Oracle Database chooses the relative names for the duplicate files.
- If the names of the database files in the source database contain a path, then this path name must be the same in the duplicate database.
- For Oracle Real Application Clusters (RAC) environments, use the same value for the `ORACLE_SID` parameter of the source and destination databases.

When you configure your environment as suggested, no additional configuration is required to name the duplicate files.

Establishing Oracle Net Connectivity Between the Source Database and Auxiliary Instance

You must be able to establish a connection between the source database and auxiliary instance for certain forms of duplication.

If any of the following conditions is true, the auxiliary instance must be available through Oracle Net Services:

- The RMAN client is run from a host other than the destination host
- The duplication technique chosen is active database duplication
- The destination host is different from the source host

To perform active database duplication, you must connect to the auxiliary instance with `SYSDBA` or `SYSBACKUP` privilege and by using a net service name. The source database to which RMAN is connected as `TARGET` uses this net service name to connect directly to the auxiliary database instance.

To establish Oracle Net connectivity and set up a static listener:

- Follow the instructions in *Oracle Database Administrator's Guide* to configure a client for connection to a database and add static service information for the listener.

Example 4-2 Example: Establishing Oracle Net Connectivity Between the Source Database and Auxiliary Instance

Assume that the `DB_NAME` of the source database is `src` and the source host is `src.example.com`. The `DB_NAME` of the auxiliary instance is `dup` and the auxiliary instance is created on the host `dup.example.com`.

Use the following steps to establish Oracle Net connectivity between the source database and the auxiliary instance:

1. In the `tnanames.ora` file of the source database, add the following entry that corresponds to the duplicate database:

   ```
   dupdb = (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dup.example.com)
   (PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dup)))
   ```

2. On the destination host, create the `tnsnames.ora` file in the $*ORACLE_HOME*`/admin/network` folder. Add the following entry that corresponds to the source database.

   ```
   srcdb = (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=src.example.com)
   (PORT=1521))(CONNECT_DATA=(SERVICE_NAME=src)))
   ```

Configuring the Network Between Source and Target Oracle Databases

In preparation for duplicating your Oracle Database, configure the network between your source and target Oracle Database instances.

- Adding Static Service to Listener
  Insert a static entry for the standby database in the `listener.ora` file, and save the file.
- Configuring SQL*Net to Prepare the Auxiliary Instance on the Destination Host
  Add entries for primary and standby databases in `tnsnames.ora`, and then save the file.
- Checking SQL*Net Configuration

Adding Static Service to Listener

Insert a static entry for the standby database in the `listener.ora` file, and save the file.

Example 4-3 Adding Static Service to Listener

```
[oracle @ ora12c-dup admin] $ cat listener.ora
 # listener.ora Network Configuration File:
 /u01/app/oracle/product/12.2.0/dbhome/network/admin/listener.ora
 # Generated by Oracle configuration tools.
  SID_LIST_LISTENER    =
 (SID_LIST =
 (SID_DESC =
 (GLOBAL_DBNAME = dupdb )
 (ORACLE_HOME = /u01/app/oracle/product/12.2.0/dbhome)
```

```
 (SID_NAME = dupdb )
 )
 )
 LISTENER =
 (DESCRIPTION_LIST =
 (DESCRIPTION =
 (ADDRESS = (PROTOCOL = TCP) (HOST = ora12c-dup.localdomain) (PORT =
1521))
 )
 (DESCRIPTION =
 (ADDRESS = (PROTOCOL = IPC) (KEY = EXTPROC1521))
 )
 )
 ADR_BASE_LISTENER = / u01 / app / oracle
[oracle @ oel62-ora12c-dup admin] $ lsnrctl status
```

Configuring SQL*Net to Prepare the Auxiliary Instance on the Destination Host

Add entries for primary and standby databases in `tnsnames.ora`, and then save the file.

Example 4-4 Adding Net Services

```
[oracle @ ora12c-dup ~] $ cd $ ORACLE_HOME / network / admin
[oracle @ ora12c-dup admin] $ cat tnsnames.ora
 # tnsnames.ora Network Configuration File: /u01/app/oracle/product/12.2.0/
dbhome/network/admin/tnsnames.ora
 # Generated by Oracle configuration tools.
 DUPDB =
 (DESCRIPTION =
 (ADDRESS_LIST =
 (ADDRESS = (PROTOCOL = TCP) (HOST = ora12c-dup) (PORT = 1521))
 )
 (CONNECT_DATA =
 (SERVICE_NAME = dupdb )
 )
 )
 PRMDB =
 (DESCRIPTION =
 (ADDRESS = (PROTOCOL = TCP) (HOST = ora12c-prm.localdomain) (PORT =
1521))
 (CONNECT_DATA =
 (SERVER = DEDICATED)
 (SERVICE_NAME = prmdb )
 )
 )
```

Checking SQL*Net Configuration

Run the `tnsping` command from the source and destination servers.

```
[oracle @ ora12c-prm admin] $ tnsping standby-db-unique-name
[oracle @ ora12c-dup admin] $ tnsping primary-db-unique-name
```

Oracle recommends that you connect remotely as `sysdba` to ensure that the communication is established properly.

Start SQL*Plus using a command in the following format:

```
sqlplus {username | /} [as sysdba]
```

For example:

```
$ sqlplus / AS SYSDBA
Enter password: password
```

Alternatively, to start SQL*Plus connected to a database other than the default, enter the SQL*Plus command in the format:

```
$> sqlplus username/password@connect_identifier
```

Starting the Auxiliary Instance

The initialization parameter file that you create is used to start the auxiliary instance.

RMAN shuts down and restarts the auxiliary instance as part of the duplication. Hence, it is a good idea to create a server-side initialization parameter file for the auxiliary instance in the default location. If you do not have a server-side initialization parameter file in the default location, then you must specify the client-side initialization parameter file with the `PFILE` parameter on the `DUPLICATE` command.

> **Note:**
>
> Because the auxiliary instance does not yet have a control file, you can only start the instance in `NOMOUNT` mode. Do not create a control file or try to mount or open the auxiliary instance.

To start the auxiliary instance:

1. Start RMAN.

   ```
   % rman
   ```

2. Connect to the auxiliary instance as a user with the `SYSDBA` or `SYSBACKUP` privilege. The following example uses password file authentication to connect to the auxiliary instance.

   ```
   RMAN> CONNECT SYS@dupdb AS SYSDBA;
   ```

The following example uses operating system authentication to connect to the auxiliary instance by using the SYSBACKUP privilege.

```
RMAN> CONNECT / AS SYSBACKUP;
```

3. Start the auxiliary instance in NOMOUNT mode.

```
RMAN > STARTUP FORCE NOMOUNT;
```

Making the Oracle Keystore Available to the Destination Host

If transparent encryption is configured on the source database, then you must ensure that the Oracle software keystore from the source database is available to the auxiliary instance. Manually copy the keystore from the source database to the destination host.

The Oracle software keystore contains the TDE master key used to:

- decrypt encrypted backups when performing backup-based duplication.
- decrypt database or tablespace data when performing active database duplication of TDE-encrypted databases or tablespaces.

The following are the requirements for the keystore at the duplicate database:

- The keystore must be in the default location, or in the location indicated by the sqlnet.ora file.
- Permissions on the Oracle keystore file must be set so that the database can access the file.
- During duplication, the auxiliary instance is restarted thereby causing the Oracle software keystore to become unavailable. To ensure that the auxiliary instance has access to the keystore, set the ENCRYPTION_WALLET_LOCATION parameter in the sqlnet.ora file such that it points to the keystore location.
- With Oracle Real Application Clusters (Oracle RAC), register the auxiliary instance statically with an Oracle Grid Infrastructure listener and use the ENVS parameter in the sqlnet.ora file of the Oracle Grid home to specify environment variables that set the keystore location and the unique name of the database.

  The following example sets the ENVS parameter in sqlnet.ora to specify the keystore location and unique database name:

  ```
   (ENVS="ORACLE_UNQNAME=cdbrptl,
  ENCRYPTION_WALLET_LOCATION=(SOURCE=(METHOD=FILE)
  (METHOD_DATA=(DIRECTORY=/etc/ORACLE/WALLETS/cdbrpt1)))")
  ```

- If the source database uses a password-based software keystore (not an auto-login software keystore), then you must provide the keystore password before you begin the duplication.

  Use the SET command with the DECRYPTION WALLET OPEN IDENTIFIED BY clause to specify the password that must be used to open the keystore.

The following command specifies the password used to open the keystore (where *password* is a placeholder for the actual password that you enter):

```
SET DECRYPTION WALLET OPEN IDENTIFIED BY password;
```

See Also:

- *Oracle Database Advanced Security Guide* for information about specifying the Oracle keystore location in `sqlnet.ora`
- *Oracle Database Advanced Security Guide* for information about the default Oracle keystore location
- *Oracle Database Advanced Security Guide* for information about converting a standard Oracle keystore to an auto-login keystore
- *Oracle Database Backup and Recovery Reference* for information about the `SET` command

Starting RMAN and Connecting to Databases

You must start the RMAN client and connect to the database instances as required by the chosen duplication technique. The RMAN client can be located on any host so long as it can connect to the necessary databases over the network.

To start RMAN and connect to the target and auxiliary instances:

1. Start the RMAN client on any host that can connect to the necessary database instances.

   For example, enter the following command at the operating system prompt on the destination host:

   ```
   % rman
   ```

2. At the RMAN prompt, run `CONNECT` commands for the database instances that are required for your duplication technique.

   Note:

   When you duplicate a whole CDB or one or more PDBs, connect to the root of both instances.

   - For active database duplication using image copies, you must connect to the source database as `TARGET` and to the auxiliary instance as `AUXILIARY`. You must supply the net service name to connect to the `AUXILIARY` instance. A recovery catalog connection is optional. On both instances, the password for the user performing the duplication must be the same. Any user with a `SYSDBA` or `SYSBACKUP` privilege can perform duplication.

- For active database duplication using backup sets, you must connect to the source database as TARGET by using a net service name. The auxiliary instance uses this net service name to connect to the source database and retrieve the backup sets that are required for the duplication. Connect to the auxiliary instance as AUXILIARY. If you are connecting to the auxiliary instance remotely or intend to use the PASSWORD FILE option of the DUPLICATE command, then connect to the auxiliary instance with a net service name. On both instances, the password for the user performing the duplication must be the same. Any user with a SYSDBA or SYSBACKUP privilege can perform duplication. A recovery catalog connection is optional.
- For backup-based duplication *without* a target connection, you must connect to the auxiliary instance as AUXILIARY and the recovery catalog as CATALOG.
- For backup-based duplication *with* a target connection, you must connect to the source database as TARGET and the auxiliary instance as AUXILIARY. A recovery catalog is optional.
- For backup-based duplication without target and recovery catalog connections, you must connect to the auxiliary instance as AUXILIARY.

> **Note:**
>
> You cannot connect as TARGET to a standby database.

Example: Connecting to the Required Databases When Performing Active Database Duplication

In this example, a connection is established to the source database and the auxiliary instance using net service names. The Net Service name of the source database is srcdb and that of the auxiliary instance is dupdb.

To connect to required databases from the destination host:

1. Start the RMAN client on the destination host.

   ```
   % rman
   ```

2. Connect to the source database as TARGET.

   ```
   RMAN> CONNECT TARGET sys@srcdb;
   ```

   Enter the password for the SYS user on the source database when prompted.

3. Connect to the auxiliary instance as AUXILIARY.

   ```
   RMAN> CONNECT AUXILIARY sys@dupdb;
   ```

   Enter the password for the SYS user on the auxiliary instance when prompted.

5
Duplicating the Primary Oracle Database

Complete the steps to duplicate the entire primary Oracle Database to the auxiliary Oracle Database instance on the destination server.

- Verifying the Environment
  Complete the steps before you begin duplication.
- Running the RMAN DUPLICATE Command
  Run the DUPLICATE command on the destination server.
- Postmigration Verification
  Verify if the database has been restored and recovered on the destination server.

Verifying the Environment

Complete the steps before you begin duplication.

1. Ensure that the source database host is reachable from the destination host. You must be able to SSH between the two hosts.
2. On the destination host, use the TNSPING utility to verify the listener port on the source host works fine.

   For example:

   ```
   tnsping source host:1521
   ```

3. On the destination host, use Easy Connect to verify the connection to the source database:

   ```
   host:port/servicename
   ```

   For example:

   ```
   sqlplus system@ip-address:1521/proddb
   ```

   Ensure that the connection string does not exceed 64 characters.
4. Copy the required `sqlpatch` files (for rollback) from the source database home to the target database.
5. Ensure that at least one archivelog has been created on the source database; otherwise, RMAN duplication fails with an error.
6. If the source database uses wallets, then back up the password-based wallet and copy it to the standard location on the destination host.

   For example:

   ```
   /opt/oracle/dcs/commonstore/wallets/tde/db_unique_name/
   ```

7. Make sure the compatibility parameters in the source database are set to at least 11.2.0.4.0 for an 11.2.0.4 database and at least 12.1.0.2.0 for a 12.1.0.2 database.

Running the RMAN DUPLICATE Command

Run the DUPLICATE command on the destination server.

For example:

```
DUPLICATE DATABASE TO dupdb
FOR STANDBY
FROM ACTIVE DATABASE
PASSWORD FILE
SPFILE PARAMETER_VALUE_CONVERT='/app/dbhome1','/app/db_home2'
SET db_file_name_convert='/app/dbhome1/dbs','/app/db_home2/database/dbs'
SET log_file_name_convert='/app/dbhome1/log','/app/db_home2/logfiles'
NOFILENAMECHECK;
```

Postmigration Verification

Verify if the database has been restored and recovered on the destination server.

For example:

`V$DATABASE` displays information about the database from the control file.

```
[oracle @ ora18c-dup admin] $ sqlplus

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

`V$INSTANCE` displays the state of the current instance.

```
SQL> select instance_name, host_name from v$instance;
```

`V$DATAFILE` displays information about the datafiles from the control file.

`V$CONTROLFILE` displays the names of the control files.

`V$LOGFILE` contains information about redo log files.

`V$TEMPFILE` displays temp file information.

```
SQL> select name from v$datafile
union
select name from v$controlfile
union
select member from v$logfile
union
select name from v$tempfile;
```

`V$VERSION` displays the version number of Oracle Database.

```
SQL> select banner from v$version;
```

6

Refresh and Switchover to the Physical Standby Oracle Database

Refresh the physical standby Oracle Database with the changes made to the primary Oracle Database, and then switch roles.

- Steps to Refresh a Physical Standby Database with Changes Made to the Primary Database
  Use the `RECOVER STANDBY DATABASE` command with the `FROM SERVICE` clause to refresh a physical standby database with changes that were made to the primary database.
- Performing a Switchover to a Physical Standby Database
  These steps describe how to perform a switchover to a physical standby database.

Steps to Refresh a Physical Standby Database with Changes Made to the Primary Database

Use the `RECOVER STANDBY DATABASE` command with the `FROM SERVICE` clause to refresh a physical standby database with changes that were made to the primary database.

This example assumes that the `DB_UNIQUE_NAME` of the primary database is `MAIN` and its net service name is `primary_db`. The `DB_UNIQUE_NAME` of the standby database is `STANDBY` and its net service name is `standby_db`.

To refresh the physical standby database with changes made to the primary database:

1. Ensure that the following prerequisites are met:
   - Oracle Net connectivity is established between the physical standby database and the primary database.

     You can do this by adding an entry corresponding to the primary database in the `tnsnames.ora` file of the physical standby database.
   - The password files on the primary database and the physical standby database are the same.
   - The `COMPATIBLE` parameter in the initialization parameter file of the primary database and physical standby database is set to 12.0.
2. Start RMAN and connect as target to the physical standby database. It is recommended that you also connect to a recovery catalog.

   The following commands connect as `TARGET` to the physical standby database and as `CATALOG` to the recovery catalog. The connection to the physical standby is established using the `sbu` user, who has been granted `SYSBACKUP` privilege. The

net service name of the physical standby database is `standby_db` and that of the recovery catalog is `catdb`.

```
CONNECT TARGET "sbu@standby_db AS SYSBACKUP";
CONNECT CATALOG rman@catdb;
```

3. Roll forward the physical standby database using the `RECOVER STANDBY DATABASE` command with the `FROM SERVICE` clause.

   The `FROM SERVICE` clause specifies the service name of the primary database using which the physical standby must be rolled forward. The standby database is restarted after the roll forward operation.

   The following example rolls forward the physical standby database using the primary database whose service name is `primary_db`.

   ```
   RECOVER STANDBY DATABASE FROM SERVICE primary_db;
   ```

4. (For Active Data Guard only) Perform the following steps to recover redo data and open the physical standby database in read-only mode:

   ```
   ALTER DATABASE RECOVER MANAGED STANDBY DATABASE UNTIL CONSISTENT;
   ALTER DATABASE OPEN READ ONLY;
   ```

5. Start the managed recovery processes on the physical standby database.

   The following command starts the managed recovery process:

   ```
   ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
   ```

   When using Data Guard Broker, use the following command to start the managed recovery process:

   ```
   DGMGRL> edit database standby_db set state='APPLY-ON';
   ```

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for information about establishing Oracle Net connectivity

Performing a Switchover to a Physical Standby Database

These steps describe how to perform a switchover to a physical standby database.

> **Note:**
>
> If there is a far sync instance (or a combination of preferred and alternate far sync instances) connecting the primary and standby databases, then the procedure to switchover to the standby is the same as described in this topic. Whether the far sync instances are available or unavailable does not affect switchover. During switchover, the primary and standby must be able to communicate directly with each other and perform the switchover role transition steps oblivious of the far sync instances. See “Using Far Sync Instances” in *Oracle Data Guard Concepts and Administration* for examples of how to set up such configurations correctly so that the far sync instances can service the new roles of the two databases after switchover.

1. Verify that the target standby database is ready for switchover.

   The new switchover statement has a `VERIFY` option that results in checks being performed of many conditions required for switchover. Some of the items checked are: whether Redo Apply is running on the switchover target; whether the release version of the switchover target is 12.1 or later; whether the switchover target is synchronized; and whether it has MRP running.

   Suppose the primary database has a `DB_UNIQUE_NAME` of `BOSTON` and the switchover target standby database has a `DB_UNIQUE_NAME` of `CHICAGO`. On the primary database `BOSTON`, issue the following SQL statement to verify that the switchover target, `CHICAGO`, is ready for switchover:

   ```
   SQL> ALTER DATABASE SWITCHOVER TO CHICAGO VERIFY;
   ERROR at line 1:
   ORA-16470: Redo Apply is not running on switchover target
   ```

   If this operation had been successful, a `Database Altered` message would have been returned but in this example an `ORA-16470` error was returned. This error means that the switchover target `CHICAGO` is not ready for switchover. Redo Apply must be started before the switchover operation.

   After Redo Apply is started, issue the following statement again:

   ```
   SQL> ALTER DATABASE SWITCHOVER TO CHICAGO VERIFY;
   ERROR at line 1:
   ORA-16475: succeeded with warnings, check alert log for more details
   ```

   The switchover target, `CHICAGO`, is ready for switchover. However, the warnings indicated by the `ORA-16475` error may affect switchover performance. The alert log contains messages similar to the following:

   ```
   SWITCHOVER VERIFY WARNING: switchover target has dirty online redo logfiles that
   require clearing. It takes time to clear online redo logfiles. This may slow
   down switchover process.
   ```

You can fix the problems or if switchover performance is not important, those warnings can be ignored. After making any fixes you determine are necessary, issue the following SQL statement again:

```
SQL> ALTER DATABASE SWITCHOVER TO CHICAGO VERIFY;
Database altered.
```

The switchover target, CHICAGO, is now ready for switchover.

2. Initiate the switchover on the primary database, BOSTON, by issuing the following SQL statement:

```
SQL> ALTER DATABASE SWITCHOVER TO CHICAGO;
Database altered.
```

If this statement completes without any errors, proceed to Step 3.

If an error occurs, mount the old primary database (BOSTON) and the old standby database (CHICAGO). On both databases, query DATABASE_ROLE from V$DATABASE. There are three possible combinations of database roles for BOSTON and CHICAGO. The following table describes these combinations and provides the likely cause and a high level remedial action for each situation. For details on specific error situations, see “Troubleshooting Oracle Data Guard” in *Oracle Data Guard Concepts and Administration*.

Value of DATABASE_ROLE column in V$DATABASE	Cause and Remedial Action
BOSTON database is primary, CHICAGO database is standby	Cause: The BOSTON database failed to convert to a standby database role. Action: See the alert log for details on the error that prevented BOSTON from switching to a standby role, take the necessary actions to fix the error, reopen one of the nodes of BOSTON if necessary, and repeat the switchover process from Step 1.

Value of DATABASE_ROLE column in V$DATABASE	Cause and Remedial Action

BOSTON database is standby, CHICAGO database is standby	Cause: The CHICAGO database failed to convert to a primary database role.
	Action: Issue the following SQL statement to convert either BOSTON or CHICAGO to a primary database:
	`SQL> ALTER DATABASE SWITCHOVER TO target_db_name FORCE;`
	For example:
	• On the CHICAGO database, issue the following SQL statement to convert it to a primary database:
	`ALTER DATABASE SWITCHOVER TO CHICAGO FORCE;`
	• On the BOSTON database, issue the following SQL statement to convert it to a primary database:
	`ALTER DATABASE SWITCHOVER TO BOSTON FORCE;`
	If the SQL statement fails with an ORA-16473 error, then you must start Redo Apply before reissuing the command.
	Restart Redo Apply as follows:
	`SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT;`
	Reissue the switchover command as follows:
	`SQL> ALTER DATABASE SWTICHOVER TO BOSTON FORCE;`
	`Database altered.`

BOSTON database is standby, CHICAGO database is primary	Cause: The BOSTON and CHICAGO databases have successfully switched to their new roles, but there was an error communicating the final success status back to BOSTON. Action: Continue to Step 3 to finish the switchover operation.

3. Issue the following SQL statement on the new primary database, CHICAGO, to open it.

   ```
   SQL> ALTER DATABASE OPEN;
   ```

4. Issue the following SQL statement to mount the new physical standby database, BOSTON:

   ```
   SQL> STARTUP MOUNT;
   ```

   Or, if BOSTON is an Oracle Active Data Guard physical standby database, then issue the following SQL statement to open it read only:

   ```
   SQL> STARTUP;
   ```

5. Start Redo Apply on the new physical standby database. For example:

   ```
   SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
   ```

7

Checking Compatibility Before Upgrading Oracle Database

Check the Oracle Database server upgrade compatibility matrix before upgrading the Oracle Database.

- Checking the Compatibility Level of Oracle Database
  Use this SQL query to check that the compatibility level of your database corresponds to the value of the `COMPATIBLE` initialization parameter:
- Values for the COMPATIBLE Initialization Parameter in Oracle Database
  Review to find the default, minimum, and maximum values for the `COMPATIBLE` initialization parameter.

Checking the Compatibility Level of Oracle Database

Use this SQL query to check that the compatibility level of your database corresponds to the value of the `COMPATIBLE` initialization parameter:

```
SQL> SELECT name, value FROM v$parameter
         WHERE name = 'compatible';
```

Values for the COMPATIBLE Initialization Parameter in Oracle Database

Review to find the default, minimum, and maximum values for the `COMPATIBLE` initialization parameter.

The following table lists the default, minimum, and maximum values for `COMPATIBLE` in Oracle Database 18c, and in each release supported for upgrading to Oracle Database 18c:

Table 7-1 The COMPATIBLE Initialization Parameter

Oracle Database Release	Default Value	Minimum Value	Maximum Value
Oracle Database 18c	`18.0.0`	`11.2.0`	The COMPATIBLE parameter should not be changed for an RU or an RUR, either for CDB or Non-CDB instances.
Oracle Database 12*c* Release 2 (12.2)	`12.2.0`	`11.2.0`	`12.2.0`
Oracle Database 12*c* Release 1 (12.1)	`12.0.0`	`11.0.0`	`12.1.0`

Table 7-1 (Cont.) The COMPATIBLE Initialization Parameter

Oracle Database Release	Default Value	Minimum Value	Maximum Value
Oracle Database 11*g* Release 2 (11.2)	`11.2.0`	`10.0.0`	`11.2.0`

8

Preparing to Upgrade Oracle Database

Before you upgrade your database, Oracle recommends that you review the new features and determine the best upgrade path and method to use, and carry out procedures to prepare your database for upgrade. Oracle strongly recommends that you test the upgrade process and prepare a backup strategy.

- Installing Oracle Software in a New Oracle Home
  Choose a new location for Oracle Home and then install the new Oracle Database Software for single-instance.
- The Graphical User Interface Method for Upgrading Oracle Database
  Database Upgrade Assistant (DBUA) interactively steps you through the upgrade process and configures the database for the new Oracle Database release.
- The Manual, Command-Line Method for Upgrading Oracle Database
  Manual upgrades give you finer control over the upgrade process.
- Prepare a Backup Strategy Before Upgrading Oracle Database
  You must design and carry out an appropriate backup strategy to ensure a successful upgrade.
- Database Preparation Tasks to Complete Before Starting Oracle Database Upgrades
  Ensure that you have completed these database preparation tasks before starting an Oracle Database upgrade.
- Using the Pre-Upgrade Information Tool for Oracle Database
  Review these topics to understand and to use the Pre-Upgrade information tool (`preupgrade.jar`).
- Enabling Oracle Database Vault After Upgrading Oracle Database
  Depending on your target database release, you can be required to disable Oracle Database Vault to complete an Oracle Database upgrade.

Installing Oracle Software in a New Oracle Home

Choose a new location for Oracle Home and then install the new Oracle Database Software for single-instance.

- Choose a New Location for Oracle Home when Upgrading
  You must choose a location for Oracle home for the new release of Oracle Database that is separate from the Oracle home of your current release.
- Installing the New Oracle Database Software for Single Instance
  Use this procedure overview to assist you to install the software for the new Oracle Database release for a single instance deployment.

Choose a New Location for Oracle Home when Upgrading

You must choose a location for Oracle home for the new release of Oracle Database that is separate from the Oracle home of your current release.

Using separate installation locations enables you to keep your existing Oracle software installed along with the new Oracle software. This method enables you to test the upgrade process on a test database before replacing your production environment entirely.

Installing the New Oracle Database Software for Single Instance

Use this procedure overview to assist you to install the software for the new Oracle Database release for a single instance deployment.

> **Note:**
>
> You cannot upgrade a database using Database Upgrade Assistant (DBUA) when the source and target Oracle homes are owned by different users. Attempting to do so returns error `PRKH-1014`. Either ensure that the source and target databases have the same owner, or perform a manual upgrade.

To install the new Oracle Database software for this release:

1. Follow the instructions in your Oracle operating system-specific documentation to prepare for installation of Oracle Database software.
2. Start Oracle Universal Installer, and select a software-only installation.

   When installation of Oracle Database software has completed successfully, click **Exit** to close Oracle Universal Installer.
3. Run the Pre-Upgrade Information Tool. The tool enables you to check the types of items that DBUA checks. The tool identifies issues, and can help you fix some issues that it finds.

   By default, the tool is in the location *`New_release_Oracle_home`*`/rdbms/admin/preupgrade.jar`.

   If you use Oracle Label Security, Oracle Database Vault, or both, then select **Enterprise Edition** on the Select Database Edition page, click **Select Options**, and enable one or both components from the components list.

The Graphical User Interface Method for Upgrading Oracle Database

Database Upgrade Assistant (DBUA) interactively steps you through the upgrade process and configures the database for the new Oracle Database release.

DBUA starts the Pre-Upgrade Information Tool, which fixes some configuration settings to the values required for the upgrade. For example, the tool can change initialization parameters to values required for the upgrade. The tool also provides you with a list of items that you can fix manually before you continue with the upgrade.

The Manual, Command-Line Method for Upgrading Oracle Database

Manual upgrades give you finer control over the upgrade process.

A manual upgrade consists of running SQL scripts and utilities from a command line to upgrade a database to the new Oracle Database release.

Before the Upgrade

- Analyze the database using the Pre-Upgrade Information Tool.

  The Pre-Upgrade Information Tool is a Java JAR file that is supplied with Oracle Database. When you start the tool, it self-extracts, and then executes SQL scripts.

  The Pre-Upgrade Information Tool displays warnings about possible upgrade issues with the database, and generates fixup scripts for you to use to address some issues. It also displays information about required initialization parameters for the new release of Oracle Database.
- Prepare the new Oracle home.
- Perform a backup of the database.

Depending on the Oracle Database release you upgrade, you can be required to perform more pre-upgrade steps. These steps can include adjusting the parameter file for the upgrade, removing desupported initialization parameters, or adjusting initialization parameters that can cause upgrade problems.

Prepare a Backup Strategy Before Upgrading Oracle Database

You must design and carry out an appropriate backup strategy to ensure a successful upgrade.

To develop a backup strategy, consider the following questions:

- How long can the production database remain inoperable before business consequences become intolerable?
- What backup strategy is necessary to meet your availability requirements?
- Are backups archived in a safe, offsite location?
- How quickly can backups be restored (including backups in offsite storage)?
- Have recovery procedures been tested successfully?

Your backup strategy should answer all of these questions, and include procedures for successfully backing up and recovering your database. For information about implementing backup strategies using RMAN, review *Oracle Database Backup and Recovery User's Guide*.

Related Topics

- *Oracle Database Backup and Recovery User's Guide*

Database Preparation Tasks to Complete Before Starting Oracle Database Upgrades

Ensure that you have completed these database preparation tasks before starting an Oracle Database upgrade.

- Patch Set Updates and Requirements for Upgrading Oracle Database
  Before starting upgrades, update your new release Oracle Database to the latest Oracle bundle patch, patch set update (BP or PSU), or Release Update (Update), or Release Update Revision (Revision).
- Copying Transparent Encryption Oracle Wallets
  If you use Oracle wallet with Transparent Data Encryption (TDE), then copy the `sqlnet.ora` and wallet file to the new Oracle home.
- Recommendations for Oracle Net Services When Upgrading Oracle Database
  Review these procedures and parameter changes for Oracle Net Services before you upgrade.
- Understanding Password Case Sensitivity and Upgrades
  By default, Oracle Database 12*c* Release 2 (12.2) and later releases are upgraded to an Exclusive Mode. Exclusive Modes do not support case-insensitive password-based authentication.
- Checking for Accounts Using Case-Insensitive Password Version
  Use these procedures to identify if the Oracle Database that you want to upgrade has accounts or configuration parameters that are using a case-insensitive password version.
- Running Upgrades with Read-Only Tablespaces
  Use the Parallel Upgrade Utility with the `-T` option to take schema-based tablespaces offline during upgrade.

Patch Set Updates and Requirements for Upgrading Oracle Database

Before starting upgrades, update your new release Oracle Database to the latest Oracle bundle patch, patch set update (BP or PSU), or Release Update (Update), or Release Update Revision (Revision).

The software for new Oracle Database releases contains a full release that includes all the latest patches and updates for Oracle Database at the time of the release.

Before you start an upgrade or downgrade process, Oracle strongly recommends that you update both your earlier release and your new release Oracle Database. For Oracle Database 12*c* or earlier releases, update to the latest Oracle bundle patch, or patch set update (BP or PSU). For Oracle Database 12*c* release 2 (12.2), Oracle Database 18c, or later releases, update to the latest quarterly Release Update (Update) or Release Update Revision (Revision).

My Oracle Support provides detailed notes about how you can obtain the latest patches, as well as tools for lifecycle management and automated patching. For example:

- My Oracle Support note 854428.1 contains information about patch sets and updates.

- My Oracle Support note 730365.1 contains an upgrade reference list for most available Oracle Database releases, including download information, patch numbers, and links to other notes.
- My Oracle Support note 2180188.1 contains lists of one-off patches for upgrades, downgrades, and coexistence with previous releases.
- My Oracle Support note 1227443.1 contains a list of Oracle Database PSU/BP/Update/Revision known issues. This note provides information about all known issues notes for Oracle Database, Oracle Grid Infrastructure, and the Oracle JavaVM Component (OJVM).
- My Oracle Support note 2118136.2 contains a download assistant to help you select the updates, revisions, Patch Set Updates (PSU), SPU (CPU), Bundle Patches, Patchsets, and Base Releases that you need for your environment. Oracle highly recommends that you start here.

Related Topics

- My Oracle Support Note 854428.1
- My Oracle Support Note 730365.1
- My Oracle Support Note 2180188.1
- My Oracle Support Note 1227443.1
- My Oracle Support Note 2118136.2

Copying Transparent Encryption Oracle Wallets

If you use Oracle wallet with Transparent Data Encryption (TDE), then copy the `sqlnet.ora` and wallet file to the new Oracle home.

You must copy the `sqlnet.ora` and the wallet file manually before starting the upgrade.

1. Log in as an authorized user.
2. Manually copy the `sqlnet.ora` file, and the wallet file, `ewallet.p12`, to the new release Oracle home.
3. Open the Oracle wallet in mount.

   For example:

```
SQL> STARTUP MOUNT;
SQL> ALTER SYSTEM SET ENCRYPTION WALLET OPEN
```

Recommendations for Oracle Net Services When Upgrading Oracle Database

Review these procedures and parameter changes for Oracle Net Services before you upgrade.

In Oracle Database 12*c*, new underlying net services parameters enable data compression, which reduces the size of the session data unit that is transmitted over a SQL TCP connection.

The following new parameters for the `sqlnet.ora` file specify compression, and the preferred compression scheme:

- `SQLNET.COMPRESSION`
- `SQLNET.COMPRESSION_LEVELS`
- `SQLNET.COMPRESSION_THRESHOLD`

These parameters introduced with Oracle Database 12*c* are not supported in earlier releases. They are only available in Oracle Database 12*c*, and later releases For more information about these `sqlnet.ora` compression parameters, refer to *Oracle Net Services Reference*.

If the Oracle Database that you are upgrading does not have a listener configured, then before you run DBUA, you must run Oracle Net Configuration Assistant (NETCA) to configure the listening protocol address and service information for the new release of Oracle Database, including a `listener.ora` file. You must create a new version of the listener for releases of Oracle Database earlier than release 11.2. The current listener is backward-compatible with earlier Oracle Database releases.

When you upgrade an Oracle RAC database with DBUA, it automatically migrates the listener from your old Oracle home to the new Oracle Grid Infrastructure home. You must administer the listener by using the `lsnrctl` command in the Oracle Grid Infrastructure home. Do not attempt to use the `lsnrctl` commands from Oracle home locations for earlier releases.

Related Topics

- *Oracle Database Net Services Reference*

Understanding Password Case Sensitivity and Upgrades

By default, Oracle Database 12*c* Release 2 (12.2) and later releases are upgraded to an Exclusive Mode. Exclusive Modes do not support case-insensitive password-based authentication.

Accounts that have only the `10G` password version become inaccessible when the server runs in an Exclusive Mode.

In previous Oracle Database releases, you can configure the authentication protocol so that it allows case-insensitive password-based authentication by setting `SEC_CASE_SENSITIVE_LOGON=FALSE`. Starting with Oracle Database 12*c* release 2 (12.2), the default password-based authentication protocol configuration excludes the use of the case-insensitive `10G` password version. By default, the `SQLNET.ORA` parameter `SQLNET.ALLOWED_LOGON_VERSION_SERVER` is set to `12`, which is an Exclusive Mode. When the database is configured in Exclusive Mode, the password-based authentication protocol requires that one of the case-sensitive password versions (`11G` or `12C`) is present for the account being authenticated. This mode excludes the use of the `10G` password version used in earlier releases. After upgrading to Oracle Database 12*c* release 2 and later releases, accounts that have only the case-insensitive `10G` password version become inaccessible. This change occurs because the server runs in an Exclusive Mode by default. When Oracle Database is configured in Exclusive Mode, it cannot use the old `10G` password version to authenticate the client. The server is left with no password version with which to authenticate the client.

For greater security, Oracle recommends that you leave case-sensitive password-based authentication enabled. This setting is the default. However, you can

temporarily disable case-sensitive authentication during the upgrade to new Oracle Database releases. After the upgrade, you can then decide if you want to enable the case-sensitive password-based authentication feature as part of your implementation plan to manage your password versions.

Before upgrading, Oracle recommends that you determine if this change to the default password-based authentication protocol configuration affects you. Perform the following checks:

- Identify if you have accounts that use only `10G` case-insensitive password authentication versions.
- Identify if you have Oracle Database 11*g* release 2 (11.2.0.3) database or earlier clients that have not applied critical patch update `CPUOct2012`, or a later patch update, and have any account that does not have the case-insensitive `10G` password version.
- Ensure that you do not have the deprecated parameter SEC_CASE_SENSITIVE_LOGON set to FALSE. Setting this parameter to FALSE prevents the use of the case-sensitive password versions (the `11G` and `12C` password versions) for authentication.

Options for Accounts Using Case-Insensitive Versions

If you have user accounts that have only the case-insensitive `10G` password version, then you must choose one of the following alternatives:

- Before upgrade, update the password versions for each account that has only the `10G` password version. You can update the password versions by expiring user passwords using the `10G` password version, and requesting that these users log in to their account. When they attempt to log in, the server automatically updates the list of password versions, which includes the case-sensitive password versions.
- Change the setting of the SQLNET.ORA parameter `SQLNET.ALLOWED_LOGON_VERSION_SERVER` to any of the settings that are not Exclusive Mode. For example: `SQLNET.ALLOWED_LOGON_VERSION_SERVER=11`

Related Topics

- *Oracle Database 2 Day DBA*
- *Oracle Database Net Services Reference*
- *Oracle Database Security Guide*

Checking for Accounts Using Case-Insensitive Password Version

Use these procedures to identify if the Oracle Database that you want to upgrade has accounts or configuration parameters that are using a case-insensitive password version.

By default, in Oracle Database 12*c* release 2 (12.2) and later releases, the `10G` password version is not generated or allowed.

If you do not set `SQLNET.ALLOWED_LOGON_VERSION_SERVER` to a permissive authentication protocol that permits case-insensitive versions, and you do not want user accounts authenticated with case-insensitive password versions to be locked out of the database, then you must identify affected accounts, and ensure that they are using case-sensitive password versions.

Example 8-1 Finding User Accounts That Use Case-Insensitive (10G) Version

Log in to SQL*Plus as an administrative user, and enter the following SQL query:

```
SELECT USERNAME,PASSWORD_VERSIONS FROM DBA_USERS;
```

The following result shows password versions for the accounts:

```
USERNAME                       PASSWORD_VERSIONS
------------------------------ -----------------
JONES                          10G 11G 12C
ADAMS                          10G 11G
CLARK                          10G 11G
PRESTON                        11G
BLAKE                          10G
```

In this example, the backgrounds for each user account password verification version in use are different:

- `JONES` was created in Oracle Database `10G`, and the password for `JONES` was reset in Oracle Database `12C` when the setting for the `SQLNET.ALLOWED_LOGON_VERSION_SERVER` parameter was set to `8`. As a result, this password reset created all three versions. `11G` and `12C` use case-sensitive passwords.
- `ADAMS` and `CLARK` were originally created with the `10G` version, and then `11G`, after they were imported from an earlier release. These account passwords were then reset in `11G`, with the deprecated parameter SEC_CASE_SENSITIVE_LOGON set to TRUE.
- The password for `BLAKE` was created with the `10G` version, and the password has not been reset. As a result, user BLAKE continues to use the `10G` password version, which uses a case-insensitive password.

The user `BLAKE` has only the `10G` password version before upgrade:

```
SQL> SELECT USERNAME,PASSWORD_VERSIONS FROM DBA_USERS;

USERNAME PASSWORD_VERSIONS
------------------------------ -----------------
BLAKE 10G
```

If you upgrade to a new Oracle Database release without taking any further action, then this account becomes inaccessible. Ensure that the system is not configured in Exclusive Mode (by setting the `SQLNET.ORA` parameter `SQLNET.ALLOWED_LOGON_VERSION_SERVER` to a more permissive authentication mode) before the upgrade.

Example 8-2 Fixing Accounts with Case-Insensitive Passwords

Complete the following procedure:

1. Use the following SQL query to find the accounts that only have the `10G` password version:

```
select USERNAME
   from DBA_USERS
  where ( PASSWORD_VERSIONS = '10G '
         or PASSWORD_VERSIONS = '10G HTTP ')
    and USERNAME <> 'ANONYMOUS';
```

2. Configure the system so that it is not running in Exclusive Mode by editing the setting of the `SQLNET.ORA` parameter `SQLNET.ALLOWED_LOGON_VERSION_SERVER` to a level appropriate for affected accounts. For example:

   ```
   SQLNET.ALLOWED_LOGON_VERSION_SERVER=11
   ```

   After you make this change, proceed with the upgrade.

3. After the upgrade completes, use the following command syntax to expire the accounts you found in step 1, where *username* is the name of a user returned from the query in step 1:

   ```
   ALTER USER username PASSWORD EXPIRE;
   ```

4. Ask the users for whom you have expired the passwords to log in.

5. When these users log in, they are prompted to reset their passwords. The system internally generates the missing `11G` and `12C` password versions for their account, in addition to the `10G` password version. The `10G` password version is still present, because the system is running in the permissive mode.

6. Ensure that the client software with which users are connecting has the `O5L_NP` capability flag.

   > **Note:**
   >
   > All Oracle Database release 11.2.0.4 and later clients, and all Oracle Database release 12.1 and later clients have the `O5L_NP` capability. Other clients require the `CPUOct2012` patch to acquire the `O5L_NP` capability.
   >
   > The `O5L_NP` capability flag is documented in *Oracle Database Net Services Reference*, in the section on the parameter `SQLNET.ALLOWED_LOGON_VERSION_SERVER`.

7. After all clients have the `O5L_NP` capability, raise the server security back to Exclusive Mode by using the following procedure:

   a. Remove the `SEC_CASE_SENSITIVE_LOGON` setting from the instance initialization file, or set the `SEC_CASE_SENSITIVE_LOGON` instance initialization parameter to `TRUE`. For example:

      ```
      SEC_CASE_SENSITIVE_LOGON = TRUE
      ```

   b. Remove the `SQLNET.ALLOWED_LOGON_VERSION_SERVER` parameter from the server `SQLNET.ORA` file, or set it back to Exclusive Mode by changing the value of `SQLNET.ALLOWED_LOGON_VERSION_SERVER` in the server `SQLNET.ORA` file back to 12. For example:

      ```
      SQLNET.ALLOWED_LOGON_VERSION_SERVER = 12
      ```

8. Use the following SQL query to find the accounts that still have the `10G` password version:

```
select USERNAME
  from DBA_USERS
 where PASSWORD_VERSIONS like '%10G%'
   and USERNAME <> 'ANONYMOUS';
```

9. Use the list of accounts returned from the query in step 8 to expire all the accounts that still have the `10G` password version. Expire the accounts using the following syntax, where *`username`* is a name on the list returned by the query:

   ```
   ALTER USER username PASSWORD EXPIRE;
   ```

10. Request the users whose accounts you expired to log in to their accounts.

    When the users log in, they are prompted to reset their password. The system internally generates only the `11G` and `12C` password versions for their account. Because the system is running in Exclusive Mode, the `10G` password version is no longer generated.

11. Check that the system is running in a secure mode by rerunning the query from step 1. Ensure that no users are found. When the query finds no users, this result means that no `10G` password version remains present in the system.

Example 8-3 Checking for the Presence of SEC_CASE_SENSITIVE_LOGON Set to FALSE

Oracle Database does not prevent the use of the `FALSE` setting for `SEC_CASE_SENSITIVE_LOGON` when the `SQLNET.ALLOWED_LOGON_VERSION_SERVER` parameter is set to `12` or `12a`. This setting can result in all accounts in the upgraded database becoming inaccessible.

```
SQL> SHOW PARAMETER SEC_CASE_SENSITIVE_LOGON

NAME                                 TYPE        VALUE
------------------------------------ -----------
------------------------------
sec_case_sensitive_logon             boolean     FALSE
```

You can change this parameter by using the following command:

```
SQL> ALTER SYSTEM SET SEC_CASE_SENSITIVE_LOGON = TRUE;

System altered.
```

> **Note:**
>
> Unless the value for the parameter `SQLNET.ALLOWED_LOGON_VERSION_SERVER` is changed to a version that is more permissive than `12`, such as `11`, do not set the `SEC_CASE_SENSITIVE_LOGON` parameter to FALSE.

Related Topics

- *Oracle Database Net Services Reference*
- *Oracle Database Security Guide*

Running Upgrades with Read-Only Tablespaces

Use the Parallel Upgrade Utility with the `-T` option to take schema-based tablespaces offline during upgrade.

Oracle Database can read file headers created in earlier releases, so you are not required to do anything to them during the upgrade. The file headers of `READ ONLY` tablespaces are updated when they are changed to `READ WRITE`.

If the upgrade suffers a catastrophic error, so that the upgrade is unable to bring the tablespaces back online, then review the upgrade log files. The log files contain the actual SQL statements required to make the tablespaces available. To bring the tablespaces back online, you must run the SQL statements in the log files for the database, or run the log files for each PDB.

Viewing Tablespace Commands in Upgrade Log Files

If a catastrophic upgrade failure occurs, then you can navigate to the log directory (*`Oracle_base`*`/cfgtoologs/dbua`), and run commands in the log files manually to bring up tablespaces. You can view tablespace commands in the following log files:

- Non-CDB Upgrades: `catupgrd0.log`
- PDB databases: `catupgrd`*`pdbname`*`0.log`, where *`pdbname`* is the name of the PDB that you are upgrading.

At the beginning of each log file, you find SQL statements such as the following, which sets tables to `READ ONLY`:

```
SQL> ALTER TABLESPACE ARGROTBLSPA6 READ ONLY;

Tablespace altered.

SQL> ALTER TABLESPACE ARGROTBLSPB6 READ ONLY;

Tablespace altered.
```

Near the end of each log file, you find SQL statements to reset tables to `READ WRITE`:

```
SQL> ALTER TABLESPACE ARGROTBLSPA6 READ WRITE;

Tablespace altered.

SQL> ALTER TABLESPACE ARGROTBLSPB6 READ WRITE;

Tablespace altered.
```

> **See Also:**
>
> *Oracle Database Administrator's Guide* for information about transporting tablespaces between databases

Using the Pre-Upgrade Information Tool for Oracle Database

Review these topics to understand and to use the Pre-Upgrade information tool (`preupgrade.jar`).

- Setting Up Environment Variables for the Pre-Upgrade Information Tool
  Before you run the Pre-Upgrade Information Tool, in preparation for your non-CDB upgrade, set up the user environment variables for the Oracle user that runs the tool.
- Running the Pre-Upgrade Information Tool
  To check your system and database to see if it is ready for upgrade, use the Pre-Upgrade Information Tool (`preupgrade.jar`)
- Pre-Upgrade Information Tool Warnings and Recommendations for Oracle Database
  Analyze any Pre-Upgrade Information Tool warnings before you upgrade to the new release of Oracle Database. For each item that the tool reports, it provides you with information about how to fix the issue or warning.
- Pre-Upgrade Information Tool Output Example
  In this example, you can see how the Pre-Upgrade Information Tool displays recommended fixes, but does not carry out fixes automatically.

Setting Up Environment Variables for the Pre-Upgrade Information Tool

Before you run the Pre-Upgrade Information Tool, in preparation for your non-CDB upgrade, set up the user environment variables for the Oracle user that runs the tool.

You must set up the user environment variables for the Pre-Upgrade information tool. This example shows how to use shell commands to set up user environment variables to point to an earlier release Oracle home.

In this example, the operating system is Linux or UNIX, the system identifier is `sales01`, and the earlier release Oracle home path is `/u01/app/oracle/product/12.1.0/dbhome_1`

1. Log in as the Oracle installation owner (`oracle`).
2. Set up the user environment variables to point to the earlier release Oracle home that you want to upgrade.

   For example:

   ```
   $ export ORACLE_HOME=/u01/app/oracle/product/12.1.0/dbhome_1
   $ export ORACLE_BASE=/u01/app/oracle
   ```

```
$ export ORACLE_SID=sales01
$ export PATH=.:$ORACLE_HOME/bin:$PATH
```

Running the Pre-Upgrade Information Tool

To check your system and database to see if it is ready for upgrade, use the Pre-Upgrade Information Tool (`preupgrade.jar`)

The Pre-Upgrade Information Tool is in the new release Oracle home, in the file path `ORACLE_HOME/rdbms/admin/preupgrade.jar`. Oracle has configured it with the system checks necessary for the new Oracle Database release. However, the checks that the tool performs are carried out on the earlier release Oracle Database home. Set up the Oracle user environment variables so that they point to the earlier release Oracle home.

Run the Pre-Upgrade Information Tool by using the Java version in your earlier release Oracle home. For multitenant architecture (CDB and PDB) upgrades, open up all the PDBs that you want the tool to analyze before you run the tool.

Set the environment variables for your user account to point to the earlier release ORACLE_HOME, ORACLE_BASE, and ORACLE_SID.

Pre-Upgrade Information Tool Location

The `preupgrade.jar` file is located in the new Oracle home:

New_release_Oracle_home`/rdbms/admin/preupgrade.jar`

You can also copy the `preupgrade.jar` binaries to a path of your choosing. For example:

```
/tmp/preupgrade.jar
```

Pre-Upgrade Information Tool Syntax

```
$Earlier_release_Oracle_home/jdk/bin/java -jar $New_release_Oracle_home
/rdbms/admin/preupgrade.jar [FILE|TERMINAL] [TEXT|XML] [DIR output_dir]
```

Example 8-4 Non-CDB In the Source Oracle Home Example

1. Set your user environment variables to point to the earlier release Oracle home.

   ```
   $ export ORACLE_HOME=/u01/app/oracle/product/12.1.0/dbhome_1
   $ export ORACLE_BASE=/u01/app/oracle
   $ export ORACLE_SID=sales01
   $ export PATH=.:$ORACLE_HOME/bin:$PATH
   ```

2. Run the new release Oracle Database Pre-Upgrade Information Tool on the earlier release Oracle Database server, using the environment settings you have set to the earlier release Oracle home.

   ```
   $ORACLE_HOME/jdk/bin/java -jar /u01/app/oracle/product/18.0.0/dbhome_1/
   rdbms/admin/preupgrade.jar TERMINAL TEXT
   ```

Related Topics

- *Oracle Database Upgrade Guide*

Pre-Upgrade Information Tool Warnings and Recommendations for Oracle Database

Analyze any Pre-Upgrade Information Tool warnings before you upgrade to the new release of Oracle Database. For each item that the tool reports, it provides you with information about how to fix the issue or warning.

For more detailed information , refer to My Oracle Support note 472937.1 for information about installed database components and schemas. Refer to My Oracle Support note 753041.1 for information about diagnosing components with `NON VALID` status.

- Updating Access Control Lists and Network Utility Packages
  Use this procedure to update access control lists (ACLs) and Network Utility Packages.
- Evaluate Dependencies and Add ACLs for Network Utility Packages
  You can receive a warning about network utility package dependencies. Use this procedure to evaluate the dependencies, and provide access by adding the appropriate access control lists (ACLs).

Related Topics

- https://support.oracle.com/rs?type=doc&id=472937.1
- https://support.oracle.com/rs?type=doc&id=753041.1

Updating Access Control Lists and Network Utility Packages

Use this procedure to update access control lists (ACLs) and Network Utility Packages.

Starting with Oracle Database 12*c*, the access control of the UTL packages is implemented using Oracle Database Real Application Security. UTL packages include UTL_TCP, UTL_SMTP, UTL_MAIL, UTL_HTTP, and UTL_INADDR. The access control does not require Oracle XML DB.

1. Ensure that the logged-in user has the `connect` privilege for the host and port specified by `DBMS_LDAP.init`. There is new behavior for the `DBMS_LDAP` PL/SQL package and the `HttpUriType` type. Because of this new behavior, you must create or update access control lists (ACLs) after you upgrade to the new Oracle Database release.

   For example, if your application depends on the `DBMS_LDAP` package, then the error "ORA-24247: network access denied by access control list (ACL)" can occur. To avoid this error, the logged-in user must have the `connect` privilege for the host and port specified by `DBMS_LDAP.init`.
2. If you have any of the following packages installed, then you can be required to reinstall these packages after upgrade:
   - `UTL_TCP`
   - `UTL_SMTP`
   - `UTL_MAIL`

- UTL_HTTP
- UTL_INADDR

Ensure that you have the latest version of these packages for the new Oracle Database release.

> **See Also:**
>
> *Oracle Database Real Application Security Administrator's and Developer's Guide* for information about configuring access control lists

Evaluate Dependencies and Add ACLs for Network Utility Packages

You can receive a warning about network utility package dependencies. Use this procedure to evaluate the dependencies, and provide access by adding the appropriate access control lists (ACLs).

1. Run the Pre-Upgrade Information Tool.
2. Check the output from the Pre-Upgrade Information Tool (`preupgrade.jar`) for warning messages, such as the following example:

```
WARNING: --> Database contains schemas with objects dependent on network
packages.
.... Refer to the Database Upgrade Guide for instructions to configure Network
ACLs.
.... USER WKSYS has dependent objects.
.... USER SYSMAN has dependent objects.
.... USER FLOWS_010600 has dependent objects.
.
```

3. Query the view `DBA_DEPENDENCIES` to obtain more information about the dependencies. For example:

```
SELECT * FROM DBA_DEPENDENCIES
WHERE referenced_name IN
('UTL_TCP','UTL_SMTP','UTL_MAIL','UTL_HTTP','UTL_INADDR','DBMS_LDAP')
  AND owner NOT IN ('SYS','PUBLIC','ORDPLUGINS');
```

4. To ensure that the new access controls are part of your upgrade testing, prepare a post-upgrade script to make the scripts available in your database environment.

   Use the package `DBMS_NETWORK_ACL_ADMIN` to update your database access control lists (ACLs). You use this package to create, assign, and add privileges to the new access controls so that the updated access control packages can work as they did in prior releases. Refer to the example script provided in *Oracle Database Real Application Security Administrator's and Developer's Guide* to see how to use `DBMS_NETWORK_ACL_ADMIN` to update your access control list.

5. After the upgrade, grant specific required privileges. Access is based on the usage in the original database.

Related Topics

- *Oracle Database Real Application Security Administrator's and Developer's Guide*

Pre-Upgrade Information Tool Output Example

In this example, you can see how the Pre-Upgrade Information Tool displays recommended fixes, but does not carry out fixes automatically.

You have control over how and when the fixup scripts are run.

The following example shows the output that is generated and written to `preupgrade.log` by running the Oracle Database 18c Pre-Upgrade Information Tool on a release 12.2.0.1 CDB:

```
$ java -jar preupgrade.jar TEXT TERMINAL
Report generated by Oracle Database Pre-Upgrade Information Tool Version
18.0.0.0.0 on 2018-06-13T15:08:45

Upgrade-To version: 18.0.0.0.0

======================================
Status of the database prior to upgrade
======================================
      Database Name:  CDB2
     Container Name:  CDB$ROOT
       Container ID:  1
            Version:  12.2.0.1.0
         Compatible:  12.2.0
          Blocksize:  8192
           Platform:  Linux x86 64-bit
     Timezone File:  26
  Database log mode:  NOARCHIVELOG
           Readonly:  FALSE
            Edition:  EE

  Oracle Component                       Upgrade Action    Current Status
  ----------------                       --------------    --------------
  Oracle Server                          [to be upgraded]  VALID
  JServer JAVA Virtual Machine           [to be upgraded]  VALID
  Oracle XDK for Java                    [to be upgraded]  VALID
  Real Application Clusters              [to be upgraded]  OPTION OFF
  Oracle Workspace Manager               [to be upgraded]  VALID
  Oracle Label Security                  [to be upgraded]  VALID
  Oracle XML Database                    [to be upgraded]  VALID
  Oracle Java Packages                   [to be upgraded]  VALID

==============
BEFORE UPGRADE
==============

  REQUIRED ACTIONS
  ================
  None

  RECOMMENDED ACTIONS
  ===================
  1.  Run 12.2.0.1.0 $ORACLE_HOME/rdbms/admin/utlrp.sql to recompile
```

```
invalid
      objects.  You can view the individual invalid objects with

        SET SERVEROUTPUT ON;
        EXECUTE DBMS_PREUP.INVALID_OBJECTS;

      3 objects are INVALID.

      There should be no INVALID objects in SYS/SYSTEM or user schemas
before
      database upgrade.

  2.  Review and remove any unnecessary HIDDEN/UNDERSCORE parameters.

      The database contains the following initialization parameters whose
name
      begins with an underscore:

      _exclude_seed_cdb_view

      Remove hidden parameters before database upgrade unless your
application
      vendors and/or Oracle Support state differently.  Changes will need
to be
      made in the pfile/spfile.

  3.  (AUTOFIXUP) Gather stale data dictionary statistics prior to database
      upgrade in off-peak time using:

        EXECUTE DBMS_STATS.GATHER_DICTIONARY_STATS;

      Dictionary statistics do not exist or are stale (not up-to-date).

      Dictionary statistics help the Oracle optimizer find efficient SQL
      execution plans and are essential for proper upgrade timing. Oracle
      recommends gathering dictionary statistics in the last 24 hours
before
      database upgrade.

      For information on managing optimizer statistics, refer to the
12.2.0.1
      Oracle Database SQL Tuning Guide.

  INFORMATION ONLY
  ================
  4.  To help you keep track of your tablespace allocations, the following
      AUTOEXTEND tablespaces are expected to successfully EXTEND during the
      upgrade process.

                                                 Min Size
      Tablespace                    Size         For Upgrade
      ----------                 ----------  -----------
      SYSAUX                        550 MB       616 MB
      SYSTEM                        700 MB      1117 MB
      TEMP                           22 MB       150 MB
```

```
      UNDOTBS1                            315 MB       433 MB

      Minimum tablespace sizes for upgrade are estimates.

  5.  No action needed.

      Using default parallel upgrade options, this CDB with 1 PDBs will
first
      upgrade the CDB$ROOT, and then upgrade at most 1 PDBs at a time
using 2
      parallel processes per PDB.

      The number of PDBs upgraded in parallel and the number of parallel
      processes per PDB can be adjusted as described in Database Upgrade
Guide.

  ORACLE GENERATED FIXUP SCRIPT
  ============================
  All of the issues in database CDB2 container CDB$ROOT
  which are identified above as BEFORE UPGRADE "(AUTOFIXUP)" can be
resolved by
  executing the following from within the container

    SQL>@/u01/app/oracle/cfgtoollogs/CDB2/preupgrade/preupgrade_fixups.sql

=============
AFTER UPGRADE
=============

  REQUIRED ACTIONS
  ================
  None

  RECOMMENDED ACTIONS
  ===================
  6.  Upgrade the database time zone file using the DBMS_DST package.

      The database is using time zone file version 26 and the target
18.0.0.0.0
      release ships with time zone file version 31.

      Oracle recommends upgrading to the desired (latest) version of the
time
      zone file.  For more information, refer to "Upgrading the Time Zone
File
      and Timestamp with Time Zone Data" in the 18.0.0.0.0 Oracle Database
      Globalization Support Guide.

  7.  (AUTOFIXUP) Gather dictionary statistics after the upgrade using the
      command:

        EXECUTE DBMS_STATS.GATHER_DICTIONARY_STATS;

      Oracle recommends gathering dictionary statistics after upgrade.
```

```
      Dictionary statistics provide essential information to the Oracle
      optimizer to help it find efficient SQL execution plans. After a
database
      upgrade, statistics need to be re-gathered as there can now be tables
      that have significantly changed during the upgrade or new tables
that do
      not have statistics gathered yet.

  8.  Gather statistics on fixed objects after the upgrade and when there
is a
      representative workload on the system using the command:

        EXECUTE DBMS_STATS.GATHER_FIXED_OBJECTS_STATS;

      This recommendation is given for all preupgrade runs.

      Fixed object statistics provide essential information to the Oracle
      optimizer to help it find efficient SQL execution plans.  Those
      statistics are specific to the Oracle Database release that generates
      them, and can be stale upon database upgrade.

      For information on managing optimizer statistics, refer to the
12.2.0.1
      Oracle Database SQL Tuning Guide.

  ORACLE GENERATED FIXUP SCRIPT
  ============================
  All of the issues in database CDB2 container CDB$ROOT
  which are identified above as AFTER UPGRADE "(AUTOFIXUP)" can be
resolved by
  executing the following from within the container

    SQL>@/u01/app/oracle/cfgtoollogs/CDB2/preupgrade/postupgrade_fixups.sql


Report generated by Oracle Database Pre-Upgrade Information Tool Version
18.0.0.0.0 on 2018-06-13T15:08:58

Upgrade-To version: 18.0.0.0.0

======================================
Status of the database prior to upgrade
======================================
      Database Name:  CDB2
     Container Name:  PDB$SEED
       Container ID:  2
            Version:  12.2.0.1.0
         Compatible:  12.2.0
          Blocksize:  8192
           Platform:  Linux x86 64-bit
      Timezone File:  26
  Database log mode:  NOARCHIVELOG
           Readonly:  TRUE
            Edition:  EE
```

```
  Oracle Component                       Upgrade Action    Current Status
  ----------------                       --------------    --------------
  Oracle Server                          [to be upgraded]  VALID
  Real Application Clusters              [to be upgraded]  OPTION OFF
  Oracle Workspace Manager               [to be upgraded]  VALID
  Oracle XML Database                    [to be upgraded]  VALID

==============
BEFORE UPGRADE
==============

  REQUIRED ACTIONS
  ================
  None

  RECOMMENDED ACTIONS
  ===================
  1.  Run 12.2.0.1.0 $ORACLE_HOME/rdbms/admin/utlrp.sql to recompile
invalid
      objects.  You can view the individual invalid objects with

        SET SERVEROUTPUT ON;
        EXECUTE DBMS_PREUP.INVALID_OBJECTS;

      6 objects are INVALID.

      There should be no INVALID objects in SYS/SYSTEM or user schemas
before
      database upgrade.

  2.  Review and remove any unnecessary HIDDEN/UNDERSCORE parameters.

      The database contains the following initialization parameters whose
name
      begins with an underscore:

      _exclude_seed_cdb_view

      Remove hidden parameters before database upgrade unless your
application
      vendors and/or Oracle Support state differently.  Changes will need
to be
      made in the pfile/spfile.

  3.  (AUTOFIXUP) Gather stale data dictionary statistics prior to database
      upgrade in off-peak time using:

        EXECUTE DBMS_STATS.GATHER_DICTIONARY_STATS;

      Dictionary statistics do not exist or are stale (not up-to-date).

      Dictionary statistics help the Oracle optimizer find efficient SQL
      execution plans and are essential for proper upgrade timing. Oracle
      recommends gathering dictionary statistics in the last 24 hours
before
```

```
      database upgrade.

      For information on managing optimizer statistics, refer to the
12.2.0.1
      Oracle Database SQL Tuning Guide.

  4.  (AUTOFIXUP) Gather statistics on fixed objects prior the upgrade.

      None of the fixed object tables have had stats collected.

      Gathering statistics on fixed objects, if none have been gathered
yet, is
      recommended prior to upgrading.

      For information on managing optimizer statistics, refer to the
12.2.0.1
      Oracle Database SQL Tuning Guide.

  INFORMATION ONLY
  ================
  5.  To help you keep track of your tablespace allocations, the following
      AUTOEXTEND tablespaces are expected to successfully EXTEND during the
      upgrade process.

                                                 Min Size
      Tablespace                    Size         For Upgrade
      ----------                 ----------  -----------
      SYSAUX                        235 MB       500 MB
      SYSTEM                        210 MB       584 MB
      TEMP                           20 MB       150 MB
      UNDOTBS1                      210 MB       412 MB

      Minimum tablespace sizes for upgrade are estimates.

  6.  No action needed.

      Using default parallel upgrade options, this CDB with 1 PDBs will
first
      upgrade the CDB$ROOT, and then upgrade at most 1 PDBs at a time
using 2
      parallel processes per PDB.

      The number of PDBs upgraded in parallel and the number of parallel
      processes per PDB can be adjusted as described in Database Upgrade
Guide.

  ORACLE GENERATED FIXUP SCRIPT
  =============================
  All of the issues in database CDB2 container PDB$SEED
  which are identified above as BEFORE UPGRADE "(AUTOFIXUP)" can be
resolved by
  executing the following from within the container

    SQL>@/u01/app/oracle/cfgtoollogs/CDB2/preupgrade/preupgrade_fixups.sql
```

```
=============
AFTER UPGRADE
=============

  REQUIRED ACTIONS
  ================
  None

  RECOMMENDED ACTIONS
  ===================
  7.  Upgrade the database time zone file using the DBMS_DST package.

      The database is using time zone file version 26 and the target
18.0.0.0.0
      release ships with time zone file version 31.

      Oracle recommends upgrading to the desired (latest) version of the
time
      zone file.  For more information, refer to "Upgrading the Time Zone
File
      and Timestamp with Time Zone Data" in the 18.0.0.0.0 Oracle Database
      Globalization Support Guide.

  8.  (AUTOFIXUP) Gather dictionary statistics after the upgrade using the
      command:

        EXECUTE DBMS_STATS.GATHER_DICTIONARY_STATS;

      Oracle recommends gathering dictionary statistics after upgrade.

      Dictionary statistics provide essential information to the Oracle
      optimizer to help it find efficient SQL execution plans. After a
database
      upgrade, statistics need to be re-gathered as there can now be tables
      that have significantly changed during the upgrade or new tables
that do
      not have statistics gathered yet.

  9.  Gather statistics on fixed objects after the upgrade and when there
is a
      representative workload on the system using the command:

        EXECUTE DBMS_STATS.GATHER_FIXED_OBJECTS_STATS;

      This recommendation is given for all preupgrade runs.

      Fixed object statistics provide essential information to the Oracle
      optimizer to help it find efficient SQL execution plans.  Those
      statistics are specific to the Oracle Database release that generates
      them, and can be stale upon database upgrade.

      For information on managing optimizer statistics, refer to the
12.2.0.1
      Oracle Database SQL Tuning Guide.
```

```
  ORACLE GENERATED FIXUP SCRIPT
  ============================
  All of the issues in database CDB2 container PDB$SEED
  which are identified above as AFTER UPGRADE "(AUTOFIXUP)" can be
resolved by
  executing the following from within the container

    SQL>@/u01/app/oracle/cfgtoollogs/CDB2/preupgrade/postupgrade_fixups.sql


==================
PREUPGRADE SUMMARY
==================
  /u01/app/oracle/cfgtoollogs/CDB2/preupgrade/preupgrade.log
  /u01/app/oracle/cfgtoollogs/CDB2/preupgrade/preupgrade_fixups.sql
  /u01/app/oracle/cfgtoollogs/CDB2/preupgrade/postupgrade_fixups.sql

Execute fixup scripts across the entire CDB:

Before upgrade:

1. Execute preupgrade fixups with the below command
$ORACLE_HOME/perl/bin/perl -I$ORACLE_HOME/perl/lib -I$ORACLE_HOME/rdbms/
admin $ORACLE_HOME/rdbms/admin/catcon.pl -l /u01/app/oracle/cfgtoollogs/
CDB2/preupgrade/ -b preup_CDB2 /u01/app/oracle/cfgtoollogs/CDB2/preupgrade/
preupgrade_fixups.sql

2. Review logs under /u01/app/oracle/cfgtoollogs/CDB2/preupgrade/

After the upgrade:

1. Execute postupgrade fixups with the below command
$ORACLE_HOME/perl/bin/perl -I$ORACLE_HOME/perl/lib -I$ORACLE_HOME/rdbms/
admin $ORACLE_HOME/rdbms/admin/catcon.pl -l /u01/app/oracle/cfgtoollogs/
CDB2/preupgrade/ -b postup_CDB2 /u01/app/oracle/cfgtoollogs/CDB2/
preupgrade/postupgrade_fixups.sql

2. Review logs under /u01/app/oracle/cfgtoollogs/CDB2/preupgrade/

Preupgrade complete: 2018-06-13T15:09:00
```

Enabling Oracle Database Vault After Upgrading Oracle Database

Depending on your target database release, you can be required to disable Oracle Database Vault to complete an Oracle Database upgrade.

- Upgrading Oracle Database Without Disabling Oracle Database Vault
  If your target Oracle Database release is 12.2 or later, then you can upgrade without disabling Oracle Database Vault.

- Common Upgrade Scenarios with Oracle Database Vault
  The requirements to enable Oracle Database Vault after upgrades change, depending on your source Oracle Database release.

Upgrading Oracle Database Without Disabling Oracle Database Vault

If your target Oracle Database release is 12.2 or later, then you can upgrade without disabling Oracle Database Vault.

If you have Oracle Database Vault enabled in your source Oracle Database release, then you can upgrade Oracle Database to Oracle Database 18c without first disabling Oracle Database Vault. After the upgrade, if your source Oracle Database release is Oracle Database 12*c* release 1 (12.1) or later, then Oracle Database Vault is enabled with the same enforcement settings that you had in place before the upgrade. For example, if your source database is Oracle Database release 12.1, and Oracle Database Vault was disabled in that release, then it remains disabled after you upgrade. If your source Oracle Database release 12.1 database had Oracle Database Vault enabled before the upgrade, then Oracle Database Vault is enabled after the upgrade.

If you manually disable Oracle Database Vault before the upgrade, then you must enable Oracle Database Vault manually after the upgrade.

If you did not have Oracle Database Vault enabled before the upgrade, then you can enable it manually after the upgrade.

Enable Oracle Database Vault in the upgraded database by using the procedure `dvsys.dbms_macadm.enable_dv()`. Run this procedure with a user account that is granted `DV_OWNER`. After you run the procedure, restart the database instance so that the procedure takes effect.

Related Topics

- *Oracle Database Vault Administrator's Guide*

Common Upgrade Scenarios with Oracle Database Vault

The requirements to enable Oracle Database Vault after upgrades change, depending on your source Oracle Database release.

- Upgrades from Oracle Database 11*g* release 2 (11.2) or earlier: After the upgrade, Oracle Database Vault is disabled by default.
- Upgrades from Oracle Database 12*c* release 1 (12.1) or later: After the upgrade, Oracle Database Vault has the same enforcement status that you had in place before the upgrade.

Table 8-1 Common Oracle Database Vault Upgrade Scenarios and Upgrade Preparation Tasks

Source Database Release	Target Database Release	Do you need to disable Database Vault Before Upgrade	What is Database Vault Status After Upgrade
11.2 or earlier	12.1	Yes	Disabled. You need to enable Database Vault manually after the upgrade.
11.2.or earlier	12.2, 18.1 and later	No	Disabled. You need to enable Database Vault manually after the upgrade.
12.1, 12.2, 18.1, and later	12.2, 18.1 and later	No	Database Vault has the same enforcement status that you had in place before the upgrade.

9
Upgrading Oracle Database

Oracle offers several methods to upgrade your database, which support the complexities of your enterprise.

- Using DBUA to Upgrade the Database on Linux, Unix, and Windows Systems
  To upgrade a database using the DBUA graphical user interface, perform these steps from within the new Oracle home.
- Manually Upgrading Non-CDB Architecture Oracle Databases
  This procedure provides steps for upgrading non-CDB architecture Oracle Databases.

Using DBUA to Upgrade the Database on Linux, Unix, and Windows Systems

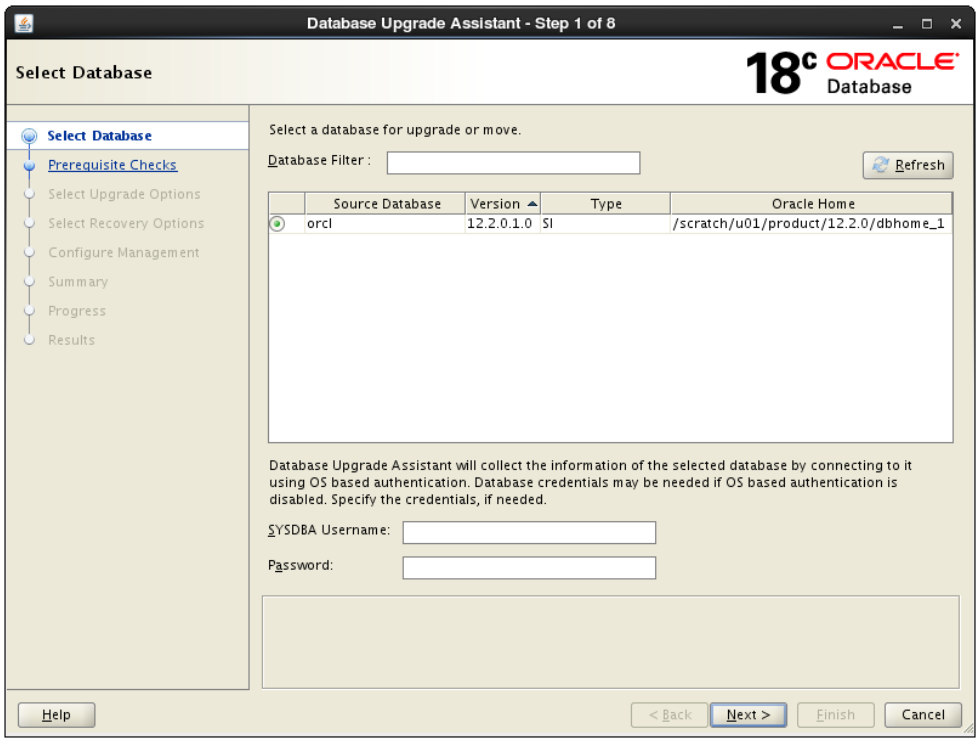
To upgrade a database using the DBUA graphical user interface, perform these steps from within the new Oracle home.

On Microsoft Windows systems (Windows), run DBUA either as an Oracle Database administrative user (a user with the operating system-assigned `ORA_DBA` role), or the Oracle installation owner account installation.

1. Start Oracle Database Upgrade Assistant (DBUA) from the Oracle home where the new database software is installed. The `dbua` executable is located in the directory path *`ORACLE_HOME`*`/bin`.
   - On Linux or Unix platforms, log in as a user with SYSDBA privileges, and enter the following command at a system prompt in the new home for Oracle Database 18c:

     ```
     ./dbua
     ```

   - On Windows operating systems, select **Start**, then **Programs**, then **Oracle *HOME_NAME***, then **Configuration and Migration Tools**, and then **Database Upgrade Assistant**.
2. The Select Database window displays. If you have earlier release Oracle Database installations, then these installations are listed as available to upgrade.



If you need help on any DBUA window, or if you want to consult more documentation about DBUA, then click **Help** to open the online help.

If needed, enter the SYSDBA user name and password for the database that you select.

If you run DBUA from a user account that does not have `SYSDBA` privileges, or if the source database does not have operating system authentication, then you must enter the user name and password credentials to enable `SYSDBA` privileges for the selected database. If the user account you use has SYSDBA privileges, or you used operating system authentication, then you do not need to enter the user name and password.

Click **Next** after making your selection.

Note:

- You can select only one database at a time.
- With single-instance upgrades, if the database does not appear in the list, then check to see if an entry with the database name exists in `/etc/oratab`. If the database is not listed there, then direct DBUA to upgrade particular databases:
  - If your single-instance database is not listed in `/etc/oratab`, and DBUA can connect to the database, then you can direct DBUA to upgrade that database explicitly by starting DBUA using the command-line arguments `-sid` *`Oracle_SID`* , `-oracleHome` *`Oracle_home`*, and `sysDBAPassword` *`mypassword`* as a command-line argument. For example:

    ```
    dbua -sid Oracle_SID –oracleHome /u01/app/oracle/18.1.0/
    dbhome1 -sysDBAUserName SYS -sysDBAPassword mypassword
    ```

- If your account does not have SYSDBA privileges, or you do not have operating system authentication set up, then you can use the following syntax to connect, where *`mydb`* is your Oracle Database SID, *`username`* is a user name with SYSDBA privileges, and *`password`* is that user name's password:

  ```
  dbua -sid mydb –oracleHome /u01/app/oracle/18.1.0/dbhome1 -
  sysDBAUserName - username -sysDBAPassword - password
  ```

- Oracle Real Application Clusters (Oracle RAC) upgrades: If the database does not appear on the list, then enter the following `crsctl` command to check for Oracle RAC instances:

  ```
  crsctl status resource -t
  ```

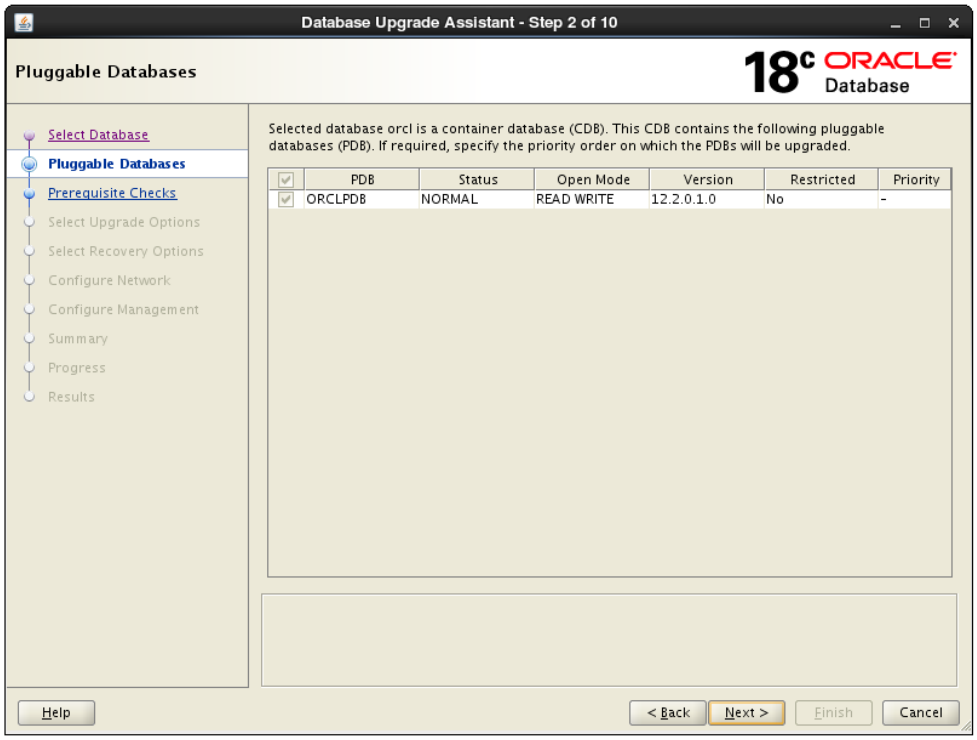
  You can also enter the following command to check for a particular Oracle RAC database, where *`db_name`* is the Oracle RAC database name:

  ```
  crsctl status resource ora.db_name.db
  ```

- On Microsoft Windows, the following security changes affect authentication and user accounts:
  - For security reasons, Windows NTS authentication using the NTLM protocol is no longer supported. Kerberos authentication is the only supported authentication. In this release, NTS does not work either in Windows NT domains, or in domains with Windows NT controllers.
  - Oracle uses standard Microsoft Windows user accounts instead of the Windows `LocalSystem` account to run Oracle database services. Reducing the account access privileges for the Oracle installation owner provides better security on Microsoft Windows.

3. If the selected database is a multitenant container database (CDB), then DBUA displays the Pluggable Databases window. The Pluggable Databases window lists

the pluggable databases contained in the CDB. The listed PDBs are upgraded as part of the upgrade for the selected CDB.

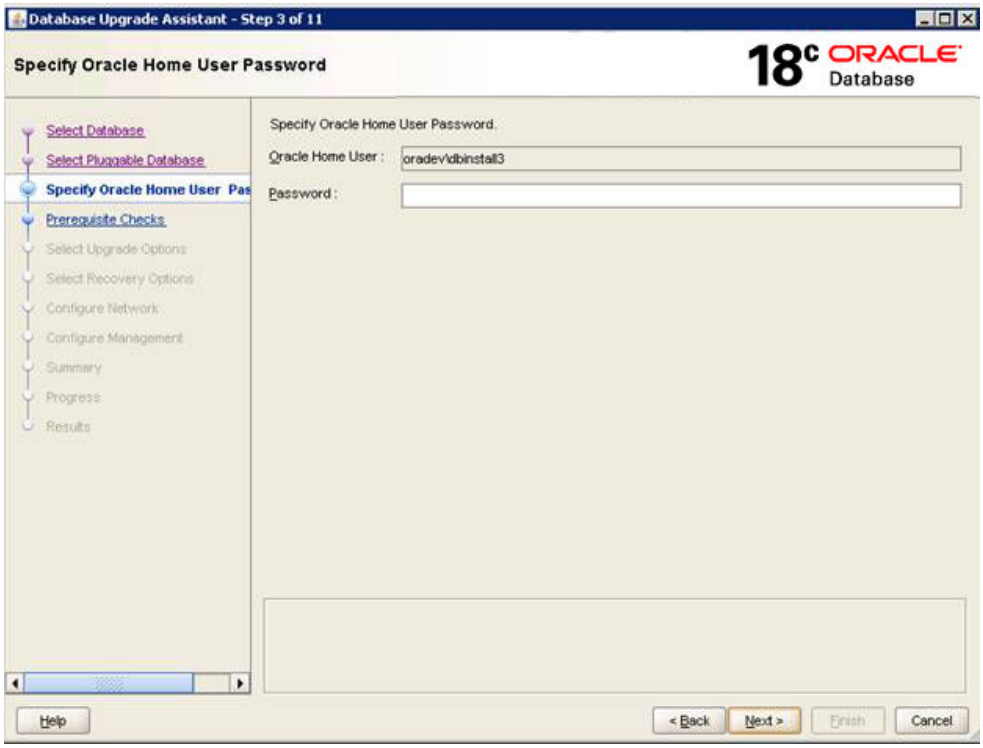


Starting in release 12.2, you can select the upgrade priority for PDBs. Click in the priority column for each PDB, and enter a numeric value for upgrade priority, where 1 is upgraded first, 2 is upgraded second, and so on.

By default, CDB$ROOT, PDB$SEED, and all PDBs that are plugged into the CDB are upgraded. If you do not want some PDBs to be upgraded now, then unplug those PDBs.

When you have completed selecting PDBs and upgrade priorities, click **Next**.

4. Windows platforms only: If the upgrade target home is a secure home that is associated with an Oracle home user, then the Specify Oracle Home User Password window opens. For other platforms, proceed to the next step.



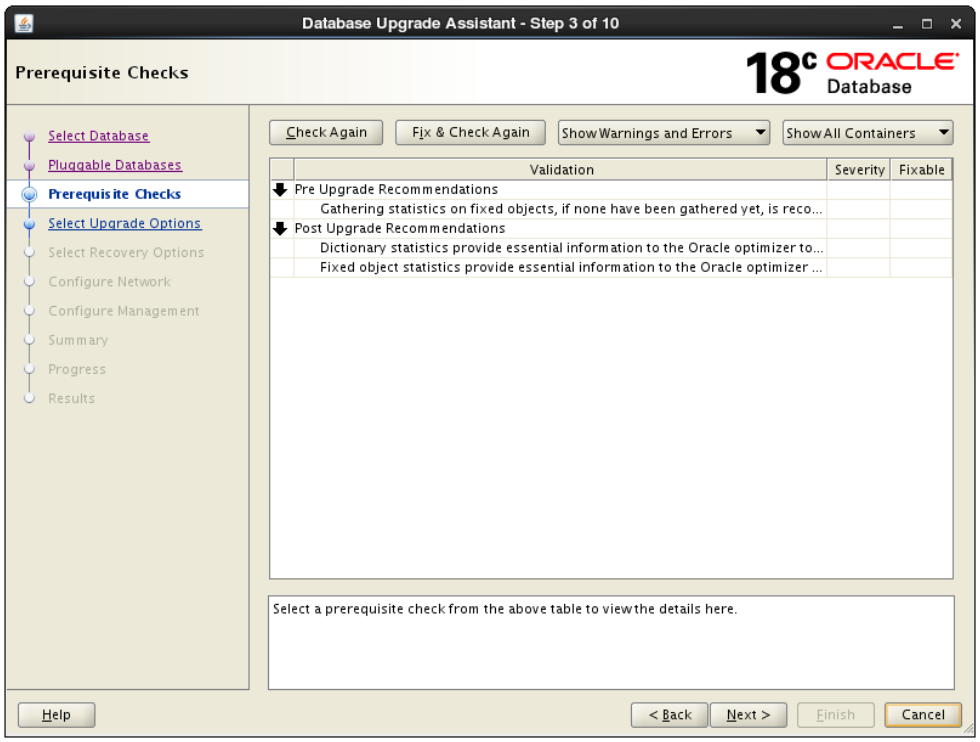
Provide the Oracle home user name, and provide the password for this user account, and click **Next**.

5. The Prerequisite Checks window opens. DBUA analyzes the databases, performing pre-upgrade checks and displaying warnings as necessary. The following is a list of examples of DBUA checks, and of actions that DBUA performs on the database:

   - Empty database recycle bin.
   - Identify invalid objects.
   - Identify deprecated and desupported initialization parameters.
   - Identify time zone data file version.

   The analysis takes several minutes to complete.

   When DBUA finishes its analysis, the Prerequisite Checks window displays again, showing the results of the checks.



The Prerequisite Checks window shows the checks that DBUA has completed, and the severity of any errors discovered. When DBUA finds errors, it indicates which errors are fixable, and what action you can take to correct the error.

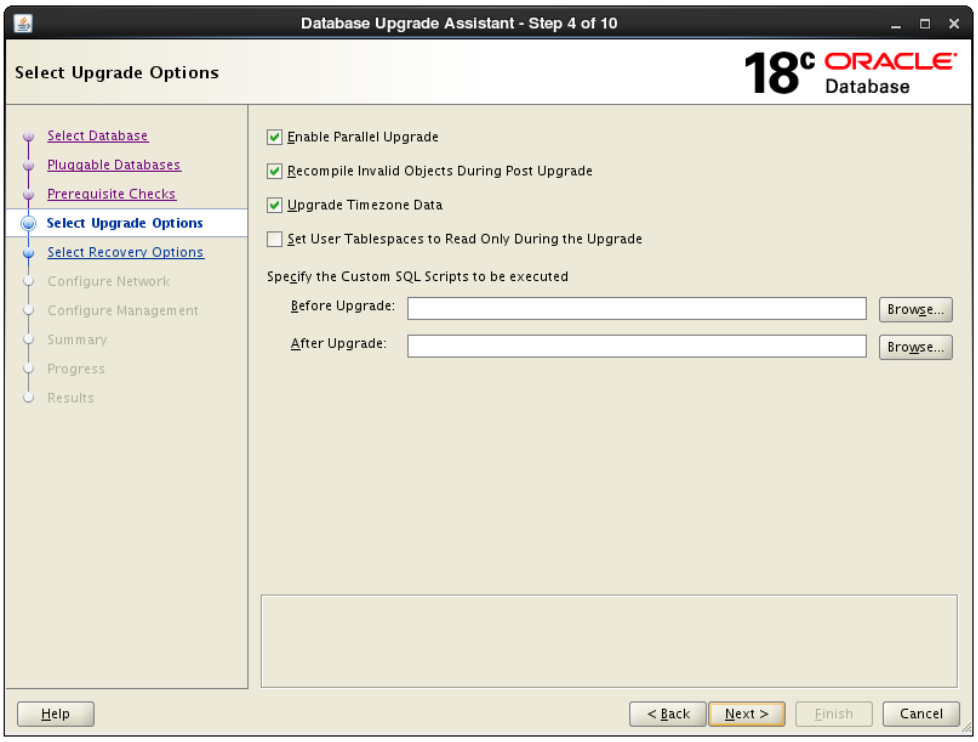
Select **Fix & Check Again** if any errors that DBUA can fix appear.

If DBUA detects errors that it cannot correct, then fix the cause of the error manually, and select **Check Again**.

If DBUA finds no errors or warnings, then the DBUA automatically bypasses this window and proceeds to the next window.

When you have fixed detected errors, click **Next**.

6. The Select Upgrade Options window displays.



This window provides the following options:

Enable Parallel Upgrade

Select this option if you want to enable parallelism during the upgrade process. Running upgrade processes in parallel reduces the time required to perform the upgrade, based on the number of CPUs available to handle the running of scripts and processes simultaneously.

Recompile Invalid Objects During Post Upgrade

This option recompiles all invalid PL/SQL modules after the upgrade is complete. If you do not have DBUA recompile invalid objects in its post-upgrade phase, then you must manually recompile invalid objects after the database is upgraded.

Upgrade Time Zone Data

This option updates the time zone data file for this release. If you do not select this option, then you must update the time zone configuration file manually after the upgrade.

Specify custom SQL scripts to be executed.

If you want to run custom SQL scripts as part of your upgrade, then select this option. As needed, click **Browse** for the **Before Upgrade** or **After Upgrade** input fields. Navigate to the location where your custom SQL scripts are located.

When you have made your selections, click **Next**.

7. The Select Recovery Options window appears. To recover the database if a failure occurs during upgrade, select from one of the following options:

   - **Use Flashback and Guaranteed Restore Point**.

You can create a new Guaranteed Restore Point, or use an existing one. If you use an existing restore point, then click the selection field to select the restore point that you want to use.

> **Note:**
>
> If the database that you are upgrading has Oracle Data Guard physical standbys, then you must first create a guaranteed restore point on each standby before you create one on the primary database. If you do not create restore points on each standby first, then you must recreate all standby databases again after using the guaranteed restore point to downgrade the primary database. After the upgrade is successful, you must manually drop all guaranteed restore points on the standbys.

- **Use RMAN Backup**

  You can create a new offline RMAN backup, or use an existing backup. Click **Browse** to specify a path for the backup.

- **Use Latest Available RMAN Backup**

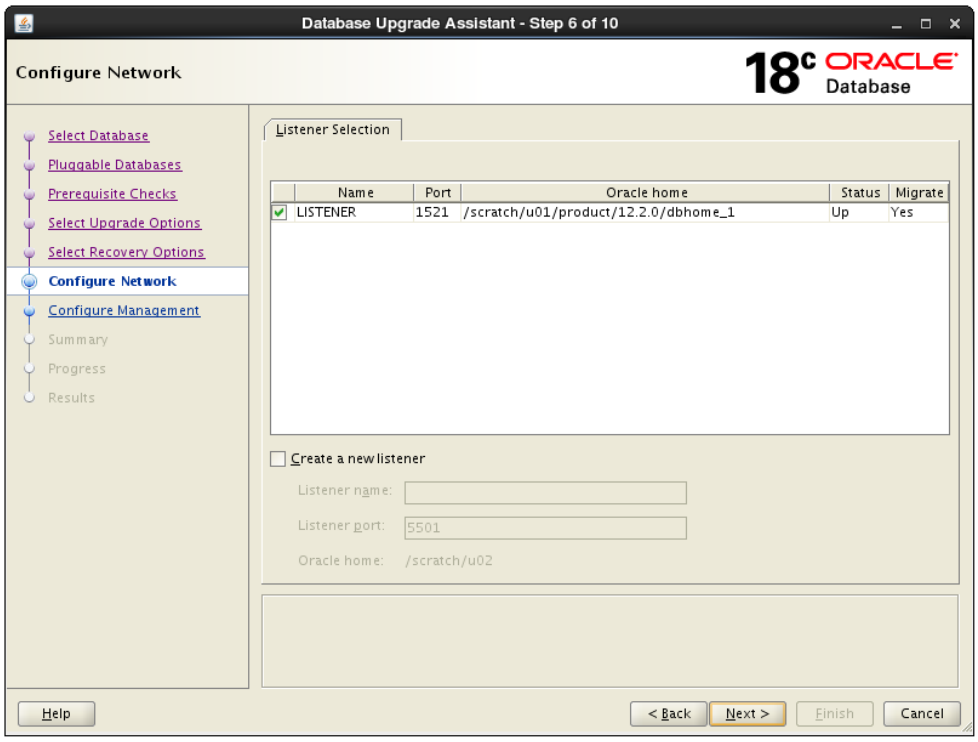
  You can use an existing backup. Click **View/Edit Restore Script** to select the backup that you want to use.

- **I have my own backup and restore strategy**.

  Select this option only if you have a third-party backup solution in place for your existing database.

When you have made your selections, click **Next**.

8. For single-instance database installations, the Configure Network window opens. Select one or more listeners from the source Oracle home that you want to migrate to the new upgraded Oracle home, or create a new listener during installation.



The Listener Selection area of the Network Configuration window shows a table with the following columns:

- Select column. Select the listeners that you want to update.
- **Name** This column shows listener names.
- **Port** This column shows the ports on which listeners are configured.
- **Oracle Home** This column shows the Oracle home where listeners are configured.
- **Status** This column shows the listener status (up or down).
- **Migrate** Select this column, and choose `Yes` to migrate, or `No` if you do not want to migrate.

You can also select to create a new listener. If you create a new listener, then provide the listener name, the Oracle home where you want it placed, and the port that you want to configure the listener to monitor.

After you make your choices, DBUA completes the following steps for any listeners that you migrate:

a. DBUA adds the selected listener to the `listener.ora` file of the target Oracle home, and starts it.

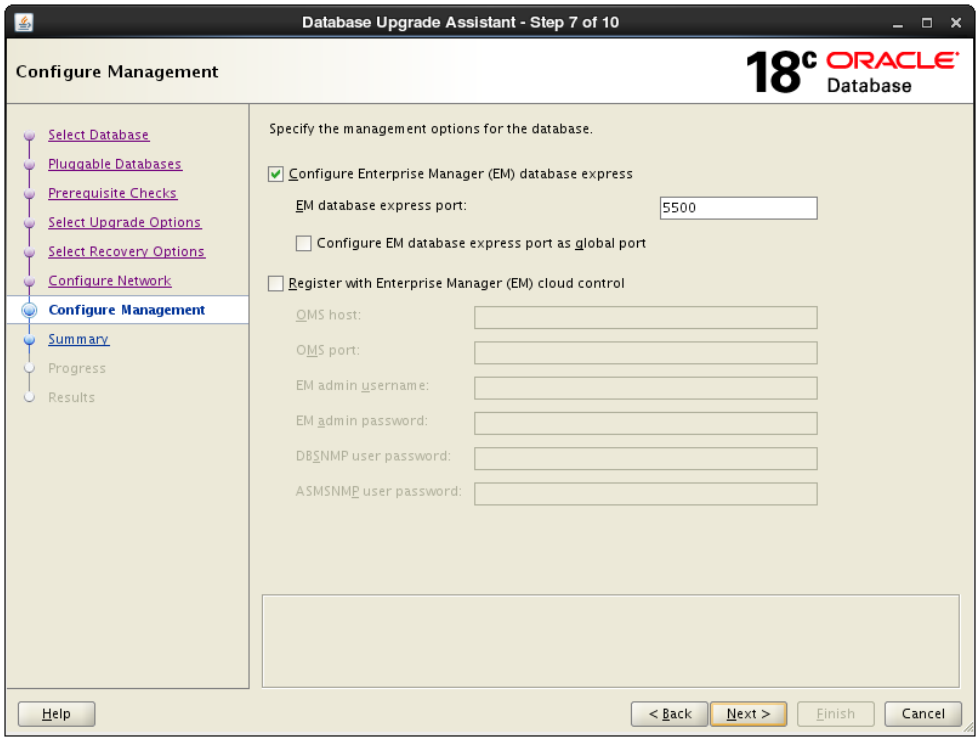
b. DBUA removes the entry of the upgraded database from the old (source) `listener.ora` file.

c. DBUA reloads the `listener.ora` file in both the source and target Oracle Database environments.

> **Note:**
>
> If there are other databases registered with the same listener, then their new client connection requests can be affected during listener migration.

Click **Next** when you have completed your choices.

9. The Configure Management window appears. In the Configure Management window, select the management options:



- **Configure Enterprise Manager (EM) database express**

  Oracle Enterprise Manager Database Express is a web-based database management application that is built into Oracle Database 12*c*. EM Express replaces the DB Control component that was available in releases 10*g* and 11*g*. Enter the EM Database Express Port number. For example: 5500. You can also select the checkbox to configure the express port as the global port.

- **Register with Enterprise Manager (EM) Cloud Control**

  Registering with Oracle Enterprise Manager Cloud Control adds the database and its related entities, such as Listener, Oracle ASM disk groups, and Oracle Clusterware, as targets that you can manage with EM Cloud Control.

  If you select this option, then you must provide information in the following fields:

  – OMS Host

  – OMS Port

- EM Admin Username
- EM Admin Password
- DBSNMP User Password
- ASMSNMP User Password

When you have completed entering information, click **Next**.

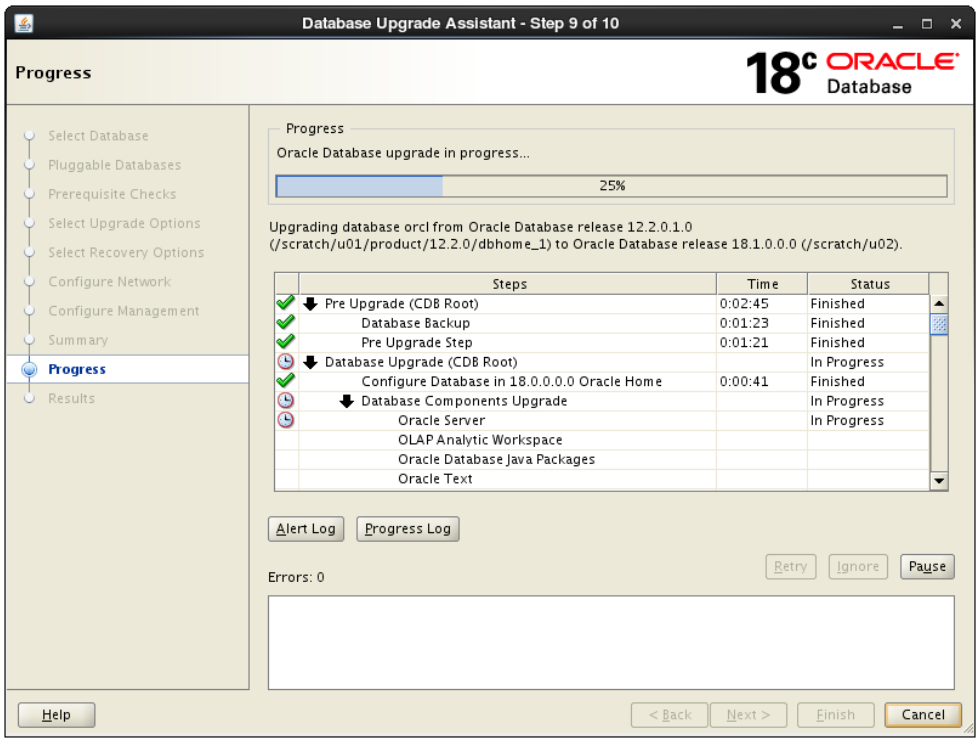
10. The Summary window opens. The Summary window shows the information that you have provided for the upgrade. Scroll down the list to review the information. The summary includes information such as the following:

    - Source Database
    - Target Database
    - Pluggable Databases
    - Pre-Upgrade Checks
    - Initialization Parameters changes
    - Timezone Upgrade

    Check your selections. Then, either select a link to the item that you want to change, or click **Back** to go to earlier pages, or select **Finish**:

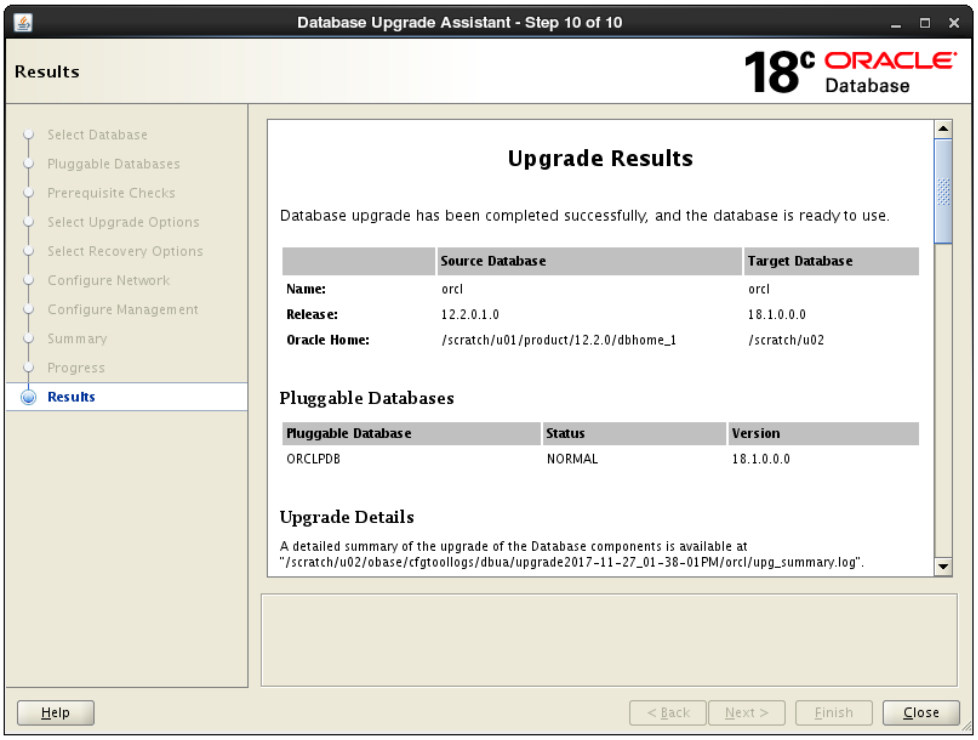
    - If you see information in the Summary window that you want to correct, then click a link on an item that you want to update, or click **Back** to navigate backward through the DBUA configuration interview.
    - Click **Finish** if the information that you see in the Summary window is correct. The upgrade begins after you select **Finish**.

    The Progress window displays with the progress bar, as DBUA begins the upgrade. The Progress window displays a table that shows the steps DBUA is completing during the upgrade. This table shows the time duration, and the upgrade steps status as the upgrade proceeds. DBUA provides a **Stop** button in case you must cancel the upgrade at this point.



When the upgrade has progressed through finishing the upgrade of the CDB root and each PDB seed, the Progress window marks the status as **Finished**.

11. After the upgrade is complete, the Results window opens. The Results window displays information about the original database, and about the upgraded database. The Upgrade Results report also shows changes that DBUA made to the initialization parameters. If you are upgrading a multitenant architecture database, then the Results window also shows pluggable databases, and the directory where log files are stored after the upgrade. Scroll down to see more details about preupgrade checks. If the upgrade is successful, then the Upgrade Results field reports the results, and you do not see warning messages. If the upgrade was unsuccessful, as this example image shows, then the **Restore Database** button is displayed on the lower right corner below the display field. You can click this button to start a database restoration.



12. Optional: Examine the log files to obtain more details about the upgrade process. If the Oracle base environment variable is set, then the DBUA log files are located in the path `/ORACLE_BASE/cfgtoollogs/dbua/upgradesession_timestamp/SID`. If Oracle base is not set, then the DBUA log files are located in the path `/ORACLE_HOME/cfgtoollogs/dbua/upgradesession_timestamp/SID`

> **Note:**
>
> An HTML version of the Upgrade Results window is also saved in the log files directory. You can click the links in this HTML window to view the log windows in your browser.

If you are satisfied with the upgrade results, then click **Close** to quit DBUA.

13. After your upgrade is completed, carry out post-upgrade procedures described in this book. When you have completed post-upgrade procedures, your upgraded database is ready to use.

> **Caution:**
>
> To prevent unauthorized use of the database, Oracle recommends that you change all user passwords immediately after you upgrade your database.
>
> If the default security settings for Oracle Database 12*c* are in place, then passwords must be at least eight characters. Passwords such as `welcome` and `oracle` are not allowed.

Manually Upgrading Non-CDB Architecture Oracle Databases

This procedure provides steps for upgrading non-CDB architecture Oracle Databases.

> **Note:**
>
> Starting with Oracle Database 12c Release 1 (12.1), non-CDB architecture is deprecated. It can be desupported in a future release.

Before using this procedure, complete the following steps:

- Install the Oracle Database software
- Prepare the new Oracle home
- Run the Pre-Upgrade Information Tool

Steps:

1. If you have not done so, run the Pre-Upgrade Information Tool. Review the Pre-Upgrade Information Tool output and correct all issues noted in the output before proceeding.

   For example, on Linux or Unix systems:

   ```
   $ORACLE_HOME/jdk/bin/java -jar /opt/oracle/product/18.0.0/rdbms/admin/
   preupgrade.jar FILE TEXT
   ```

2. Ensure that you have a proper backup strategy in place.
3. If you have not done so, prepare the new Oracle home.
4. (Conditional) For Oracle RAC environments only, enter the following commands to set the initialization parameter value for CLUSTER_DATABASE to FALSE:

   ```
   ALTER SYSTEM SET CLUSTER_DATABASE=FALSE SCOPE=SPFILE;
   ```

5. Shut down the database. For example:

   ```
   SQL> SHUTDOWN IMMEDIATE
   ```

6. If your operating system is Windows, then complete the following steps:
   a. Stop the `OracleService`*`SID`* Oracle service of the database you are upgrading, where *`SID`* is the instance name. For example, if your *`SID`* is `ORCL`, then enter the following at a command prompt:

      ```
      C:\> NET STOP OracleServiceORCL
      ```

   b. Delete the Oracle service at a command prompt using `ORADIM`. Refer to your platform guide for a complete list of the `ORADIM` syntax and commands.

For example, if your *SID* is ORCL, then enter the following command.

```
C:\> ORADIM -DELETE -SID ORCL
```

c. Create the service for the new release Oracle Database at a command prompt using the ORADIM command of the new Oracle Database release.

Use the following syntax, where *SID* is your database SID, *PASSWORD* is your system password, *USERS* is the value you want to set for maximum number of users, and *ORACLE_HOME* is your Oracle home:

```
C:\> ORADIM -NEW -SID SID -SYSPWD PASSWORD -MAXUSERS USERS
     -STARTMODE AUTO -PFILE ORACLE_HOME\DATABASE\INITSID.ORA
```

Most Oracle Database services log on to the system using the privileges of the Oracle software installation owner. The service runs with the privileges of this user. The ORADIM command prompts you to provide the password to this user account. You can specify other options using ORADIM.

In the following example, if your *SID* is ORCL, your *password* (SYSPWD) is TWxy5791, the maximum number of users (MAXUSERS) is 10, and the Oracle home path is C:\ORACLE\PRODUCT\18.0.0\DB, then enter the following command:

```
C:\> ORADIM -NEW -SID ORCL -SYSPWD TWxy5791 -MAXUSERS 10
-STARTMODE AUTO -PFILE C:\ORACLE\PRODUCT\18.0.0\DB\DATABASE
\INITORCL.ORA
```

ORADIM writes a log file to the ORACLE_HOME\database directory.

> **Note:**
>
> If you use an Oracle Home User account to own the Oracle home, then the ORADIM command prompts you for that user name and password.

7. If your operating system is Linux or UNIX, then perform the following checks:
   a. Your ORACLE_SID is set correctly
   b. The oratab file points to the new Oracle home
   c. The following environment variables point to the new Oracle Database directories:
      - ORACLE_HOME
      - PATH
   d. Any scripts that clients use to set the $ORACLE_HOME environment variable must point to the new Oracle home.

> **Note:**
>
> If you are upgrading an Oracle Real Application Clusters database, then perform these checks on all Oracle Grid Infrastructure nodes where the Oracle Real Application Clusters database has instances configured.

8. Log in to the system as the Oracle installation owner for the new Oracle Database release.
9. Start SQL*Plus in the new Oracle home from the admin directory in the new Oracle home directory.

   For example:

   ```
   $ cd $ORACLE_HOME/rdbms/admin
   $ pwd
   /u01/app/oracle/product/18.0.0/dbhome_1/rdbms/admin
   $ sqlplus
   ```

10. Copy the SPFILE.ORA or INIT.ORA file from the old Oracle home to the new Oracle home.
11. Connect to the database that you want to upgrade using an account with SYSDBA privileges:

    ```
    SQL> connect / as sysdba
    ```

12. Start the non-CDB Oracle Database in upgrade mode:

    ```
    SQL> startup upgrade
    ```

    If errors appear listing desupported initialization parameters, then make a note of the desupported initialization parameters and continue with the upgrade. Remove the desupported initialization parameters the next time you shut down the database.

    > **Note:**
    >
    > Starting up the database in `UPGRADE` mode enables you to open a database based on an earlier Oracle Database release. It also restricts log-ins to `AS SYSDBA` sessions, disables system triggers, and performs additional operations that prepare the environment for the upgrade.

13. Exit SQL*Plus.

    For example:

    ```
    SQL> EXIT
    ```

14. Run the Parallel Upgrade Utility (`catctl.pl`) script, using the upgrade options that you require for your upgrade.

You can run the Parallel Upgrade Utility as a command-line shell command by using the `dbupgrade` shell command, which is located in *`Oracle_home`*`/bin`. If you set the PATH environment variable to include `Oracle_home/bin`, then you can run the command directly from your command line. For example:

```
$ dbupgrade -d /u01/app/oracle/12.2.0/dbhome_1
```

> **Note:**
>
> - When you run the Parallel Upgrade Utility command, use the `-d` option to specify the filepath for the target Oracle home. Use the `-l` option to specify the directory that you want to use for spool log files.

15. The database is shut down after a successful upgrade. Restart the instance so that you reinitialize the system parameters for normal operation. For example:

    ```
    SQL> STARTUP
    ```

    This restart, following the database shutdown, flushes all caches, clears buffers, and performs other housekeeping activities. These measures are an important final step to ensure the integrity and consistency of the upgraded Oracle Database software.

    > **Note:**
    >
    > If you encountered a message listing desupported initialization parameters when you started the database, then remove the desupported initialization parameters from the parameter file before restarting it. If necessary, convert the `SPFILE` to a `PFILE`, so that you can edit the file to delete parameters.

16. Run `catcon.pl` to start `utlrp.sql`, and to recompile any remaining invalid objects.

    For example:

    ```
    $ORACLE_HOME/perl/bin/perl catcon.pl -n 1 -e -b utlrp -d '''.''' 
    utlrp.sql
    ```

    Because you run the command using `-b utlrp`, the log file `utlrp0.log` is generated as the script is run. The log file provides results of the recompile.

17. Run `postupgrade_fixups.sql`. For example:

    ```
    SQL> @postupgrade_fixups.sql
    ```

> **Note:**
>
> If you did not specify to place the script in a different location, then it is in the default path *Oracle_base*/cfgtoollogs/*SID*/preupgrade, where *Oracle_base* is your Oracle base home path, and *SID* is your unique database name.

18. Run `utlu122s.sql`. The script verifies that all issues are fixed.

    For example:

    ```
    SQL> @$ORACLE_HOME/rdbms/admin/utllu122s.sql
    ```

    The log file `utlu122s0.log` is generated as the script is run, which provides the upgrade results. You can also review the upgrade report in `upg_summary.log`.

    To see information about the state of the database, run `utlu122s.sql` as many times as you want, at any time after the upgrade is completed. If the `utlu122s.sql` script returns errors, or shows components that do not have the status `VALID`, or if the version listed for the component is not the most recent release, then refer to the troubleshooting section in this guide.

19. Ensure that the time zone data files are current by using the `DBMS_DST PL/SQL` package to upgrade the time zone file. You can also adjust the time zone data files after the upgrade.

20. Exit from SQL*Plus

    For example:

    ```
    SQL> EXIT
    ```

21. (Conditional) If you are upgrading an Oracle Real Application Clusters database, then use the following command syntax to upgrade the database configuration in Oracle Clusterware:

    ```
    srvctl upgrade database -db db-unique-name -oraclehome oraclehome
    ```

    In this syntax example, *db-unique-name* is the database name (not the instance name), and *oraclehome* is the Oracle home location in which the database is being upgraded. The `SRVCTL` utility supports long GNU-style options, in addition to short command-line interface (CLI) options used in earlier releases.

22. (Conditional) For Oracle RAC environments only, after you have upgraded all nodes, enter the following commands to set the initialization parameter value for CLUSTER_DATABASE to TRUE, and start the database, where *db_unique_name* is the name of the Oracle RAC database:

    ```
    ALTER SYSTEM SET CLUSTER_DATABASE=TRUE SCOPE=SPFILE;
    srvctl start database -db db_unique_name
    ```

Your database is now upgraded. You are ready to complete post-upgrade procedures.

Caution:

If you retain the old Oracle software, then never start the upgraded database with the old software. Only start Oracle Database using the start command in the new Oracle Database home.

Before you remove the old Oracle environment, relocate any data files in that environment to the new Oracle Database environment.

See Also:

Oracle Database Administrator's Guide for information about relocating data files

10

Post-Upgrade Tasks for Oracle Database

After you have finished upgrading Oracle Database, complete the required post-upgrade tasks and consider these recommendations for the new release.

- Check the Upgrade With Post-Upgrade Status Tool
  Review the upgrade spool log file and use the Post-Upgrade Status Tool, `utlu122s.sql`.
- How to Show the Current State of the Oracle Data Dictionary
  Use one of three methods to check the state of the Oracle Data Dictionary for diagnosing upgrades and migrations.
- Required Tasks to Complete After Upgrading Oracle Database
  Review and complete these required tasks that are specified for your environment after you complete your upgrade.
- Recommended and Best Practices to Complete After Upgrading Oracle Database
  Oracle recommends that you complete these good practices guidelines for updating Oracle Database. These practices are recommended for both manual and DBUA upgrades.

Check the Upgrade With Post-Upgrade Status Tool

Review the upgrade spool log file and use the Post-Upgrade Status Tool, `utlu122s.sql`.

The Post-Upgrade Status Tool is a SQL script that is included with Oracle Database. You run the Post-Upgrade Status Tool in the environment of the new release. You can run the Post-Upgrade Status Tool at any time after you upgrade the database.

How to Show the Current State of the Oracle Data Dictionary

Use one of three methods to check the state of the Oracle Data Dictionary for diagnosing upgrades and migrations.

Running the `dbupgdiag.sql` Script

The `dbupgdiag.sql` script collects upgrade and migration diagnostic information about the current state of the data dictionary.

You can run the script in SQL*Plus both before the upgrade on the source database, and after the upgrade on the upgraded database as the SYS user. Refer to My Oracle Support note 556610.1 for more information about using the `dbupgdiag.sql` script to collect upgrade and migrate diagnostic information.

Running a SQL Query on DBA_REGISTERY

To show the current state of the dictionary, perform a SQL query similar to the following example:

```
SQL> spool /tmp/regInvalid.out
SQL> set echo on
-- query registry
SQL> set lines 80 pages 100
SQL> select substr(comp_id,1,15) comp_id,substr(comp_name,1,30)
      comp_name,substr(version,1,10) version,status
from dba_registry order by modified;
```

Running a Query to Check for Invalid Objects

To query invalid objects, perform a SQL query similar to the following example:

```
SQL> select owner, object_name, object_type from dba_invalid_objects order
by owner, object_type;
```

After you have upgraded the database, and you have run `utlrp.sql`, this view query should return no rows.

Related Topics

- https://support.oracle.com/rs?type=doc&id=556610.1

Required Tasks to Complete After Upgrading Oracle Database

Review and complete these required tasks that are specified for your environment after you complete your upgrade.

You must complete these postupgrade tasks after you upgrade Oracle Database. You must complete these tasks both when you perform the upgrade manually, and when you upgrade by using Database Upgrade Assistant (DBUA).

- Setting Environment Variables on Linux and Unix Systems After Manual Upgrades
  If you performed a manual upgrade of Oracle Database, then you must ensure that the required operating system environment variables point to the directories of the new Oracle Database release.
- Recompiling All Invalid Objects
  Oracle recommends that you run the `utlrp.sql` script after you install, patch, or upgrade a database, to identify and recompile invalid objects.
- Track Invalid Object Recompilation Progress
  Use these SQL queries to track the progress of `utlrp.sql` script recompilation of invalid objects.
- Running OPatch Commands After Upgrading Oracle Database
  After you upgrade Oracle Database, you must run OPatch commands from the new Oracle home.

- Setting oratab and Scripts to Point to the New Oracle Location After Upgrading Oracle Database
  You must set scripts to point to the new Oracle home location.
- Check PL/SQL Packages and Dependent Procedures
  It is possible that packages that you installed in the earlier release Oracle Database are not available in the new release, which cam affect applications.
- Upgrading Tables Dependent on Oracle-Maintained Types
  Starting with Oracle Database 12*c* Release 2 (12.2) and later releases, you must manually upgrade user tables that depend on Oracle-Maintained types.
- Enabling the New Extended Data Type Capability
  Enabling a system to take advantage of the new extended data types requires specific upgrade actions.
- Adjusting Minimum and Maximum for Parallel Execution Servers
  Depending on your environment, you can reduce the default setting of the `PARALLEL_MIN_SERVERS` parameter.
- About Recovery Catalog Upgrade After Upgrading Oracle Database
  If you use a version of the recovery catalog schema that is older than that required by the RMAN client, then you must upgrade it.
- Upgrading the Time Zone File Version After Upgrading Oracle Database
  If the Pre-Upgrade Information Tool instructs you to upgrade the time zone files after completing the database upgrade, then use the `DBMS_DST` PL/SQL package to upgrade the time zone file.
- Upgrading Statistics Tables Created by the DBMS_STATS Package After Upgrading Oracle Database
  If you created statistics tables using the `DBMS_STATS.CREATE_STAT_TABLE` procedure, then upgrade these tables by running `DBMS_STATS.UPGRADE_STAT_TABLE`.
- Upgrading Externally Authenticated SSL Users After Upgrading Oracle Database
  If you are upgrading from Oracle9*i* Release 2 (9.2) or Oracle Database 10*g* Release 1 (10.1), and you are using externally authenticated SSL users, then you must run the SSL external users conversion (`extusrupgrade`) script to upgrade those users.
- Configuring the FTP and HTTP Ports and HTTP Authentication for Oracle XML DB
  Oracle Database Configuration Assistant (DBCA) does not configure ports for Oracle XML DB on Oracle Database 12*c* and later release upgrades use digest authentication.
- Install Oracle Text Supplied Knowledge Bases After Upgrading Oracle Database
  After an Oracle Database upgrade, all user extensions to the Oracle Text supplied knowledge bases must be regenerated.
- Update Oracle Application Express Configuration After Upgrading Oracle Database
  Oracle Application Express affects upgrade procedures, depending on the Oracle Application Express release and your database installation type.
- Configure Access Control Lists (ACLs) to External Network Services
  Oracle Database 12*c* and later releases include fine-grained access control to the `UTL_TCP`, `UTL_SMTP`, `UTL_MAIL`, `UTL_HTTP`, or `UTL_INADDR` packages.

- Check for the SQLNET.ALLOWED_LOGON_VERSION Parameter Behavior
  Connections to Oracle Database from clients earlier than release 10*g* fail with the error `ORA-28040: No matching authentication protocol`.

Setting Environment Variables on Linux and Unix Systems After Manual Upgrades

If you performed a manual upgrade of Oracle Database, then you must ensure that the required operating system environment variables point to the directories of the new Oracle Database release.

Confirm that the following environment variables point to the directories of the new Oracle home:

- `ORACLE_HOME`
- `PATH`

> **Note:**
>
> DBUA automatically makes necessary changes to Oracle environment variables.

Related Topics

- *Oracle Database Administrator's Guide*

Recompiling All Invalid Objects

Oracle recommends that you run the `utlrp.sql` script after you install, patch, or upgrade a database, to identify and recompile invalid objects.

The `utlrp.sql` script recompiles all invalid objects. Run the script immediately after installation, to ensure that users do not encounter invalid objects.

1. Start SQL*Plus:

   ```
   sqlplus "/ AS SYSDBA"
   ```

2. Run the `utlrp.sql` script, where *`Oracle_home`* is the Oracle home path:

   ```
   SQL> @Oracle_home/rdbms/admin/utlrp.sql
   ```

The `utlrp.sql` script automatically recompiles invalid objects in either serial or parallel recompilation, based on both the number of invalid objects, and on the number of CPUs available. CPUs are calculated using the number of CPUs (`cpu_count`) multiplied by the number of threads for each CPU (`parallel_threads_per_cpu`). On Oracle Real Application Clusters (Oracle RAC), this number is added across all Oracle RAC nodes.

Track Invalid Object Recompilation Progress

Use these SQL queries to track the progress of `utlrp.sql` script recompilation of invalid objects.

Oracle recommends that you run the `utlrp.sql` script after upgrade to recompile invalid objects. You can run SQL queries to monitor the script.

Example 10-1 Number of Invalid Objects Remaining

Enter this query to return the number of remaining invalid objects. This number decreases over time as the `utlrp.sql` script runs.

```
SELECT COUNT(*) FROM obj$ WHERE status IN (4, 5, 6);
```

Example 10-2 Number of Objects Recompiled

Enter this query to return the number of objects that `utlrp.sql` has compiled. This number increases over time as the script runs.

```
SELECT COUNT(*) FROM UTL_RECOMP_COMPILED;
```

Example 10-3 Number of Objects Recompiled with Errors

Enter this query to return the number of objects that `utlrp.sql` has compiled with errors.

```
select COUNT(DISTINCT(obj#)) "OBJECTS WITH ERRORS" from utl_recomp_errors;
```

If the number is higher than expected, then examine the error messages reported with each object. If you see errors due to system misconfiguration or resource constraints, then fix the cause of these errors, and run `utlrp.sql` again.

Running OPatch Commands After Upgrading Oracle Database

After you upgrade Oracle Database, you must run OPatch commands from the new Oracle home.

OPatch is a Java-based utility that you install with Oracle Universal Installer. Opatch is platform-independent. It runs on all supported operating systems. Another version of OPatch, called standalone OPatch, is also available. It runs on Oracle homes without Oracle Universal Installer.

Patches are a small collection of files copied over to an existing installation. They are associated with particular versions of Oracle products. When applied to the correct version of an installed product, patches result in an upgraded version of the product.

Run Opatch to Check the Oracle Database Inventory

Log in as the Oracle installation owner, and run the `lsinventory` command from the new Oracle home. The command generates an accurate and complete inventory of the Oracle software installed on the system:

```
opatch lsinventory –patch
```

Refer to My Oracle Support note 756671.1 regularly to obtain current recommendations regarding Release Updates (Updates) and Release Update Revisions (Revisions).

Related Topics

- https://support.oracle.com/rs?type=doc&id=756671.1

Setting oratab and Scripts to Point to the New Oracle Location After Upgrading Oracle Database

You must set scripts to point to the new Oracle home location.

After you upgrade Oracle Database to a new release, you must ensure that your `oratab` file and any client scripts that set the value of *`ORACLE_HOME`* point to the new Oracle home that is created for the new Oracle Database release. DBUA automatically points `oratab` to the new Oracle home. However, you must check client scripts regardless of the method you use to upgrade.

If you upgrade your database manually, then you must log in as the Oracle installation owner for the new Oracle Database release, and update the `oratab` file manually. The location of the `oratab` file can vary, depending on your operating system.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for information about setting operating system environment variables
>
> My Oracle Support: Find or Create Oratab File (Doc ID 394251.1)
>
> https://support.oracle.com/rs?type=doc&id=394251.1

Check PL/SQL Packages and Dependent Procedures

It is possible that packages that you installed in the earlier release Oracle Database are not available in the new release, which cam affect applications.

After the upgrade, check to ensure that any packages that you have used in your own scripts, or that you call from your scripts, are available in the new release. Testing procedures dependent on packages should be part of your upgrade plan.

Code in database applications can reference objects in the connected database. For example, Oracle Call Interface (OCI) and precompiler applications can submit

anonymous PL/SQL blocks. Triggers in Oracle Forms applications can reference a schema object. Such applications are dependent on the schema objects they reference. Dependency management techniques vary, depending on the development environment. Oracle Database does not automatically track application dependencies.

Related Topics

- *Oracle Database Administrator's Guide*

Upgrading Tables Dependent on Oracle-Maintained Types

Starting with Oracle Database 12*c* Release 2 (12.2) and later releases, you must manually upgrade user tables that depend on Oracle-Maintained types.

If your database has user tables that are dependent on Oracle-Maintained types (for example, AQ queue tables), then run the `utluptabdata.sql` command after the upgrade to carry out `ALTER TABLE UPGRADE` on any user tables affected by changes in Oracle-Maintained types. This change in behavior enables user tables to remain in `READ ONLY` state during an upgrade. Users are prevented from logging into applications using SYSDBA privileges (`AS SYSDBA`), and changing application tables that are dependent on Oracle-Maintained types.

To identify tables that you need to upgrade after the database upgrade completes, connect to the database `AS SYSDBA`, and run the following query:

```
COLUMN owner FORMAT A30
COLUMN table_name FORMAT A30
SELECT DISTINCT owner, table_name
FROM dba_tab_cols
WHERE data_upgraded = 'NO'
ORDER BY 1,2;
```

This query lists all tables that are not listed as `UPGRADED`. However, the `utluptabdata.sql` script only upgrades tables that depend on Oracle-Maintained types. If any tables are listed by the query, then run the `utluptabdata.sql` script to perform `ALTER TABLE UPGRADE` commands on dependent user tables, so that these Oracle-Maintained types are upgraded to the latest version of the type.

You must run the `utluptabdata.sql` script either with a user account with `ALTER` privileges for all of the tables dependent on Oracle-Maintained types, or with a user granted the SYSDBA system privileges, and that is logged in `AS SYSDBA`.

When the parameter `SERVEROUTPUT` is set to `ON`, the `utluptabdata.sql` script displays the names of all upgraded tables, and lists any error encountered during the table upgrade. To set the server output to `ON`, run the following command:

```
SET SERVEROUTPUT ON
@utluptabdata.sql
```

Enabling the New Extended Data Type Capability

Enabling a system to take advantage of the new extended data types requires specific upgrade actions.

Oracle Database 12*c* introduced `MAX_STRING_SIZE` to control the maximum size of `VARCHAR2`, `NVARCHAR2`, and `RAW` data types in SQL. Setting `MAX_STRING_SIZE = EXTENDED` enables the 32767 byte limit introduced in Oracle Database 12*c*.

To be able to set `MAX_STRING_SIZE = EXTENDED`, you must set the `COMPATIBLE` initialization parameter to `12.0.0.0` or higher

In addition, you must run the script `utl32k.sql` script while the database is open in upgrade mode so that you invalidate and recompile objects that are affected by the change in data type sizes. For example:

```
CONNNECT SYS / AS SYSDBA
SHUTDOWN IMMEDIATE;
STARTUP UPGRADE;
ALTER SYSTEM SET max_string_size=extended;
START $ORACLE_HOME/rdbms/admin/utl32k.sql
SHUTDOWN IMMEDIATE;
STARTUP;
```

Caution:

You can change the value of `MAX_STRING_SIZE` from `STANDARD` to `EXTENDED`. However, you cannot change the value of `MAX_STRING_SIZE` from `EXTENDED` to `STANDARD`. By setting `MAX_STRING_SIZE = EXTENDED`, you are taking an explicit action that can introduce application incompatibility in your database.

See Also:

Oracle Database Reference for complete information about `MAX_STRING_SIZE`, including recommendations and procedures

Adjusting Minimum and Maximum for Parallel Execution Servers

Depending on your environment, you can reduce the default setting of the `PARALLEL_MIN_SERVERS` parameter.

In Oracle Database 12*c*, the default for `PARALLEL_MIN_SERVERS` changed from `0` to a value that was provided based on your hardware platform. This change was made to provide sufficient minimal support for parallel execution. If you find that the new default setting is too high, then adjust the setting for your requirements. The default for `PARALLEL_MAX_SERVERS` has not changed. If the default in your old environment is unchanged, then you do not need to take further action.

> **See Also:**
>
> *Oracle Database Reference* for information about `PARALLEL_MIN_SERVERS`

About Recovery Catalog Upgrade After Upgrading Oracle Database

If you use a version of the recovery catalog schema that is older than that required by the RMAN client, then you must upgrade it.

> **See Also:**
>
> - *Oracle Database Backup and Recovery User's Guide* for information on managing an RMAN recovery catalog
> - *Oracle Database Backup and Recovery User's Guide* for complete information about upgrading the recovery catalog and the `UPGRADE CATALOG` command

Upgrading the Time Zone File Version After Upgrading Oracle Database

If the Pre-Upgrade Information Tool instructs you to upgrade the time zone files after completing the database upgrade, then use the `DBMS_DST` PL/SQL package to upgrade the time zone file.

Oracle Database supplies multiple versions of time zone files. There are two types of files associated with each time zone file: a large file, which contains all the time zones defined in the database, and a small file, which contains only the most commonly used time zones. The large versions are designated as `timezlrg_`*`version_number`*`.dat`. The small versions are designated as `timezone_`*`version_number`*`.dat`. The files are located in the `oracore/zoneinfo` subdirectory under the Oracle Database home directory.

Related Topics

- *Oracle Database Globalization Support Guide*
- https://support.oracle.com/rs?type=doc&id=1585343.1

Upgrading Statistics Tables Created by the DBMS_STATS Package After Upgrading Oracle Database

If you created statistics tables using the `DBMS_STATS.CREATE_STAT_TABLE` procedure, then upgrade these tables by running `DBMS_STATS.UPGRADE_STAT_TABLE`.

In the following example, `green` is the owner of the statistics table and `STAT_TABLE` is the name of the statistics table.

```
EXECUTE DBMS_STATS.UPGRADE_STAT_TABLE('green', 'stat_table');
```

Perform this procedure for each statistics table.

> **See Also:**
>
> *Oracle Database PL/SQL Packages and Types Reference* for information about the DBMS_STATS package

Upgrading Externally Authenticated SSL Users After Upgrading Oracle Database

If you are upgrading from Oracle9*i* Release 2 (9.2) or Oracle Database 10*g* Release 1 (10.1), and you are using externally authenticated SSL users, then you must run the SSL external users conversion (`extusrupgrade`) script to upgrade those users.

The `extusrupgrade` script has the following syntax, where *`ORACLE_HOME`* is the Oracle database home, *`hostname`* is the name of the host on which the database is running, *`port_no`* is the listener port number, *`sid`* is the system identifier for the database instance, and *`db_admin`* is the database administrative user with privileges to modify user accounts.

```
ORACLE_HOME/rdbms/bin/extusrupgrade --dbconnectstring
hostname:port_no:sid --dbuser db_admin --dbuserpassword
password -a
```

For example:

```
extusrupgrade --dbconnectstring dlsun88:1521:10gR2 --dbuser system --
dbuserpassword manager -a
```

> **Note:**
>
> If you are upgrading from Oracle Database 10*g* Release 2 (10.2) or later, then you are not required to run the `extusrupgrade` script.

> **See Also:**
>
> *Oracle Database Enterprise User Security Administrator's Guide* for more information about the `extusrupgrade` script

Configuring the FTP and HTTP Ports and HTTP Authentication for Oracle XML DB

Oracle Database Configuration Assistant (DBCA) does not configure ports for Oracle XML DB on Oracle Database 12*c* and later release upgrades use digest authentication.

Oracle recommends that when you configure ports, you also configure the authentication for HTTP for accessing Oracle XML DB Repository to take advantage of improved security features.

Starting with Oracle Database 12*c*, Oracle enhanced database security by supporting digest authentication. Digest authentication is an industry-standard protocol that is commonly used with the HTTP protocol. It is supported by most HTTP clients. Digest authentication ensures that passwords are always transmitted in a secure manner, even when an encrypted (HTTPS) connection is not in use. Support for digest authentication enables organizations to deploy applications that use Oracle XML DB HTTP, without having to worry about passwords being compromised. Digest authentication support in Oracle XML DB also ensures that the Oracle XML DB HTTP server remains compatible with Microsoft Web Folders WebDAV clients.

After installing or upgrading for the new release, you must manually configure the FTP and HTTP ports for Oracle XML DB as follows:

1. Use `DBMS_XDB_CONFIG.setHTTPPort(`*`HTTP_port_number`*`)` to set the HTTP port for Oracle XML DB:

   ```
   SQL> exec DBMS_XDB_CONFIG.setHTTPPort(port_number);
   ```

2. Use `DBMS_XDB_CONFIG.setFTPPort(`*`FTP_port_number`*`)` to set the FTP port for Oracle XML DB:

   ```
   SQL> exec DBMS_XDB_CONFIG.setFTPPort(FTP_port_number);
   ```

   > **Note:**
   >
   > You can query the port numbers to use for FTP and HTTP in the procedure by using `DBMS_XDB_CONFIG.getFTPPort` and `DBMS_XDB_CONFIG.getHTTPPort` respectively.

3. To see all the used port numbers, query `DBMS_XDB_CONFIG.usedport`.

> **See Also:**
>
> *Oracle XML DB Developer's Guide* for more information about accessing the Oracle XML DB Repository data using FTP, HTTP, HTTPS, and WebDAV protocols

Install Oracle Text Supplied Knowledge Bases After Upgrading Oracle Database

After an Oracle Database upgrade, all user extensions to the Oracle Text supplied knowledge bases must be regenerated.

Regenerating the user extensions affect all databases installed in the given Oracle home.

After an upgrade, the Oracle Text-supplied knowledge bases that are part of the companion products for the new Oracle Database are not immediately available. Any Oracle Text features dependent on the supplied knowledge bases that were available before the upgrade do not function after the upgrade. To re-enable such features, you must install the Oracle Text supplied knowledge bases from the installation media for the new Oracle Database release.

> **See Also:**
>
> - *Oracle Text Application Developer's Guide* for information about Oracle Text-supplied knowledge bases
> - *Oracle Database Installation Guide* for companion products

Update Oracle Application Express Configuration After Upgrading Oracle Database

Oracle Application Express affects upgrade procedures, depending on the Oracle Application Express release and your database installation type.

If the Oracle Database release that you upgrade includes Oracle Application Express release 3.2 or later, then you do not need to carry out additional configuration after upgrading to the new Oracle Database release. However, if Oracle Application Express is in the registry, so that Oracle Application Express is included in the upgrade, then set the `open_cursors` parameter to a minimum of 200.

If the Oracle Database you upgrade is an Oracle Express Edition database, then it contains an earlier release of Oracle Application Express that is tailored for the Oracle Express Edition environment. The latest Oracle Application Express release is automatically installed during the upgrade. You must complete a series of postinstallation steps to configure Oracle Application Express for use with the new Oracle Database release.

> **See Also:**
>
> - *Oracle Application Express Installation Guide* for postinstallation tasks for Oracle Application Express
> - `http://www.oracle.com/technetwork/developer-tools/apex/overview/index.html`

Configure Access Control Lists (ACLs) to External Network Services

Oracle Database 12*c* and later releases include fine-grained access control to the `UTL_TCP`, `UTL_SMTP`, `UTL_MAIL`, `UTL_HTTP`, or `UTL_INADDR` packages.

If you have applications that use these packages, then after upgrading Oracle Database you must configure network access control lists (ACLs) in the database before the affected packages can work as they did in earlier releases. Without the ACLs, your applications can fail with the error "ORA-24247: network access denied by access control list (ACL)."

> **See Also:**
>
> *Oracle Database Security Guide* for more complicated situations, such as connecting some users to host A and other users to host B

Check for the SQLNET.ALLOWED_LOGON_VERSION Parameter Behavior

Connections to Oracle Database from clients earlier than release 10*g* fail with the error `ORA-28040: No matching authentication protocol`.

Starting with Oracle Database 18c, the default value for the SQLNET.ALLOWED_LOGON_VERSION parameter changes from 11 in Oracle Database 12*c* (12.2) to 12 in Oracle Database 18c. The use of this parameter is deprecated.

SQLNET.ALLOWED_LOGON_VERSION is now replaced with the SQLNET.ALLOWED_LOGON_VERSION_SERVER and SQLNET.ALLOWED_LOGON_VERSION_CLIENT parameters. If you have not explicitly set the SQLNET.ALLOWED_LOGON_VERSION_SERVER parameter in the upgraded database, then connections from clients earlier than release 10*g* fail with the error `ORA-28040: No matching authentication protocol`. For better security, check the password verifiers of your database users, and then configure the database to use the correct password verifier by setting the SQLNET.ALLOWED_LOGON_VERSION_SERVER and SQLNET.ALLOWED_LOGON_VERSION_CLIENT parameters.

If you have password-protected roles (secure roles) in your existing database, and if you upgrade to Oracle Database 18c with the default `SQLNET.ALLOWED_LOGON_VERSION_SERVER` setting of 12, because those secure roles

only have release 10*g* verifiers, then the password for each secure role must be reset by the administrator so that the secure roles can remain usable after the upgrade.

See Also:

- *Oracle Database Security Guide* for information about ensuring against password security threats
- *Oracle Database Security Guide* for information about setting the password versions of users

Recommended and Best Practices to Complete After Upgrading Oracle Database

Oracle recommends that you complete these good practices guidelines for updating Oracle Database. These practices are recommended for both manual and DBUA upgrades.

- Back Up the Database
  Perform a full backup of the production database.
- Scenario Non-CDB Running the postupgrade_fixups.sql Script
  Review this procedure to understand how to use the `postupgrade_fixups.sql` scripts for Non-CDB databases.
- Gathering Dictionary Statistics After Upgrading
  To help to assure good performance, use this procedure to gather dictionary statistics after completing your upgrade.
- Regathering Fixed Objects Statistics with DBMS_STATS
  After an upgrade, or after other database configuration changes, Oracle strongly recommends that you regather fixed object statistics after you have run representative workloads on Oracle Database.
- Reset Passwords to Enforce Case-Sensitivity
  For upgraded databases, improve security by using case-sensitive passwords for default user accounts and user accounts.
- Add New Features as Appropriate
  Review new features as part of your database upgrade plan.
- Develop New Administrative Procedures as Needed
  Plan a review of your scripts and procedures, and change as needed.
- Set Threshold Values for Tablespace Alerts
  After an upgrade, thresholds for Oracle Database 18c Tablespace Alerts are set to null, disabling the alerts.
- Migrating From Rollback Segments To Automatic Undo Mode
  If your database release is earlier than Oracle Database 11*g*, then you must migrate the database that is being upgraded from using rollback segments (manual undo management) to automatic undo management.

- Configure Oracle Data Guard Broker
  `InitialConnectIdentifier` is replaced by `DGConnectIdentifier`, which affects upgrades from Oracle Database 10*g*.
- Migrating Tables from the LONG Data Type to the LOB Data Type
  You can use the `ALTER TABLE` statement to change the data type of a `LONG` column to `CLOB` and that of a `LONG RAW` column to `BLOB`.
- Identify Oracle Text Indexes for Rebuilds
  You can run a script that helps you to identify Oracle Text index indexes with token tables that can benefit by being rebuilt after upgrading to the new Oracle Database release..
- About Testing the Upgraded Production Oracle Database
  Repeat tests on your production database that you carried out on your test database to ensure applications operate as expected.

Back Up the Database

Perform a full backup of the production database.

Although this step is not required, Oracle strongly recommends that you back up your production database.

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* for details about backing up a database with RMAN

Scenario Non-CDB Running the postupgrade_fixups.sql Script

Review this procedure to understand how to use the `postupgrade_fixups.sql` scripts for Non-CDB databases.

The postupgrade fixup scripts are generated when you run the Pre-Upgrade Information Tool (`preupgrade.jar`). Run the postupgrade scripts any time after completing an upgrade. For Non-CDB databases, the postupgrade fixup scripts provide general warnings, errors, and informational recommendations.

You can run the script either by using the `catcon.pl` utility, or by using SQL*Plus.

The location of the postupgrade SQL scripts and log files depends on how you set output folders, or define the Oracle base environment variable. The postupgrade fixup scripts are placed in the same directory path as the preupgrade fixup scripts.

If you specify an output directory by using the `dir` option with the Pre-Upgrade Information Tool, then the output logs and files are placed under that directory in the file path /`cfgtoollogs`/*`dbunique_name`*/`preupgrade`, where *`dbunique_name`* is the name of your source Oracle Database. If you do not specify an output directory when you run the Pre-Upgrade Information Tool, then the output is directed to one of the following default locations:

- If you do not specify an output directory with `DIR`, but you have set an Oracle base environment variable, then the generated scripts and log files are created in the following file path:

  Oracle-base`/cfgtoollogs/`*dbunique_name*`/preupgrade`

- If you do not specify an output directory, and you have not defined an Oracle base environment variable, then the generated scripts and log files are created in the following file path:

  Oracle-home`/cfgtoollogs/`*dbunique_name*`/preupgrade`

The postupgrade fixup scripts that the Pre-Upgrade Information Tool creates depend on whether your source database is a Non-CDB database, or a CDB database:

- Non-CDB: `postupgrade_fixups.sql`

Example 10-4 Example of Spooling Postupgrade Fixup Results for a Non-CDB Oracle Database

Set the system to spool results to a log file so you can read the output. However, do not spool to the `admin` directory:

```
SQL> SPOOL postupgrade.log
SQL> @postupgrade_fixups.sql
SQL> SPOOL OFF
```

Turn off the spooling of script results to the log file:

```
SPOOL OFF
```

Gathering Dictionary Statistics After Upgrading

To help to assure good performance, use this procedure to gather dictionary statistics after completing your upgrade.

Oracle recommends that you gather dictionary statistics both before and after upgrading the database, because Data Dictionary tables are modified and created during the upgrade. With Oracle Database 12*c* release 2 (12.2) and later releases, you gather statistics as a manual procedure after the upgrade, when you bring the database up in normal mode.

- Non-CDB Oracle Database: Oracle recommends that you use the `DBMS_STATS.GATHER_DICTIONARY_STATS` procedure to gather these statistics. For example, enter the following SQL statement:

  ```
  SQL> EXEC DBMS_STATS.GATHER_DICTIONARY_STATS;
  ```

- CDB: Oracle recommends that you use `catcon` to gather Data Dictionary statistics across the entire multitenant architecture

  To gather dictionary statistics for all PDBs in a container database, use the following syntax

  ```
  $ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -l /tmp
  -b gatherstats -- --x"exec dbms_stats.gather_dictionary_stats"
  ```

To gather dictionary statistics on a particular PDB, use syntax similar to the following:

```
$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl -l /tmp -c
'SALES1' -b gatherstats -- --x"exec dbms_stats.gather_dictionary_stats"
```

In the preceding example the `-c SALES1` option specifies a PDB inclusion list for the command that you run, specifying the database named `SALES1`. The option `-b gatherstats` specifies the base name for the logs. The option `--x` specifies the SQL command that you want to execute. The SQL command itself is inside the quotation marks.

Related Topics

- *Oracle Database PL/SQL Packages and Types Reference*

Regathering Fixed Objects Statistics with DBMS_STATS

After an upgrade, or after other database configuration changes, Oracle strongly recommends that you regather fixed object statistics after you have run representative workloads on Oracle Database.

Fixed objects are the `X$` tables and their indexes. `V$` performance views are defined through `X$` tables. Gathering fixed object statistics is valuable for database performance, because these statistics help the optimizer generate good execution plans, which can improve database performance. Failing to obtain representative statistics can lead to suboptimal execution plans, which can cause significant performance problems.

Gather fixed objects statistics by using the `DBMS_STATS.GATHER_FIXED_OBJECTS_STATS` PL/SQL procedure. `DBMS_STATS.GATHER_FIXED_OBJECTS_STATS` also displays recommendations for removing all hidden or underscore parameters and events from the `INIT.ORA` or `SPFILE`.

Because of the transient nature of `X$` tables, you must gather fixed objects statistics when there is a representative workload on the system. If you cannot gather fixed objects statistics during peak load, then Oracle recommends that you do it after the system is in a runtime state, and the most important types of fixed object tables are populated.

To gather statistics for fixed objects, run the following PL/SQL procedure:

```
SQL> execute dbms_stats.gather_fixed_objects_stats;
```

Reset Passwords to Enforce Case-Sensitivity

For upgraded databases, improve security by using case-sensitive passwords for default user accounts and user accounts.

For greater security, Oracle recommends that you enable case sensitivity in passwords. Case sensitivity increases the security of passwords by requiring that users enter both the correct password string, and the correct case for each character in that string. For example, the password `hPP5620qr` fails if it is entered as `hpp5620QR` or `hPp5620Qr`.

To secure your database, create passwords in a secure fashion. If you have default passwords in your database, then change these passwords. By default, case sensitivity is enforce when you change passwords. Every password should satisfy the Oracle recommended password requirements, including passwords for predefined user accounts.

For new databases created after the upgrade, there are no additional tasks or management requirements.

Existing Database Requirements and Guidelines for Password Changes

- If the default security settings for Oracle Database 12*c* release 1 (12.1) and later are in place, then passwords must be at least eight characters, and passwords such as `welcome` and `oracle` are not allowed.
- The `IGNORECASE` parameter is deprecated. Do not use this parameter.
- For existing databases, to take advantage of password case-sensitivity, you must reset the passwords of existing users during the database upgrade procedure. Reset the password for each existing database user with an `ALTER USER` statement.
- Query the `PASSWORD_VERSIONS` column of `DBA_USERS` to find the `USERNAME` of accounts that only have the 10G password version, and do not have either the `11G` or the `12C` password version. Reset the password for any account that has only the `10G` password version.
- Finding and Resetting User Passwords That Use the 10G Password Version
  For better security, find and reset passwords for user accounts that use the `10G` password version so that they use later, more secure password versions.

> **See Also:**
>
> - *Oracle Database Security Guide* for more information about password case sensitivity
> - *Oracle Database Security Guide* for more information about password strength

Finding and Resetting User Passwords That Use the 10G Password Version

For better security, find and reset passwords for user accounts that use the `10G` password version so that they use later, more secure password versions.

Finding All Password Versions of Current Users

You can query the `DBA_USERS` data dictionary view to find a list of all the password versions configured for user accounts.

For example:

```
SELECT USERNAME,PASSWORD_VERSIONS FROM DBA_USERS;

USERNAME                       PASSWORD_VERSIONS
```

```
------------------------------ -----------------
JONES                          10G 11G 12C
ADAMS                          10G 11G
CLARK                          10G 11G
PRESTON                        11G
BLAKE                          10G
```

The PASSWORD_VERSIONS column shows the list of password versions that exist for the account. 10G refers to the earlier case-insensitive Oracle password version, 11G refers to the SHA-1-based password version, and 12C refers to the SHA-2-based SHA-512 password version.

- User jones: The password for this user was reset in Oracle Database 12*c* Release 12.1 when the SQLNET.ALLOWED_LOGON_VERSION_SERVER parameter setting was 8. This enabled all three password versions to be created.
- Users adams and clark: The passwords for these accounts were originally created in Oracle Database 10*g* and then reset in Oracle Database 11*g*. The Oracle Database 11*g* software was using the default SQLNET.ALLOWED_LOGON_VERSION setting of 8 at that time. Because case insensitivity is enabled by default, their passwords are now case sensitive, as is the password for preston.
- User preston: This account was imported from an Oracle Database 11*g* database that was running in Exclusive Mode (SQLNET.ALLOWED_LOGON_VERSION = 12).
- User blake: This account still uses the Oracle Database 10*g* password version. At this stage, user blake is prevented from logging in.

Resetting User Passwords That Use the 10G Password Version

For better security, remove the 10G password version from the accounts of all users. In the following procedure, to reset the passwords of users who have the 10G password version, you must temporarily relax the SQLNET.ALLOWED_LOGON_VERSION_SERVER setting, which controls the ability level required of clients before login can be allowed. Relaxing the setting enables these users to log in and change their passwords, and hence generate the newer password versions in addition to the 10G password version. Afterward, you can set the database to use Exclusive Mode and ensure that the clients have the O5L_NP capability. Then the users can reset their passwords again, so that their password versions no longer include 10G, but only have the more secure 11G and 12C password versions.

1. Query the DBA_USERS view to find users who only use the 10G password version.

   ```
   SELECT USERNAME FROM DBA_USERS
   WHERE ( PASSWORD_VERSIONS = '10G '
   OR PASSWORD_VERSIONS = '10G HTTP ')
   AND USERNAME <> 'ANONYMOUS';
   ```

2. Configure the database so that it does not run in Exclusive Mode, as follows:
   a. Edit the SQLNET.ALLOWED_LOGON_VERSION_SERVER setting in the sqlnet.ora file so that it is more permissive than the default. For example:

      ```
      SQLNET.ALLOWED_LOGON_VERSION_SERVER=11
      ```

   b. Restart the database.
3. Expire the users that you found when you queried the DBA_USERS view to find users who only use the 10G password version.

You must expire the users who have only the 10G password version, and do not have one or both of the 11G or 12C password versions.

For example:

```
ALTER USER username PASSWORD EXPIRE;
```

4. Ask the users whose passwords you expired to log in.

   When the users log in, they are prompted to change their passwords. The database generates the missing 11G and 12C password versions for their account, in addition to the 10G password version. The 10G password version continues to be present, because the database is running in the permissive mode.

5. Ensure that the client software with which the users are connecting has the O5L_NP ability.

   All Oracle Database release 11.2.0.3 and later clients have the O5L_NP ability. If you have an earlier Oracle Database client, then you must install the CPUOct2012 patch.

6. After all clients have the O5L_NP capability, set the security for the server back to Exclusive Mode, as follows:

   a. Remove the SEC_CASE_SENSITIVE_LOGON parameter setting from the instance initialization file, or set SEC_CASE_SENSITIVE_LOGON to TRUE.

      ```
      SEC_CASE_SENSITIVE_LOGON = TRUE
      ```

   b. Remove the SQLNET.ALLOWED_LOGON_VERSION_SERVER parameter from the server sqlnet.ora file, or set the value of SQLNET.ALLOWED_LOGON_VERSION_SERVER in the server sqlnet.ora file back to 12, to set it to an Exclusive Mode.

      ```
      SQLNET.ALLOWED_LOGON_VERSION_SERVER = 12
      ```

   c. Restart the database.

7. Find the accounts that still have the 10G password version.

   ```
   SELECT USERNAME FROM DBA_USERS
   WHERE PASSWORD_VERSIONS LIKE '%10G%'
   AND USERNAME <> 'ANONYMOUS';
   ```

8. Expire the accounts that still have the 10G password version.

   ```
   ALTER USER username PASSWORD EXPIRE;
   ```

9. Ask these users to log in to their accounts.

   When the users log in, they are prompted to reset their passwords. The database then generates only the 11G and 12C password versions for their accounts. Because the database is running in Exclusive Mode, the 10G password version is no longer generated.

10. Rerun the following query:

    ```
    SELECT USERNAME FROM DBA_USERS
    WHERE PASSWORD_VERSIONS LIKE '%10G%'
    AND USERNAME <> 'ANONYMOUS';
    ```

    If this query does not return any results, then it means that no user accounts have the 10G password version. Hence, the database is running in a more secure mode than in previous releases.

Add New Features as Appropriate

Review new features as part of your database upgrade plan.

Oracle Database New Features Guide describes many of the new features available in the new Oracle Database release. Determine which of these new features can benefit the database and applications. You can then develop a plan for using these features.

It is not necessary to make any immediate changes to begin using your new Oracle Database software. You can choose to introduce new feature enhancements into your database and applications gradually.

> **See Also:**
>
> *Oracle Database New Features Guide*

Develop New Administrative Procedures as Needed

Plan a review of your scripts and procedures, and change as needed.

After familiarizing yourself with the features of the new Oracle Database release, review your database administration scripts and procedures to determine whether any changes are necessary.

Coordinate your changes to the database with the changes that are necessary for each application. For example, by enabling integrity constraints in the database, you may be able to remove some data checking from your applications.

Set Threshold Values for Tablespace Alerts

After an upgrade, thresholds for Oracle Database 18c Tablespace Alerts are set to null, disabling the alerts.

You must identify tablespaces in the database that are candidates for monitoring, and you must set the appropriate threshold values for these tablespaces.

In newly-created Oracle Database 18c installations, the following values are used as defaults:

- 85% full warning
- 97% full critical

Migrating From Rollback Segments To Automatic Undo Mode

If your database release is earlier than Oracle Database 11*g*, then you must migrate the database that is being upgraded from using rollback segments (manual undo management) to automatic undo management.

Automatic undo management is the default undo space management mode. The `UNDO_MANAGEMENT` initialization parameter specifies which undo space management mode the system should use:

- If `UNDO_MANAGEMENT` is set to `AUTO` (or if `UNDO_MANAGEMENT` is not set), then the database instance starts in automatic undo management mode.

  A null `UNDO_MANAGEMENT` initialization parameter defaults to automatic undo management mode in Oracle Database 11*g* Release 1 (11.1) and later. In earlier releases it defaults to manual undo management mode. Use caution when upgrading earlier releases.
- If `UNDO_MANAGEMENT` is set to `MANUAL`, then undo space is allocated externally as rollback segments.

1. Set the `UNDO_MANAGEMENT` parameter to `UNDO_MANAGEMENT=MANUAL`.
2. Start the instance again and run through a standard business cycle to obtain a representative workload. Assess the workload, and compute the size of the undo tablespace that you require for automatic undo management.
3. After the standard business cycle completes, run the following function to collect the undo tablespace size, and to help with the sizing of the undo tablespace. You require SYSDBA privileges to run this function.

   ```
   DECLARE
      utbsiz_in_MB NUMBER;
   BEGIN
      utbsiz_in_MB := DBMS_UNDO_ADV.RBU_MIGRATION;
   end;
   /
   ```

   This function runs a PL/SQL procedure that provides information on how to size your new undo tablespace based on the configuration and usage of the rollback segments in your system. The function returns the sizing information directly.
4. Create an undo tablespace of the required size and turn on the automatic undo management by setting `UNDO_MANAGEMENT=AUTO` or by removing the parameter.
5. For Oracle RAC configurations, repeat these steps on all instances.

Configure Oracle Data Guard Broker

`InitialConnectIdentifier` is replaced by `DGConnectIdentifier`, which affects upgrades from Oracle Database 10*g*.

The value of `DGConnectIdentifier` is used for all Data Guard network traffic, all of the time. If you are upgrading an Oracle Database release 10*g* configuration, which requires you to first upgrade to Oracle Database 11*g*, then the value that exists for `InitialConnectIdentifier` is retained as the new value for `DGConnectIdentifier` for the database. When upgrading an Oracle Real Application Clusters (Oracle RAC)

database, database administrators must ensure that the value for the `InitialConnectIdentifier` property reaches all instances.

In a migration, you need to perform this step only for the standby database.

Migrating Tables from the LONG Data Type to the LOB Data Type

You can use the `ALTER TABLE` statement to change the data type of a `LONG` column to `CLOB` and that of a `LONG RAW` column to `BLOB`.

The `LOB` data types (`BFILE`, `BLOB`, `CLOB`, and `NCLOB`) can provide many advantages over `LONG` data types.

In the following example, the `LONG` column named `long_col` in table `long_tab` is changed to data type `CLOB`:

```
SQL> ALTER TABLE Long_tab MODIFY ( long_col CLOB );
```

After using this method to change `LONG` columns to LOBs, all the existing constraints and triggers on the table are still usable. However, all the indexes, including Domain indexes and Functional indexes, on all columns of the table become unusable and must be rebuilt using an `ALTER INDEX...REBUILD` statement. Also, the Domain indexes on the `LONG` column must be dropped before changing the `LONG` column to a LOB.

> **See Also:**
>
> *Oracle Database SecureFiles and Large Objects Developer's Guide* for information about modifying applications to use LOB data

Identify Oracle Text Indexes for Rebuilds

You can run a script that helps you to identify Oracle Text index indexes with token tables that can benefit by being rebuilt after upgrading to the new Oracle Database release..

When you upgrade from Oracle Database 12c release 1 (12.2.0.1) to Oracle Database 18c and later releases, the Oracle Text token tables (`$I`, `$P`, and so on) are expanded from 64 bytes to 255 bytes. However, if you have indexes with existing token tables using the smaller size range, then the Oracle Text indexes cannot take advantage of this widened token column range. You must rebuild the indexes to use the 255 byte size range. Oracle provides a script that can assist you to identify indexes that can benefit by being rebuilt.

Obtain the script from My Oracle Support:

https://support.oracle.com/rs?type=doc&id=2287094.1

About Testing the Upgraded Production Oracle Database

Repeat tests on your production database that you carried out on your test database to ensure applications operate as expected.

If you upgraded a test database to the new Oracle Database release, and then tested it, then you can now repeat those tests on the production database that you upgraded to the new Oracle Database release. Compare the results, noting anomalies. Repeat the test upgrade as many times as necessary.

To verify that your applications operate properly with a new Oracle Database release, test the newly upgraded production database with your existing applications. You also can test enhanced functions by adding available Oracle Database features, and then testing them. However, first ensure that the applications operate in the same manner as they did before the upgrade.

Index

S

T

U

W