Oracle® Database

Upgrading Non-CDBs on the Same System

21c
F52223-03
March 2023

Oracle Database Upgrading Non-CDBs on the Same System, 21c

F52223-03

Primary Author: Douglas Williams

Contributing Authors: Sunil Surabhi, Nirmal Kumar, Daniel Overby Hansen

Contributors: Roy Swonger, Byron Motta, Hector Vieyra Farfan, Carol Tagliaferri, Mike Dietrich, Cindy Lim

Contents

4 Post-Upgrade Tasks for Oracle Database

Preface

This scenario document explains how to upgrade Oracle Grid Infrastructure for a standalone server (Oracle Restart) to a later release.

Use Case Scenario for this Document

Use this scenario document to assist you to upgrade a single-instance on-premises Oracle Database on the same server using the AutoUpgrade Utility.

Prerequisites for this Scenario

Determine that you do not have applications or security measures that require you to carry out additional tasks not covered in this simple upgrade document.

Outline for this Scenario

1. **Checking Compatibility Before Upgrading Oracle Database**. Ensure that you can update your earlier release Oracle Database to the new Oracle Database release.
2. **Preparing to Upgrade Oracle Database**. Complete this minimal list of preupgrade steps.
3. **Upgrading Oracle Database**. Use the AutoUpgrade utility to perform the upgrade.
4. **Post-Upgrade Tasks for Oracle Database**. Complete this minimal list of post-upgrade checks.

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

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 find the `COMPATIBLE` initialization parameter value set for your database.
- Values for the COMPATIBLE Initialization Parameter in Oracle Database
  Review to find the default and minimum values for the `COMPATIBLE` initialization parameter for Oracle Database 21c.

Checking the Compatibility Level of Oracle Database

Use this SQL query to find the `COMPATIBLE` initialization parameter value set for your database.

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

Values for the COMPATIBLE Initialization Parameter in Oracle Database

Review to find the default and minimum values for the `COMPATIBLE` initialization parameter for Oracle Database 21c.

Default and Minimum COMPATIBLE Parameter Values

The minimum supported release for direct upgrade to Oracle Database 21c is Oracle Database 12c Release 2 (12.2). The minimum `COMPATIBLE` parameter value for Oracle Database 21c is `12.2.0`. The default value for the `COMPATIBLE` parameter is `21.0.0`. Before you use a direct upgrade to Oracle Database 21c, you must set the `COMPATIBLE` parameter on your source Oracle Database release to at least `12.2.0`.

The `COMPATIBLE` parameter should not be changed for a Release Update (RU) or a Release Update Revision (RUR), either for CDB or Non-CDB instances. The following table lists the default and minimum values for the `COMPATIBLE` parameter in Oracle Database 21c, compared to earlier releases supported for direct upgrade:

Caution:

After the `COMPATIBLE` parameter is increased, database downgrade is not possible.

When you plug in an earlier release PDB to a later release CDB where `COMPATIBLE` is set to a later release than the earlier release PDB, and you upgrade the PDB by using an unplug/plug/upgrade procedure, the `COMPATIBILE` setting of the upgraded PDB is automatically increased to the `COMPATIBLE` setting of the later release CDB.

Do not alter the `COMPATIBLE` parameter to a value other than a default release value. Use only one of the default values listed in the following table.

Table 1-1 The COMPATIBLE Initialization Parameter

Oracle Database Release	Default Value	Minimum Value
Oracle Database 21c	21.0.0	12.2.0
Oracle Database 19c	19.0.0	11.2.0
Oracle Database 18c	18.0.0	11.2.0
Oracle Database 12c Release 2 (12.2)	12.2.0	11.2.0

2

Preparing to Upgrade Oracle Database

Before you upgrade Oracle Database, review new features, and carry out procedures to prepare your database for upgrade.

> **Note:**
>
> 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 the target Oracle home, and then install the new Oracle Database release software for single-instance.
- About Oracle Database AutoUpgrade
  The AutoUpgrade utility is designed to automate the upgrade process, both before starting upgrades, during upgrade deployments, and during postupgrade checks and configuration migration
- 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.
- About the AutoUpgrade Analyze Processing Mode
  The AutoUpgrade Analyze (`analyze`) processing mode checks your database to see if it is ready for upgrade.
- AutoUpgrade with Source and Target Database Homes on Same Server (Typical)
  When your Oracle Database Source and Target Oracle homes are installed on the same physical server, use this example.
- Preparing for Upgrades of Databases with Oracle Database Vault
  If the Oracle Database you plan to upgrade uses Oracle Database Vault, then you must disable Oracle Database Vault before starting the upgrade.

About Oracle Database AutoUpgrade

The AutoUpgrade utility is designed to automate the upgrade process, both before starting upgrades, during upgrade deployments, and during postupgrade checks and configuration migration

When you perform upgrades, Oracle recommends that you download the most recent version of the AutoUpgrade Utility from My Oracle Support Document 2485457.1, and use `autoupgrade.jar` to prepare for and to deploy your upgrade. You use AutoUpgrade after you have downloaded binaries for the new Oracle Database release, and set up new release Oracle homes. When you use AutoUpgrade, you can upgrade multiple Oracle Database

deployments at the same time, using a single configuration file, customized as needed for each database deployment.

The `autoupgrade.jar` file exists by default in the Oracle home. However, before you use AutoUpgrade, Oracle strongly recommends that you download the latest AutoUpgrade version. AutoUpgrade is included with each release update (RU), but the most recent AutoUpgrade version is always available from My Oracle Support Document 2485457.1.

> **Note:**
>
> AutoUpgrade is available for Oracle Database Enterprise Edition, and Oracle Database Standard Edition. It is not available for Oracle Database Express Edition.

Preventing Issues: Analyze and Fixup Modes

Before the upgrade, in Analyze mode, the AutoUpgrade utility performs read-only analysis of databases before upgrade, so that it can identify issues that require fixing. You can run the utility during normal database operations. In Fixup Mode, the AutoUpgrade utility detects and identifies both fixes that require manual intervention, and fixes that the AutoUpgrade utility can perform during the upgrade deployment phase.

Simplifying Upgrades: Deploy and Upgrade Modes

In Deploy phase, the AutoUpgrade utility modifies the databases you indicate in your configuration file. It enables you to call your own custom scripts during the upgrade to configure databases. In many cases, the AutoUpgrade utility can perform automatic fixes to databases during the upgrade process without requiring manual intervention.

Deploy and Upgrade Postupgrade Checks and Fixes

After an upgrade completes with either Deploy or Upgrade modes, AutoUpgrade performs postupgrade checks. It provides a process where you can enable your custom scripts to be run on each of the upgraded databases, in accordance with the configuration instructions you provide in the AutoUpgrade configuration file, and also can run automatic postupgrade fixups as part of the postupgrade process. In Deploy mode, AutoUpgrade also confirms that the upgrade has succeeded, and copies database files such as `sqlnet.ora`, `tnsname.ora`, and `listener.ora` from the source home to the target home. After these actions are complete, the upgraded Oracle Database release is started in the new Oracle home.

Related Topics

- My Oracle Support Document 2485457.1

About the AutoUpgrade Analyze Processing Mode

The AutoUpgrade Analyze (`analyze`) processing mode checks your database to see if it is ready for upgrade.

When you run AutoUpgrade in Analyze mode, AutoUpgrade only reads data from the database, and does not perform any updates to the database. You can run

AutoUpgrade using the Analyze mode during normal business hours. You can run AutoUpgrade in Analyze mode on your source Oracle Database home before you have set up your target release Oracle Database home.

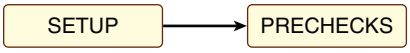
You start AutoUpgrade in Analyze mode using the following syntax, where *Java-8-home* is the location of your Java 8 distribution, or the environment variable set for the Java 8 home, and *path/yourconfig.txt* is the path and filename of your configuration file:

```
Java-8-home/bin/java -jar autoupgrade.jar -config /path/yourconfig.txt -mode
analyze
```

For example, suppose you have copied the most recent AutoUpgrade release to the new release Oracle home under rdbms/admin, and set an environment variable for that home to `21CHOME`, and copied the configuration file under the Oracle user home, under the directory `/scripts`, and called it `21config.cfg`, you then enter the following command:

```
java -jar $21CHOME/rdbms/admin/autoupgrade.jar -config /scratch/scripts/
21config.cfg -mode analyze -mode analyze
```

Oracle Database Release 12.2 (12.2.0.1) or newer Oracle homes have a valid java version by default.



The AutoUpgrade Analyze mode produces two output files, which are given the name of the system identifier (*SID*) of the database that you check:

- *SID*`.html`: View this file using a web browser.
- *SID*`_preupgrade.log`: View this file using a text editor.

Each report identifies upgrade errors that would occur if you do not correct them, either by running an automatic fixup script, or by manual correction. If errors occur, then they are reported in the user log file, and also in the `status.json` file.

The Analyze mode also generates a status directory in the path `cfgtoollogs/upgrade/auto/status`. This directory contains files that indicate if the analysis was successful or failed. This directory has two JSON files, `status.json` and `progress.json`:

- `status.json` : A high-level status JSON file that contains the final status of the upgrade.
- `progress.json`: A JSON file that contains the current progress of all upgrades being performed on behalf of the configuration file. If errors occur, then they are reported in the log file of the user running AutoUpgrade, and also in the `status.json` file.

If your target database Oracle home is not available on the server, then in your configuration file, you must set the source Oracle home parameters to the same path, so that the AutoUpgrade analyze processing mode can run. For example:

```
#
# Source Home
#
sales3.source_home=d:\app\oracle\product\12.2.0\dbhome_1
#
# Target Oracle Home
```

```
#
sales3.target_home=d:\app\oracle\product\21.0.0\dbhome_1
```

Earlier releases of AutoUpgrade required you to set `target_home`. In later releases of AutoUpgrade, this restriction has been lifted for both Analyze and Fixups modes.

AutoUpgrade with Source and Target Database Homes on Same Server (Typical)

When your Oracle Database Source and Target Oracle homes are installed on the same physical server, use this example.

Context: Source and Target homes are on the same server.

To start the analysis, enter the following command.

```
java -jar autoupgrade.jar -config config.txt -mode analyze
```

The command produces a report that indicates any error conditions that the command finds. Review the error conditions.

To start the deployment of the upgrade, enter the following command:

```
java -jar autoupgrade.jar -config config.txt -mode deploy
```

Preparing for Upgrades of Databases with Oracle Database Vault

If the Oracle Database you plan to upgrade uses Oracle Database Vault, then you must disable Oracle Database Vault before starting the upgrade.

During the upgrade process, if your source Oracle Database uses Oracle Database Vault, then you must first disable Oracle Database Vault before you start the upgrade. .

You have two options you can use:

1. Use a manual procedure: Log on as the common Database Vault (DV) administrator in the `CDB$ROOT` and grant the `DV_PATCH_ADMIN` role to `SYS`, or log in and disable Oracle Database Vault on every container. Procedures vary slightly, depending on your upgrade scenario. This procedure is described in My Oracle Support, "Requirement for Upgrading Database with Database Vault (Doc ID 2757126.1)".
2. Download the latest AutoUpgrade Jar file, and perform the procedure described here.

With either option, when you run AutoUpgrade in Analyze mode, it detects that Oracle Database Vault is enabled, and indicates in its report that you must ensure the prerequisites for Oracle Database Vault and upgrade are met.

Example 2-1 AutoUpgrade Procedure for Databases Using Oracle Database Vault

When you use AutoUpgrade, and your database si configured with Oracle Database Vault, the upgrade procedure is as follows:

1. Disable Oracle Database Vault.
2. Install the new Oracle Database release.
3. Download the latest AutoUpgrade JAR file from My Oracle Support note 2485457.1, and replace the AutoUpgrade JAR file in the new Oracle Database release, in the path *Oracle_home*/rdbms/admin
4. Run the AutoUpgrade utility (or Database Upgrade Assistant), and complete the upgrade.
5. Enable Oracle Database Vault in the upgraded Oracle Database.

Related Topics

- Disabling and Enabling Oracle Database Vault
- Requirement for Upgrading Database with Database Vault (Doc ID 2757126.1)
- AutoUpgrade Tool (Doc ID 2485457.1)

Installing Oracle Software in a New Oracle Home

Choose a new location for the target Oracle home, and then install the new Oracle Database release 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. By using separate installation locations, you can test the upgrade process on a test database before replacing your production environment entirely.

When you upgrade a database, you must install the new Oracle home in a new location (an out-of-place upgrade).

If you are upgrading a PDB by using an unplug/plug upgrade, then the target CDB into which you plug the PDB is the location for the PDB. There is no need to choose a new location for installing the target Oracle homes, because the target CDB already has its Oracle home.

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.

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. 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.

Prepare a Backup Strategy Before Upgrading Oracle Database

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

For Oracle Database Enterprise Edition, the primary fallback mechanism is Flashback Database. However, Flashback Database can't be used to revert an unplug-plug upgrade or PDB conversion. For unplug-plug upgrades, rely on other fallback strategies, such as an RMAN backup.

If you use AutoUpgrade, then Oracle recommends that you specify `target_pdb_copy_option=file_name_convert`, in the AutoUpgrade configuration file, where `file_name_convert` is a convert pattern prefixed to the data files. When you do that, AutoUpgrade directs the database to create copies of the data files before plugging in the database. Choosing to use this method enables you to use the original database as a fallback. However, be aware that when you create data file copies, the upgrade requires additional disk space and extra time.

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?
- Are backups tested to ensure that they are done properly?
- How quickly can backups be restored (including backups in offsite storage)?
- Have disaster 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*.

In addition, to ensure that you are prepared for a downgrade, review the downgrade chapter and complete any preparation steps you may need to prepare for your release.

Related Topics

- Backing Up the Database

- Using Flashback Database and Restore Points

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.

- Release Updates and Requirements for Upgrading Oracle Database
  Before starting upgrades, update your new release Oracle home to the latest Release Update (Update).
- Copying Transparent Encryption Oracle Wallets
  If your database uses Transparent Data Encryption (TDE) and a software keystore, and you rely on the `sqlnet.ora` parameter `ENCRYPTION_WALLET_LOCATION` to locate the TDE software keystore, then ensure that the new release `sqlnet.ora` has a proper configuration.
- Recommendations for Oracle Net Services When Upgrading Oracle Database
  You must ensure that the listener is running in your new release Oracle home.
- Understanding Password Case Sensitivity and Upgrades
  By default, Oracle Database 12c Release 2 (12.2) and later releases use Exclusive Mode authentication protocols. 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.

Release Updates and Requirements for Upgrading Oracle Database

Before starting upgrades, update your new release Oracle home to the latest Release Update (Update).

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

Before you start an upgrade, Oracle strongly recommends that you update your new release Oracle home to the latest quarterly Release Update (Update).

My Oracle Support provides detailed notes about how you can obtain the updates, as well as tools for lifecycle management.. For example:

- My Oracle Support note 2118136.2 contains a download assistant to help you select the updates that you need for your environment. Oracle highly recommends that you start here.
- 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).

Related Topics

- My Oracle Support Note 2118136.2
- My Oracle Support Note 1227443.1

Copying Transparent Encryption Oracle Wallets

If your database uses Transparent Data Encryption (TDE) and a software keystore, and you rely on the `sqlnet.ora` parameter `ENCRYPTION_WALLET_LOCATION` to locate the TDE software keystore, then ensure that the new release `sqlnet.ora` has a proper configuration.

> **Note:**
>
> This procedure using `sqlnet.ora` to configure keystores is deprecated. Oracle recommends that you use the instance initialization file-based approach using the `WALLET_ROOT` and `TDE_CONFIGURATION` initialization parameters. After the upgrade, review and configure the `WALLET_ROOT` initialization parameter.

If you use the `sqlnet.ora` file to configure keystores, then you must copy `sqlnet.ora` and the keystore file manually to a keystore location outside of the Oracle home. Wallets should be stored in a location outside of an Oracle Home. If you move the wallet, then you must update `sqlnet.ora` before starting the upgrade. For auto-login wallets, you must also copy the `cwallet.sso` file manually to the new keystore location.

1. Log in as the user owning the Oracle home software, typically `oracle`.
2. Manually copy the `sqlnet.ora` file, and the keystore file, `ewallet.p12`, to the new release Oracle home.
3. If you have enabled an auto-login wallet, then also copy the `cwallet.sso` file to the new release Oracle home. There is no need to complete the next step. If you have not enabled an auto-login wallet, then proceed to the next step.
4. If you have not enabled an auto-login wallet, then open the Oracle wallet in mount.

   For example:

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

   To create a local auto-login keystore for a software keystore, use the following syntax:

   ```
   ADMINISTER KEY MANAGEMENT CREATE LOCAL AUTO_LOGIN KEYSTORE
   FROM KEYSTORE 'keystore_location'
   IDENTIFIED BY software_keystore_password;
   ```

   In this specification:

   - `LOCAL` enables you to create a local auto-login software keystore. Otherwise, omit this clause if you want the keystore to be accessible by other computers.
   - *`keystore_location`* is the path to the keystore directory location of the keystore that is configured in the `sqlnet.ora` file.

- *software_keystore_password* is the existing password of the configured software keystore.

Related Topics

- About the Keystore Location in the sqlnet.ora File

Recommendations for Oracle Net Services When Upgrading Oracle Database

You must ensure that the listener is running in your new release Oracle home.

If the Oracle Database that you are upgrading does not have a listener configured, then before you start the upgrade, 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. The current listener is backward-compatible with earlier Oracle Database releases.

If you are upgrading Oracle Real Application Clusters Oracle Database, or a release older than Oracle Database 12c, then review the following additional information.

For Oracle RAC database upgrades, the listener normally is migrated during the Grid Infrastructure upgrade. 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 12c Release 2 (12.2) and later releases use Exclusive Mode authentication protocols. 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.

> **Note:**
>
> Starting with Oracle Database 21c, the `SEC_CASE_SENSITIVE_LOGON` parameter is desupported. You must use a case-sensitive password version. If a user with only a `10G` password version is upgraded to Oracle Database 21c, then that user account is locked, until an administrator resets the password.

In previous Oracle Database releases, you could configure the authentication protocol so that it allows case-insensitive password-based authentication by setting `SEC_CASE_SENSITIVE_LOGON=FALSE`. Starting with Oracle Database 12c release 2 (12.2), the default password-based authentication protocol configuration excluded 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 12c 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.

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 11g 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.

Update Accounts Using Case-Insensitive Versions

If you have user accounts that have only the case-insensitive `10G` password version, then 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.

Related Topics

- *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.

> **Note:**
>
> Starting with Oracle Database 21c, the `SEC_CASE_SENSITIVE_LOGON` parameter is desupported. You must use a case-sensitive password version.

If you do not want user accounts authenticated with case-insensitive password versions to be locked out of the database after an upgrade, then before the upgrade, you must identify affected accounts, and ensure that they are using case-sensitive password versions.

Example 2-2 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 2-3 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 2-4 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*

3

Using AutoUpgrade to Upgrade and convert Non-CDBs to PDBs

The AutoUpgrade Utility simplifies the task of upgrading and converting your earlier release Oracle Database to a later Oracle Database release using the multitenant architecture.

- Non-CDB to PDB Upgrade Guidelines and Examples
  Before conversion, back up your datafiles and database, and follow the guidelines for your source Oracle Database release.
- AutoUpgrade Configuration File for Non-CDB Upgrades on the Same System
  Use this example to see how you can upgrade your non-CDB Oracle Database
- AutoUpgrade with Source and Target Database Homes on Same Server (Typical)
  When your Oracle Database Source and Target Oracle homes are installed on the same physical server, use this example.

Non-CDB to PDB Upgrade Guidelines and Examples

Before conversion, back up your datafiles and database, and follow the guidelines for your source Oracle Database release.

To ensure that no data is lost during the conversion, Oracle strongly recommends that allow time in your upgrade plan to implement your backup strategy before you use AutoUpgrade to perform a non-CDB upgrade and conversion.

Guidelines for Upgrade Planning

The non-CDB-to-PDB conversion and upgrade process is not recoverable. To ensure a proper upgrade and conversion, and to reduce unexpected downtime, Oracle strongly recommends that you address any error conditions found during the analyze phase.

If you use the `target_pdb_copy_option` in your configuration file to create copies of your data files, then your existing database is available as a backup. This is a safe option, but will require additional time and disk space. If you do not set the `target_pdb_copy_option` in your AutoUpgrade configuration file, then the database conversion uses the same file location and file names that are used with existing database files. To prevent potential data loss, ensure that your data is backed up, and consider your file placement plans before starting AutoUpgrade.

GRP and Upgrades from Non-CDB to Multitenant Architecture

- During the upgrade, AutoUpgrade creates a guaranteed restore point (GRP) that is available only in the context of the upgrade stage of the AutoUpgrade Deploy workflow. To ensure against any potential data loss, you must implement your backup strategy before starting AutoUpgrade.
- Database conversion from non-CDB to the multitenant architecture is performed during the AutoUpgrade Drain stage. After this stage is complete, the GRP that AutoUpgrade creates is removed, and it is not possible to use the AutoUpgrade `restore` command to

restore the database. In the event that you require a recovery to the earlier non-CDB Oracle Database release, you must be prepared to recover the database manually.

Example 3-1 Upgrading and Converting a Non-CDB to Oracle Database 19c Using Multitenant Architecture

During the Deploy conversion and upgrade workflow, AutoUpgrade creates a GRP, and runs the Prefixup stage. If any part of the Deploy workflow up to the Prefixup stage completion fails, then AutoUpgrade can restore the database back to the GRP created at the start of the deployment,

However, after the Prefixup stage is complete, the upgraded database is plugged in to the target release Oracle Database container database (CDB) to complete conversion. As soon as the non-CDB is plugged into the CDB, the GRP is no longer valid, and is dropped.

If anything goes wrong during the plug-in, and you did not choose to use the `target_pdb_copy_option` in your configuration file to create copies of your data files, then be aware that AutoUpgrade cannot recover and restore the database. In that event, you must restore the database manually.

AutoUpgrade Configuration File for Non-CDB Upgrades on the Same System

Use this example to see how you can upgrade your non-CDB Oracle Database

In this scenario, you upgrade two specific PDBs, without upgrading the other PDBs in the source CDB, To perform the incremental upgrade, you direct AutoUpgrade in the configuration file to unplug the PDBs you specify from an earlier release CDB, plug them into a target release CDB, and then upgrade the earlier release PDBs on the target CDB. This selection of PDBs to unplug, plug in, and upgrade, enables you to perform an incremental upgrade of PDBs on the earlier release CDB to reduce downtime.

The following configuration file identifies the non-CDB database `emp` as the source database. The source database is upgraded, and the upgraded Oracle Database is placed in the new Oracle home `/u01/app/oracle/product/21.1.0/dbhome_1`:

```
global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade
upg1.log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade/employee
upg1.sid=emp
upg1.source_home=/u01/app/oracle/product/12.2.0/dbhome_1
upg1.target_home=/u01/app/oracle/product/21.1.0/dbhome_1
```

In this example, the following local parameters are specified:

- `upg1.log_dir` (optional) specifies that the upgrade directory for this database is placed in the folder employee, under the global AutoUpgrade log directory path.
- `upg1.sid` (required) identifies the Oracle system identifier (SID) of the database that you want to upgrade is `emp`.
- `upg1.source_home` (required) specifies the source Oracle home path.
- `upg1.target_home` (required) specifies the target Oracle home path.

The upgrade job starts immediately when you run AutoUpgrade using this configuration file. By default, a guaranteed restore point is automatically created. This feature protects the upgrade, and enables you to fall back to the earlier release in case of errors.

This configuration file provides you with a minimal use case example. You can choose to add additional parameters to modify the upgrade for your needs. For example, if you want to specify a future start time to run the upgrade, then you can use the local parameter `start_time`. In addition, you can modify default behavior. For example, by default, time zone settings are upgraded as part of the database upgrade. Oracle recommends that you upgrade time zone files as part of the upgrade. However, if you want to defer this upgrade to a later maintenance window, then you can add the local parameter `timezone_upg`.

AutoUpgrade with Source and Target Database Homes on Same Server (Typical)

When your Oracle Database Source and Target Oracle homes are installed on the same physical server, use this example.

Context: Source and Target homes are on the same server.

To start the analysis, enter the following command.

```
java -jar autoupgrade.jar -config config.txt -mode analyze
```

The command produces a report that indicates any error conditions that the command finds. Review the error conditions.

To start the deployment of the upgrade, enter the following command:

```
java -jar autoupgrade.jar -config config.txt -mode deploy
```

4
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, `utlusts.sql`.
- How to Show the Current State of the Oracle Data Dictionary
  To check the state of the Oracle Data Dictionary for diagnosing upgrades and migrations, use one of three methods.
- 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. Except where noted, these practices are recommended for all types of upgrades.

Check the Upgrade With Post-Upgrade Status Tool

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

The Post-Upgrade Status Tool is located in the path `$ORACLE_HOME/rdbms/admin`. The 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

To check the state of the Oracle Data Dictionary for diagnosing upgrades and migrations, use one of three methods.

Example 4-1 Run a SQL Query on DBA_REGISTRY

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_full,status
from dba_registry order by comp_id;
```

Example 4-2 Run SQL Queries to Check for Invalid Objects

To check for invalid objects, you can perform SQL queries, similar to the following examples:

1. This query list all the invalid objects in the database:

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

2. After you have upgraded the database, and you have run `utlrp.sql`, Oracle-maintained objects should be valid.

   To check to ensure Oracle-maintained objects are valid, enter the following query:

```
SQL> select owner, object_name, object_type from
sys.dba_invalid_objects where oracle_maintained='Y';
```

3. To list any invalid application objects in the database, enter the following query:

```
SQL> select owner, object_name, object_type from
sys.dba_invalid_objects where oracle_maintained='N';
```

Example 4-3 Run the `dbupgdiag.sql` Script

(Optional) If you want to obtain further information about the upgrade, you can choose to run the `dbupgdiag.sql` script. The `dbupgdiag.sql` script collects diagnostic information about the status of the database, either before or after the upgrade. Download the script from My Oracle Support note 556610, and run the script as the database `SYS` user. The script generates the diagnostic information in a readable format, in a log file with the name file `db_upg_diag_`*`sid`*`_`*`timestamp`*`.log`, where *`sid`* is the Oracle system identifier for the database, and *`timestamp`* is the time that the file is generated.

For example, where you download and place the script in the directory `/u01/dbupgdiag-script`:

```
/u01/dbupdiag-script/ $ sqlplus / as sysdba
sql> alter session set nls_language='American';
sql> @dbupgdiag.sql
sql> exit
```

You can run the script in SQL*Plus both before the upgrade on the source database, and after the upgrade on the upgraded database. For more information about the script, refer to the instructions and the output example file in My Oracle Support Note 556610.1.

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
  Check that required operating system environment variables point to the directories of the new Oracle Database release.
- 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 can affect applications.
- Upgrading Tables Dependent on Oracle-Maintained Types
  Starting with Oracle Database 12c Release 2 (12.2) and later releases, you must manually upgrade user tables that depend on Oracle-Maintained types.
- 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.
- 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 10g fail with the error `ORA-28040: No matching authentication protocol`.

Setting Environment Variables on Linux and Unix Systems After Manual Upgrades

Check that required operating system environment variables point to the directories of the new Oracle Database release.

Typically, operating system environment variables are set in profiles and shell scripts. Confirm that the following Oracle user environment variables point to the directories of the new Oracle home:

- `ORACLE_HOME`
- `PATH`

Look for other environment variables that refer to the earlier release Oracle home, such as `LD_LIBRARY_PATH`. In general, you should replace all occurrences of the old Oracle home in your environment variables with the new Oracle home paths.

Related Topics

- Step 2: Ensure That the Required Environment Variables Are Set

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 can affect applications.

After the upgrade, if you use AutoUpgrade, review the AutoUpgrade report on invalid objects. If you use a replay upgrade, then check to ensure that any packages that you may 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 12c Release 2 (12.2) and later releases, you must manually upgrade user tables that depend on Oracle-Maintained types.

> **Note:**
>
> If you upgraded using the AutoUpgrade utility, then AutoUpgrade automatically takes care of this task during the upgrade. You do not need to perform this task.

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
```

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

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 APEX 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 10g 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 changed from 11 in Oracle Database 12c (12.2) to 12 in Oracle Database 18c and later releases. 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 10g 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 and later releases with the default `SQLNET.ALLOWED_LOGON_VERSION_SERVER` setting of 12, because those secure roles only have release 10g 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 GuideOracle 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. Except where noted, these practices are recommended for all types of upgrades.

- Back Up the Database
  Oracle strongly recommends that you at least perform a level 1 backup, or if time allows, perform a level 0 backup.

- 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`.
- 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.
- Dropping and Recreating DBMS_SCHEDULER Jobs
  If DBMS_SCHEDULER jobs do not function after upgrading from an earlier release, drop and recreate the jobs.
- Transfer Unified Audit Records After the Upgrade
  Review these topics to understand how you can obtain better performance after you upgrade and migrate to unified auditing
- Enabling Disabled Release Update Bug Fixes in the Upgraded Database
  Because bug fixes in Release Updates that can cause execution plan changes are disabled, Oracle recommends that you enable the disabled bug fixes that you want to use.
- Reset Passwords to Enforce Case-Sensitivity
  For upgraded databases, improve security by using case-sensitive passwords for default user accounts and user accounts.
- 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 12c and later releases. Upgrades use digest authentication.
- 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.
- Migrate Your Upgraded Oracle Databases to Use Unified Auditing
  Traditional auditing is deprecated. To use the full facilities of unified auditing, you must manually migrate to unified auditing.
- 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.
- 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.

- Upgrading the Time Zone File Version After Upgrading Oracle Database
  If the AutoUpgrade preupgrade report instructs you to upgrade the time zone files after completing the database upgrade, and you do not set AutoUpgrade to complete this task for you, then use any of the supported methods to upgrade the time zone file.

Dropping and Recreating DBMS_SCHEDULER Jobs

If DBMS_SCHEDULER jobs do not function after upgrading from an earlier release, drop and recreate the jobs.

If you find that DBMS_SCHEDULER jobs are not functioning after an upgrade. drop and recreate those jobs. This issue can occur even if the upgrade process does not report issues, and system objects are valid.

Transfer Unified Audit Records After the Upgrade

Review these topics to understand how you can obtain better performance after you upgrade and migrate to unified auditing

Enabling Disabled Release Update Bug Fixes in the Upgraded Database

Because bug fixes in Release Updates that can cause execution plan changes are disabled, Oracle recommends that you enable the disabled bug fixes that you want to use.

After you upgrade your database, the bug fix patches that can cause execution plan changes included in the Release Updates are installed disabled by default. These bug fixes will not be activated until you enable the fixes. You can either enable these fixes manually, with `PFILE` or `ALTER SYSTEM` commands, or you can use the `DBMS_OPTIM_BUNDLE` package. Starting with AutoUpgrade 19.12, the `DBMS_OPTIM_BUNDLE` package includes 58 standard fixes. You can now add additional fixes using `DBMS_OPTIM_BUNDLE`. If you add fixes, then the fixes that you add are run in addition to the default fixes.

Oracle strongly recommends that you enable these disabled patches that you want to use in your production system, and run complete workload performance tests using these patches as part of your upgrade test plan.

For more information about using `DBMS_OPTIM_BUNDLE` to enable patches that were disabled because they can change execution plans, see *Oracle Database PL/SQL Packages and Types Reference*, and My Oracle Support note 2147007.1.

Related Topics

- DBMS_OPTIM_BUNDLE
- My Oracle Support Doc ID 2147007.1 Managing "installed but disabled" bug fixes in Database Release Updates using DBMS_OPTIM_BUNDLE

Migrate Your Upgraded Oracle Databases to Use Unified Auditing

Traditional auditing is deprecated. To use the full facilities of unified auditing, you must manually migrate to unified auditing.

Unified Auditing and Traditional Auditing (mixed mode) has been the default auditing mode from Oracle Database 12c onward. Mixed mode auditing was offered to enable you to become familiar with Unified Auditing, and to transition from Traditional Auditing. With the deprecation of Traditional Auditing in Oracle Database 21c, Oracle recommends that you

migrate to Unified Auditing. Refer to the migration procedure in Oracle Database Security Guide.

Related Topics

- How the Unified Auditing Migration Affects Individual Audit Features

Back Up the Database

Oracle strongly recommends that you at least perform a level 1 backup, or if time allows, perform a level 0 backup.

Related Topics

- Backing Up the Database

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

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.

> **Note:**
>
> To provide the most correct fixed object statistics for performance tuning, Oracle strongly recommends that you gather baseline statistics at a point when the system is running with a representative workload. For useful results, never run `DBMS_STATS.GATHER_FIXED_OBJECTS_STATS` immediately after the upgrade.

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 to 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.

Ensure that your database has run representative workloads, and then 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;
```

Related Topics

- Gathering Database Statistics

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. In Oracle Database 21c and later release, the `IGNORECASE` parameter for the `orapwd` file is desupported. All newly created password files are case-sensitive. 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`.

Upgraded password files from earlier Oracle Database releases can retain original case-insensitive passwords. To ensure that password files are case-sensitive, Oracle recommends

that you force case sensitivity by migrating password files from one format to another, using the following syntax:

```
orapwd input_file=input_password _file file=output_password_file
```

To secure your database, create passwords in a secure fashion. If you have default passwords in your database, then change these 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

- Passwords must be at least eight characters, and passwords such as `welcome` and `oracle` are not allowed.
- 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.

Related Topics

- Managing the Complexity of Passwords
- Guidelines for Securing User Accounts and Privileges

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 10g and then reset in Oracle Database 11g. The Oracle Database 11g 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 11g 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

You should 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. If you are in the CDB root, then restart the database (for example, SHUTDOWN IMMEDIATE followed by STARTUP). If you are in a PDB, connect to the root using the SYSDBA administrative privilege, and then enter the following statements:

```
ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE pdb_name OPEN;
```

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 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
      ```

   b. If you are in the CDB root, then restart the database (for example, SHUTDOWN IMMEDIATE followed by STARTUP). If you are in a PDB, connect to the root using the SYSDBA administrative privilege, and then enter the following statements:

      ```
      ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
      ALTER PLUGGABLE DATABASE pdb_name OPEN;
      ```

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.

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 12c and later releases. 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 12c, 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`.

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:**
>
> *Learning Database New Features*

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.

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.

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.

Upgrading the Time Zone File Version After Upgrading Oracle Database

If the AutoUpgrade preupgrade report instructs you to upgrade the time zone files after completing the database upgrade, and you do not set AutoUpgrade to complete this task for you, then use any of the supported methods to upgrade the time zone file.

> **Note:**
>
> By default, if time zone files need to be updated, then AutoUpgrade typically performs this task for you. However, if you explicitly disable the time zone file upgrade in your AutoUpgrade configuration file, then you should perform this task either as part of your upgrade plan, or at a later point in time.

Related Topics

- Datetime Data Types and Time Zone Support
- My Oracle Support Doc ID 1585343.1