Upgrade and Migrate to Oracle Database, Release 23

Methods and Comparison

March, 2025, Version 1
Copyright © 2025, Oracle and/or its affiliates
Public

ORACLE

Table of contents

ORACLE

Introduction

Oracle Database Release 23 includes new features and enhancements, as well as a long-term support commitment, that make it an attractive upgrade target for existing Oracle databases. Moving to Release 23 may be part of an effort that includes moving to newly purchased server hardware or migrating to different storage architectures such as Oracle Automatic Storage Management. Increasingly, upgrading to a new version of Oracle Database may also include migrating to the Oracle Cloud.

Because upgrade and migration scenarios can differ, Oracle provides multiple methods for you to upgrade and migrate your databases. This Technical Brief outlines these upgrade and migration methods. You will learn about different use cases and key factors to consider when choosing the method that best fits your specific requirements.

Comparing Upgrade and Migration

Although the terms are often used as synonyms in other contexts, there is a difference between database upgrade and database migration. Understanding this difference is the first step in choosing the best upgrade or migration method for your project.

Note: The term "migration" can also be used when discussing the move of data from a non-Oracle database into Oracle. This technical brief will cover migrations only when both the source and destination are Oracle databases

Database Upgrade

The act of upgrading an Oracle database involves modifying the data dictionary to be compatible with a newer version of Oracle database software. Typical actions that may be part of a database upgrade include:

- Adding, dropping, or modifying columns in system tables and views
- Creating new system packages or procedures
- Modifying existing system packages or procedures
- Creating, modifying, or dropping database types, users, roles, and privileges
- Modifying seed data that is used by Oracle database components

All of these actions affect the data dictionary of your database. They do not affect the data stored in your user or application tablespaces. Therefore, the sheer volume of data stored in your Oracle database has little or no bearing on a database upgrade.

Database Migration

The term "migration" applies to several different types of changes that can be applied to an Oracle database. In addition to database version, these can include a change to any or all of the following:

- Computer server (hardware or virtualized environment)
- Storage architecture
- Character set
- Operating system
- Schema topology (changing the partitioning scheme)
- Encryption
- Compression
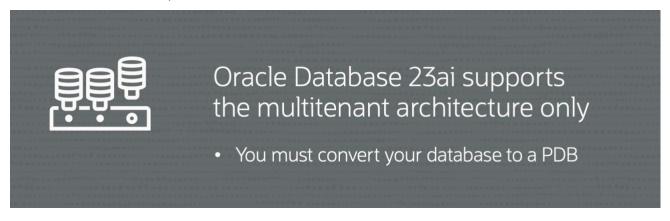- Database architecture (moving from non-CDB to PDB)

Database migration differs from database upgrade in two important ways. First, database migration generally involves moving or modifying the user and application data in the database. This means that the size of your database has a

ORACLE

dramatic impact on your database migration project. Second, any of the above migrations can be performed on a database without upgrading it to a new version. This technical brief will discuss techniques that can be used to perform both an upgrade and a migration simultaneously.

Multitenant Architecture

A multitenant container database is the only supported architecture in Oracle Database 21c and above. Before upgrading to Oracle Database Release 23, the database must be converted to multitenant.

AutoUpgrade, discussed later in this brief, can perform the conversion and the upgrade in the same operation.

The multitenant architecture enables an Oracle database to contain a portable collection of schemas, schema objects, and nonschema objects that appears to an Oracle Net client as a separate database. This self-contained collection is called a pluggable database (PDB). A multitenant container database (CDB) is an Oracle database that includes zero, one, or many user-created PDBs. This new architecture enables customers to easily consolidate multiple databases and introduces another very important type of database migration for customers to consider: migration to a PDB.

In some cases, migrating from traditional database architecture (also called a non-CDB) to a PDB can be combined with other database upgrade or migration tasks into a single operation. In other cases, migration into a PDB may involve an additional step. The steps for migrating an existing or non-CDB database to a PDB will be described under each upgrade or migration technique in this technical brief.

Selecting a Database Upgrade or Migration Method

With several upgrade and migrations available, choosing the best upgrade or migration method for a particular project requires an analysis of several important project characteristics. Each of these characteristics can influence the suitability of a given method when you upgrade or migrate:

- The version from which you are upgrading or migrating, down to the Release Update applied
- The source and destination operating system and version
- The source and destination hardware platforms and their endian characteristics
- Any plans to change the actual data layout or format, such as changing the character set, partitioning, encryption, or compression
- Availability requirements – the amount of downtime allowed for the upgrade or migration project
- The physical size of the database to be migrated
- Whether the destination of the upgrade is a PDB

There is no single upgrade or migration method that is the best option for all possible upgrade and migration scenarios. However, there is a method that is best for any given scenario based on the above factors.

AutoUpgrade Tool

A direct upgrade is one where the Oracle AutoUpgrade Tool (AutoUpgrade) is used to upgrade your database. Direct upgrade is supported when the source database is running one of the releases shown in the following table.

Note: Command-line Upgrade, (catctl.pl/dbupgrade) and the Database Upgrade Assistant (DBUA) are deprecated. Oracle recommends using AutoUpgrade.

Table 1. Direct upgrade paths for Release 23

Source Release	Source Patch Set	Autoupgrade Tool Supported?
Release 21	21.3 and greater	Yes
Release 19	19.3 and greater	Yes

If your Oracle database is on an older release, it is not possible to directly upgrade to Oracle Database Release 23. You must first upgrade to a release supporting direct upgrade to Oracle Database release 23 and then upgrade again to Oracle Database release 23. Alternatively, you can perform a migration to Release 23 using Data Pump or Transportable Tablespaces.

Upgrade and Migration Methods

There are four different upgrade and migration methods described in this brief, and three of these methods have variants that can be used in some situations.

The methods are:

- Oracle AutoUpgrade Tool
- Transportable tablespaces (TTS) export and import, using the Oracle Database feature full transportable export/import, or the traditional TTS mode
- Oracle Data Pump Export/Import, using either dump files or network mode with database links
- The Original Export/Import utilities

The following table summarizes the applicability of these upgrade and migration methods for your upgrade to Oracle Database R, based on the project characteristics listed in the previous section.

Table 2. Database Upgrade and Migration Methods

Method	Complexity	Speed	Minimum Source Version	Move to New Server	Change Endianness	Change Data Layout, character set, encryption, compression

AutoUpgrade Tool	Low	Fastest	19	Yes	No	No
PDB Unplug, Plug, Upgrade, Refreshable Clone	Med	Fastest	19	Yes	No	No
Full Transportable Export/Import	Med	Faster	11.2.0.4	Yes	Yes	No
Transportable Tablespaces	High	Faster	8.1.5	Yes	Yes, starting with 10.1	No
Data Pump expdp/impdp	Med	Fast	10.1	Yes	Yes	Yes
Original export/import	Med	Slow	5	Yes	Yes	Yes

Detailed Upgrade and Migration Method Descriptions

Method 1: AutoUpgrade Tool

The recommended and easiest way to upgrade to Oracle Database release 23 is to upgrade your existing database using the Oracle AutoUpgrade Tool. Because this method can act on your database without creating a copy or a new instance, this is sometimes referred to as upgrading "in place."

The AutoUpgrade Tool automates the upgrade process for Oracle Database releases 19c and higher. Using a single command and one configuration file, AutoUpgrade analyzes one or more source database(s) and applies fixups, performs the upgrade, and completes postupgrade checks and configuration migration. AutoUpgrade is fully restartable and uses parallel capabilities (catctl.) It supports all operating systems running Oracle Database servers, both Enterprise Edition and Standard Edition 2, and Real Application Cluster (RAC) database upgrades.

AutoUpgrade supports non-CDB and CDB upgrades as well as non-CDB to PDB conversions. In the case of a non-CDB to PDB upgrade, the config file specifies the location of the pre-created CDB. AutoUpgrade plugs in the PDBs and performs the upgrade and non-CDB to PDB conversion in one pass.

Considerations for Using AutoUpgrade

You can simultaneously upgrade from different releases in the same config file to different target releases. The AutoUpgrade console monitors the upgrade process. AutoUpgrade does intense logging of the upgrade process in addition to the typical database logging.

Use AutoUpgrade after you have downloaded binaries for the new Oracle Database release and set up new release Oracle homes. It is also recommended to apply the latest Release Update (RU) to the Oracle Home before invoking AutoUpgrade. For more details consult the Database Upgrade Quick Start Guide, latest Oracle Database Upgrade Guide in the Oracle documentation library and the My Oracle Support note (Doc ID 2485457.1.)

AutoUpgrade and Migrate a Non-CDB or PDB via Refreshable Clone

The refreshable clone feature of Oracle Multitenant can be used to upgrade a Pluggable Database (PDB) in a Container Database (CDB) and optionally migrate the PDB to a CDB on a new server. It can also be used to upgrade and migrate a non-CDB to a PDB.

This technique has multiple benefits. It allows copying from source to the target in the background to minimize downtime. The source remains intact in case a rollback is required. A refreshable clone facilitates testing the operation before implementing it.

The steps for upgrading and migrating with a refreshable clone are as follows:

1. Create an upgrade user on the source CDB or the non-CDB and grant some privileges

2. Create a database link on the target PDB to the source, a non-CDB or PDB on an older release

3. Edit the AutoUpgrade Config file to specify the 1) frequency of refreshes (updates) to the target over the database link using the source redo and 2) the duration of refresh period, at the end of which the upgrade is performed

4. After the final refresh, AutoUpgrade will upgrade the target and open it.

5. If the source was a non-CDB, AutoUpgrade will also convert the target to a PDB

Considerations for Using the Refreshable Clone feature

- The constraints associated with a PDB unplug and plug into a new CDB apply to Refreshable Clone, for instance all components in the Non-CDB or source CDB must exist in the target CDB

- Execute the AutoUpgrade analyze and fixups modes on the source before starting the clone operation

- Use the target_pdb_copy_option parameter in the AutoUpgrade config file to move the file to a new location for a seamless rollback if need be

- The AutoUpgrade deploy mode creates the refreshable clone, refreshes it for the allotted time and upgrades the clone

- Applications on the source must be stopped after the final refresh and before the upgrade begins

- Cleanup by removing the upgrade user and the database link after the operation completes

Plugging a Non-CDB Database into a CDB

The multitenant architecture is the only supported architecture for Oracle Database Release 23. If you have a non-CDB, you must convert it to a PDB as part of the upgrade process. AutoUpgrade can automate the process and perform the conversion and upgrade in one operation. Note that the source database and destination CDB must meet the following requirements:

- The hardware and operating system platforms must have the same endianness.

- The components in the PDB must be a subset of those available in the CDB.

- They must have compatible character sets and national character sets. The means that either:

- The character set in the CDB is AL32UTF8, or.

- The character set in the non-CDB is the same as the character set in the CDB, or

- The character set in the non-CDB is a binary subset of the character set in the CDB.

The procedure for creating a PDB from a non-CDB is described in the Oracle Database Administrator's Guide. Oracle recommends using the AL32UTF8 character set for CDBs.

Upgrading a CDB with Multiple PDBs

If you are upgrading a container database, you can take advantage of the flexibility of the CDB architecture to approach the upgrade in one of two ways:

- Upgrade the CDB and all PDBs with a single command

- Upgrade one PDB, or a subset of the PDBs, at a time, known as an "unplug-plug upgrade"

- Create a new CDB using the new version of Oracle Database software

- Unplug one or more PDBs from the source CDB and plug them into the new CDB

- Upgrade the older PDB(s) using either DBUA or the command-line upgrade

There are tradeoffs with each technique. Some considerations for choosing an upgrade method for a CDB are as follows.

Table 3. Choosing an Upgrade Method for a CDB

"All At Once" Upgrade	Unplug-Plug Upgrade
Less effort: upgrade up to 252 PDBs with a single command	More effort: must unplug, plug and upgrade individual PDB or set of PDBs
Less flexible: requires common downtime for all application PDBs	More flexible: allows you to plan migration windows based on user needs
Upgrade automation reduces overall downtime for the upgrade process	Easier migration to a new server
Downtime for individual PDBs will be longer	Downtime for individual PDB will be shorter
Retain original database SID for the CDB	Uses a second CDB, which may require resource allocation if on the same server

The choice of an upgrade method for a CDB thus depends on your use case and upgrade or migration scenario, and methods can be combined. For example, a single PDB could be unplugged, plugged into a new CDB and upgraded while the rest of the PDBs are kept at the source version. Then the original CDB could be upgraded all at once, and the first (upgraded) PDB moved back to the original CDB using unplug and plug.

Method 2: (Full) Transportable Export and Import

The transportable tablespaces method allows you to copy a set of tablespaces from one database to another. This can be much faster than exporting and importing data from those tablespaces because the tablespaces are copied as physical files without the need to interpret the logical entities, such as rows or indexes, contained within those files. In addition to copying the tablespaces, metadata describing the objects within the source database must be moved to the new database by Data Pump export/import.

Transported tablespaces can be copied to another database that may be on a different operating system platform or running a different version of Oracle Database software. This makes transportable tablespaces a relatively fast way to migrate and upgrade a database in a single operation. The tradeoff for this speed is that transportable tablespaces can be complicated because the user is responsible for moving metadata such as procedures, packages, constraints, and so on.

Full Transportable Export and Import

Starting with Oracle Database 19c, the full transportable export/import feature combines the speed of transportable tablespaces with a much easier process for transporting metadata and data from installed database options. For additional details see the Oracle Support note, *Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup (Doc ID 2471245.1*.) If the source and destination are running Oracle Database 19.18 or higher, there is a new version of this procedure: *M5 Cross Endian Platform Migration using Full Transportable Data Pump Export/Import and RMAN Incremental Backups (Doc ID 2999157.1)*.

Full transportable export/import is a feature of Oracle Database that makes it easy to move an entire database using the transportable tablespaces feature. It automates the process of moving metadata and can move data that resides in non-transportable tablespaces such as SYSAUX and SYSTEM. In addition, full transportable export/import can transport encrypted tablespaces.

Full transportable export/import moves metadata using either dump files or over a database link. This combination of speed and simplicity makes full transportable export/import a good choice for many migration scenarios. The detailed steps for using full transportable export/import are described in the Oracle Technical Brief, "Oracle Database: Full Transportable Export/Import."

Migrating to a Pluggable Database Using Full Transportable Export/Import

Because full transportable export/import allows you to migrate an entire database to both a new operating system platform and a new release of Oracle Database in a single operation, it is a very useful method for migrating to a PDB. The procedure for using full transportable export/import to migrate to a PDB is the same as for migrating into a non-CDB.

Examples of Using Full Transportable Export/Import

The complete implementation of the full transportable export/import feature is included in Oracle Database 12c and later releases, including Oracle Database 19c. In addition, export-side support for this feature is available starting in Oracle Database 11g Release 2 (11.2.0.3).

In these examples, assume that you start with an 11.2.0.3 source database containing the following user tablespaces:

Table 4. Source Database Tablespaces

TABLESPACE NAME	ENCRYPTED?	Datafile Name
HR	Yes	/data3/oracle/dbs/hr_1.f
ENGTABLES	No	/data4/oracle/dbs/eng_1.f
ENGINDEXES	**No**	**/data4/oracle/dbs/eng_2.f**

All three of these tablespaces, as well as all of the metadata such as procedures, packages, constraints, triggers, and so on, will be moved from the source to the destination in a single operation.

The first example is a dumpfile-based full transportable export/import operation. In this case the metadata from the source database is exported to a dump file, and both the dump file and the tablespace data files are transferred to a new system. The steps would be as follows:

6. Set user tablespaces in the source database to READ ONLY.

7. From the Oracle Database 11g Release 2 (11.2.0.3) environment, export the metadata and any data residing in administrative tablespaces from the source database using the FULL=Y and TRANSPORTABLE=ALWAYS parameters. Note that the VERSION=12 parameter is required only when exporting from an Oracle Database 11g Release 2 database:

```
expdp src112admin/<passwd>@src112 DIRECTORY=src112_dir

        DUMPFILE=src112fulltts.dmp VERSION=12 FULL=Y

        TRANSPORTABLE=ALWAYS EXCLUDE=TABLE_STATISTICS,INDEX_STATISTICS

        ENCRYPTION_PASSWORD=<enc_passwd> METRICS=Y

        LOGFILE=src112fullttsexp.log
```

8. Copy the tablespace data files from the source system to the destination system. Note that the log file from the export operation will list the data files required to be moved.

9. Create a CDB on the destination system, including a PDB into which you will import the source database. Administration of PDBs is described in Oracle Database Administrator's Guide.

10. In the Oracle Database Release 23 environment, connect to the pre-created PDB and import the dump file. The act of importing the dump file will plug the tablespace data files into the destination PDB:

```
impdp pdbadmin/<passwd>@tgtpdb  DIRECTORY=src112_dir

        DUMPFILE=src112fulltts.dmp

        ENCRYPTION_PASSWORD=<enc_passwd>

        METRICS=Y  LOGTIME=ALL LOGFILE=src112fullttsimp.log

        TRANSPORT_DATAFILES='/recovery1/data/hr_1.f'

        TRANSPORT_DATAFILES='/recovery1/data/eng_1.f'

        TRANSPORT_DATAFILES='/recovery1/data/eng_2.f'
```

ORACLE

The second example of using full transportable export/import employs the network mode of Data Pump to eliminate the need for a dumpfile. In this case, we will assume that the tablespace data files are in a location, such as a Storage Area Network (SAN) device, which is accessible to both the source and destination systems. This enables you to migrate from a non-CDB into a PDB with one Data Pump command:

1. Create a CDB on the destination system, including a PDB into which you will import the source database.

2. Create a database link in the destination PDB, pointing to the source database.

3. Set user tablespaces in the source database to READ ONLY.

4. In the Oracle Database Release 23 environment, import directly from the source database into the destination PDB using full transportable export/import in network mode:

```
impdp pdbadmin/<passwd>@tgtpdb NETWORK_LINK=src112 VERSION=12

        FULL=Y TRANSPORTABLE=ALWAYS

        EXCLUDE=TABLE_STATISTICS,INDEX_STATISTICS

        ENCRYPTION_PASSWORD=<enc_passwd>

        METRICS=Y LOGTIME=ALL LOGFILE=tgtpdb_dir:src112fullimp.log

        TRANSPORT_DATAFILES='/recovery1/data/hr_1.f '

        TRANSPORT_DATAFILES='/recovery1/data/eng_1.f '

        TRANSPORT_DATAFILES='/recovery1/data/eng_2.f '
```

Whether you use conventional dump files or network mode, full transportable export/import is a convenient way to upgrade a database to a new version, migrate to a different operating system or hardware platform, migrate into a PDB – or even to perform all three of these upgrades and migrations in a single operation!

Considerations for Using Full Transportable Export/Import

- Full transportable export/import is subject to the limitations detailed in the Oracle Database Administrator's Guide.

- Full transportable export/import can be used to migrate source databases starting with Oracle Database 11g Release 2 (11.2.0.3).

- Full transportable export/import jobs cannot be restarted. If the operation is interrupted, then the entire job must start over from the beginning.

- For complete documentation of full transportable export/import, including details about network mode migration into a PDB, see the Oracle Database Utilities Guide.

- If hardware and operating system platforms of your source and destination databases have different endian characteristics, you will need to use the RMAN CONVERT command to convert each tablespace to the new platform. See the Oracle Database Backup and Recovery Reference for a description of the RMAN CONVERT command.

Transportable Tablespaces

The transportable tablespaces feature can be used to copy a single tablespace or a set of tablespaces to a new database. While this method of migrating data is both fast and reliable, it requires a number of manual steps that can be more complicated than some DBAs would prefer.

Migrating to a Pluggable Database Using Transportable Tablespaces

Like full transportable export/import, transportable tablespaces can be used to migrate to a PDB. The procedure for using transportable tablespaces to migrate a database to a PDB is the same as a transportable tablespaces migration to a non-CDB.

Example of Using Transportable Tablespaces

Oracle recommends using full transportable export/import to migrate databases starting with Oracle Database 11g Release 2 (11.2.0.3). However, you can still use the transportable tablespaces feature to migrate databases from earlier releases. Th e use of the transportable tablespaces feature is described in detail by the following technical briefs:

"Database Upgrade Using Transportable Tablespaces"

"Platform Migration Using Transportable Tablespaces"

The steps for using this feature remain unchanged for Oracle Database 19c.

Considerations for Using Transportable Tablespaces

Transporting tablespaces between databases is subject to the limitations detailed in the Oracle Database Administrator's Guide.

Transportable tablespace export/import jobs cannot be restarted. If the operation is interrupted, then the entire job must start over from the beginning.

Transportable tablespaces can be used on databases starting with Oracle8i Database. This feature gained the ability to migrate cross-platform starting with Oracle Database 10g (10.1.0.3).

If hardware and operating system platforms of your source and destination databases have different endian characteristics, you will need to use the RMAN CONVERT command to convert each tablespace to the new platform. See the Oracle Database Backup and Recovery Reference for a description of the RMAN CONVERT command.

Method 3: Oracle Data Pump Export/Import

Oracle Data Pump provides high-speed movement of data and metadata within and between Oracle databases. Because they are extremely flexible and easy to use, the Oracle Data Pump export (expdp) and import (impdp) utilities are commonly used to migrate tables, schemas, and databases to new hardware servers, to different operating system platforms, and to new releases of Oracle Database software.

Oracle Data Pump can write data to dump files on disk, or it can transfer data from the source database to the destination directly over the network. When the data is imported, it can be transformed to match the characteristics of the destination database. Some interesting ways in which a database can be transformed upon import include migrating to a new character set, implementing encryption or compression, changing BasicFiles LOBs to SecureFiles LOBs, or changing the partitioning of tables in the database.

Data Pump Export and Import with Dump Files

Exporting data to a dump file has been the most common way to move data between databases for many years. Data Pump export chooses the best available method to extract data from the source database into flat files, and Data Pump import makes a similar decision to read from dump files and insert data into the destination database.

The biggest advantage of exporting to dump files is that you retain a persistent copy of the data on disk. This allows you to reuse the dump files for multiple imports, which can be particularly helpful when you need to test or tune your import or your source database. Another case in which exporting to dump files can be helpful is when the network between the source and destinations is relatively slow. In such cases it may be faster to physically disconnect a disk from the source system and connect it to the destination, compared to copying data over the network.

Oracle Data Pump Network Mode

Instead of exporting the source database to dump files and then importing from those same dump files into the destination database, you may choose to migrate your database using Data Pump import over a network link. This eliminates the need to store, manage, and transfer dump files. Instead, data is extracted from the source database and inserted directly into the destination database over a database link.

In addition to the reduced need for file storage and management, network mode simplifies your migration by letting you migrate cross-platform and to a new release of Oracle in a single step. Data Pump network mode jobs are fully restartable.

Migrating to a Pluggable Database Using Oracle Data Pump

Oracle Data Pump export/import is an easy and flexible way to migrate a database into a PDB. This is a particularly useful method to use when your source database resides on a different operating system, uses a different character set, or otherwise needs to be modified or reconfigured during the migration process.

Example of Using Oracle Data Pump

The steps for using Oracle Data Pump to upgrade or migrate a database have not changed from previous releases. If you would like to migrate with Oracle Data Pump using conventional dump files, the steps would be as follows:

1. Export the source database to a dump file:

    ```
    expdp srcadmin/<passwd> DIRECTORY=src_dir

            DUMPFILE=srcfull.dmp LOGFILE=srcfullexp.log

            FULL=Y METRICS=Y LOGTIME=ALL PARALLEL=4

            EXCLUDE=TABLE_STATISTICS,INDEX_STATISTICS
    ```

2. Make the dump file available on the destination system, either by copying it to the destination system or placing it on a network-mounted disk.

3. Create a CDB on the destination system, including a PDB into which you will import the source database.

4. In the destination PDB, create a directory object for the dump file to be imported.

5. Import the Data Pump dump file into the destination PDB:

    ```
    impdp pdbadmin/<passwd>@pdb DIRECTORY=src_dir

            DUMPFILE=srcfull.dmp LOGFILE=srcfullimp.log

            PARALLEL=4 METRICS=Y LOGTIME=ALL
    ```

In this example, we take advantage of Oracle Data Pump features such as the PARALLEL parameter to improve performance. In other cases, you might use network mode to eliminate the need for a dump file. Oracle Data Pump is fully documented in the Oracle Database Utilities Guide.

Note: *Beginning with Oracle Database 12c Release 2, Data Pump added the ability to export and import metadata in parallel. This greatly speeds up migration of metadata heavy databases.*

Considerations for Using Oracle Data Pump

Oracle Data Pump is available starting with Oracle Database 10g.

Oracle Data Pump does not export objects (including GRANT objects) from the SYS schema. If you have user objects in the SYS schema, such as user GRANTs on SYS-owned objects, these must be reproduced separately in the target database.

Oracle Data Pump network mode is subject to limitations on network links, as described in the Oracle Database Utilities Guide.

Starting with Oracle Database 11g Release 2 (11.2.0.3), you can set the parameter VERSION=12 when exporting to a dump file. This will specify that all data from registered database options and components should be included in the export. A dump file produced with the setting VERSION=12 can be imported starting with Oracle Database 12c.

Method 4: Original Export and Import

Oracle recommends that you use the Data Pump export and import utilities to move data between Oracle databases. However, the original Export (exp) and Import (imp) may be useful when upgrading or migrating older databases to Oracle Database 12c. For example, you may want to migrate from Oracle9i Database to Oracle Database 12c on a different operating system platform. Because Oracle Data Pump is available starting with Oracle Database 10g, you would use original Export for this operation.

CONCLUSION

This technical brief provides an overview of the different tools, techniques, and utilities provided by Oracle to help you upgrade and migrate, as well as guidance and information to help you choose among the various upgrade and migration methods available. The best method to use for your migration scenario depends on the source database version, the source and destination operating systems, your downtime requirements, and the personal preference of the DBA. Based on these factors, there is a method available that is the best fit for your migration scenario.

Upgrading or migrating to Oracle Database Release 23 can bring many benefits to your organization. You can take advantage of new features to improve performance, enhance security, and expand functionality. You can modernize your IT infrastructure. You can also improve operational efficiency to reduce costs and raise productivity.

If you would like to achieve the minimum possible downtime for your database upgrade or migration, you can start by choosing the fastest upgrade or migration method based on your source and destination platform and Oracle Database release. You can combine your chosen upgrade or migration method with minimal downtime features and products to maximize system availability.

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

🅱 blogs.oracle.com   f facebook.com/oracle   🐦 twitter.com/oracle