

Oracle Database Utilities: Full Transportable Export/Import

ORACLE WHITE PAPER | FEBRUARY 2018





Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.



ORACLE®



Disclaimer	1
Introduction	1
Benefits of Using Full Transportable Export/Import	2
Full Transportable Export/Import Support for Pluggable Databases	3
Internals of Full Transportable Export/Import	3
Example: Using Full Transportable to Move a Non-CDB into a CDB	5
Conclusion	8
Appendix: Limitations on Full Transportable Export/Import	8



Introduction

Oracle Database includes significant features such as the Oracle Multitenant Option. This multitenant architecture includes the ability to create pluggable databases (PDBs), a way to consolidate multiple databases into a single multitenant container database (CDB). Customers can save on database management costs, achieve more efficient hardware utilization, and realize economies of scale by migrating to a consolidated database environment using pluggable databases.

Traditional methods used for database migration work seamlessly with pluggable databases. The original Oracle Export/Import, Oracle Data Pump export/import, and transportable tablespaces can all be used to migrate from earlier versions of Oracle Database to a pluggable Oracle Database. In addition, the feature, full transportable export/import, is available to make migration to Oracle Database 12c and higher faster, easier, and more efficient than ever before.

This white paper describes the full transportable export/import feature in Oracle Database. After giving an overview of the benefits of using full transportable export/import, it explains how the feature works and provides a detailed example of full transportable export/import to show the syntax and process flow of using this feature. Note that, while this white paper focuses on the use of full transportable export/import in a pluggable database environment, this feature can be used generally for migrations to a Oracle Database 12c non-CDB database as well.

Benefits of Using Full Transportable Export/Import

Full transportable export/import combines the ease of use familiar to users of original Export/Import and Data Pump export/import, with the speed of migration possible with transportable tablespaces. A recap of these three migration techniques, and a comparison and contrast with full transportable export/import, helps show why full transportable export/import is faster, easier, and more efficient than previously available migration methods.

Full transportable export is available starting with Oracle Database 11g Release 2 (11.2.0.3). Full transportable import is available starting with Oracle Database 12c.

Original Export/Import is generally the slowest method for performing database migrations. While it has been available for many years, and has been a mainstay utility for many DBAs, original Export/Import was not designed with large (100+ GB) databases in mind. Its performance does not scale to the data volume of modern IT systems. Further, original Export/Import has not been enhanced to support new database functionality added in Oracle Database 10g and later releases. Therefore, while original Export/Import is a known and reliable migration technique, it should be used only for Oracle 9i and earlier databases.

Oracle Data Pump was introduced in Oracle Database 10g, and is designed to handle large volumes of data. Employing techniques such as parallel worker processes, choosing the best access method for the data being moved, and offering flexible and powerful exclude/include capabilities, Oracle Data Pump has effectively replaced original Export/Import as the most common way to move data between Oracle databases. The command line for Oracle Data Pump, while not identical to that of original Export/Import, offers a familiar feel to DBAs and is generally considered very easy to use. Even with all these benefits, however, there can be cases where Data Pump is overtaken in terms of performance by transportable tablespaces. When the data being moved is very large (more than a few hundred gigabytes), or when there are large volumes of indexes to be moved, Data Pump is fast...but other techniques may be even faster.

Transportable tablespaces are usually the fastest way to move user and application data between databases, because tablespace data files are moved en masse from the source database to the target. Moving an entire data file is generally much faster than exporting and importing individual rows or even blocks of data. However, traditional transportable tablespaces can require a fairly complicated set of steps to move user and application metadata needed to effectively use these tablespace data files in the destination database. A migration using transportable tablespaces can therefore be characterized as being very fast but more complex.

As of Oracle Database 12c, the best features of speed and usability are combined in full transportable export/import. The command line for full transportable export/import is that of Oracle Data Pump. Full transportable export/import can take advantage of Data Pump options such as the ability to move metadata over a database link, and is able to accomplish a full database migration with a single import command.

At the same time, full transportable export/import uses the transportable tablespaces mechanism to move user and application data. This results in a migration that is very fast, even for very large volumes of data. Most important, full transportable export/import moves all of the system, user, and application metadata needed for a database migration, without the complex set of steps required for a traditional transportable tablespaces operation.

Thus, full transportable export/import combines the ease of use of Oracle Data Pump with the performance of transportable tablespaces, resulting in a feature that makes database migration faster and easier.

Full Transportable Export/Import Support for Pluggable Databases

Full transportable export/import was designed with pluggable databases as a migration destination. You can use full transportable export/import to migrate from a non-CDB database into a PDB, from one PDB to another PDB, or from a PDB to a non-CDB. Pluggable databases act exactly like non-CDBs when importing and exporting both data and metadata.

The steps for migrating from a non-CDB into a pluggable database are as follows:

- » Create a new PDB in the destination CDB using the `create pluggable database` command
- » Set the user and application tablespaces in the source database to be `READ ONLY`
- » Copy the tablespace data files to the destination
- » Using an account that has the `DATAPUMP_IMP_FULL_DATABASE` privilege, either
 - » Export from the source database using `expdp` with the `FULL=Y TRANSPORTABLE=ALWAYS` options, and import into the target database using `impdp`, or
 - » Import over a database link from the source to the target using `impdp`
- » Perform post-migration validation or testing according your normal practice

Internals of Full Transportable Export/Import

While not intended to be an in depth technological treatise, this section describes the mechanisms behind full transportable export/import. The intent of this section is to give you a better understanding of what full transportable export/import does, and how it achieves the optimal combination of usability and performance.


Full Transportable Export/Import Concepts

To understand the internals of full transportable export/import, you need to know the difference between moving data in a conventional manner versus a transportable approach, and the distinction between administrative and user tablespaces.

When using conventional methods to move data, Oracle Data Pump uses either external tables or direct path unload to extract data. While the choice between these two access methods is based on the structure and types of the data being unloaded, both methods efficiently extract logical subsets of data from an Oracle database.

In contrast, a transportable move of data and indexes involves the physical movement of one or more tablespace data files. The data segments inside the tablespace data files are not read individually. Instead, the export operation extracts the metadata that describes the objects containing storage within each data file, and each file is moved as a single entity. Moving large volumes of data using transportable tablespaces can be faster than conventional data movement because there is no need to interpret and extract individual rows of data or index entries. It is possible to move individual tables or partitions in a transportable manner, but the entire tablespace data file is moved in these cases as well.

Understanding the difference between conventional and transportable data movement is helpful when considering the distinction between administrative and user tablespaces. For the purposes of a full transportable export, administrative tablespaces are the tablespaces provided by Oracle, such as `SYSTEM`, `SYS_AUX`, `TEMP`, and `UNDO`. These tablespaces contain the procedures, packages, and seed data for the core Oracle database functionality and Oracle-provided database components such as Oracle Spatial, Oracle Text, OLAP, JAVAVM, and XML Database. In contrast, user tablespaces are those tablespaces defined by database users or applications. These may store user data, application data, and any other information defined by users of the database.



The first step of a full transportable export is to create the destination database. This newly created database includes a set of administrative tablespaces appropriate to the target environment, complete with Oracle-supplied components and packages. Starting with Oracle Database 12c, Oracle-supplied objects are neither exported nor imported by Oracle Data Pump. Thus, when migrating data from the administrative tablespaces of the source database, full transportable export uses conventional data movement to extract only the data and metadata for user-defined objects that will be need to be added to the destination database.

The user tablespaces, on the other hand, are moved to the destination database as full tablespace data files, under the assumption that a full export migrates all user and application data and metadata from the source to the destination system. User tablespaces are thus moved transportably, resulting in maximum performance when migrating user data.

One consideration specific to full transportable export/import arises when there exists a database object (e.g., a partitioned table) that is stored across both user and administrative tablespaces. Storing an object in this way is generally not good practice, but it is possible. If there is an object with storage in both administrative and user tablespaces, then you can either redefine that object before transporting your data, or use conventional Data Pump export/import. The example later in this white paper shows how to detect this condition prior to starting a full transportable export.

Full Transportable Export

The key to the improved usability of full transportable export/import is that the export is able to extract the metadata for all user and administrative objects. A callout mechanism and API is provided for internal use by Oracle components, allowing for a complete extract of the metadata needed to create a full copy of the database upon import.

All user and application objects that reside in administrative tablespaces are unloaded conventionally, including both their metadata (e.g. a table definition) and their data (e.g. the rows in that table). In contrast, objects stored in user tablespaces have only their metadata unloaded by Data Pump; the data for those objects is moved transportably in the tablespace data files.

Invoking Full Transportable Export

Oracle Data Pump export (expdp) is the command line interface for full transportable export. You can initiate a full transportable export by specifying two parameters in the parameter file or on the command line:

`TRANSPORTABLE=ALWAYS` and `FULL=Y`. These parameter values tell Data Pump to use full transportable rather than conventional export methods.

In addition, there is a special consideration if the `COMPATIBLE` database initialization parameter of the source database is not set to a value of at least 12.0. This would be true, for example, if you perform a full transportable export from Oracle Database 11g Release 2 (11.2.0.3). In this case, you must also specify the Data Pump parameter `VERSION=12` to denote the fact that the result of the export will be imported into an Oracle Database 12c Release 1 (12.1) or later database.

Full transportable Import

Like a conventional Data Pump import, full transportable import can be used to import a dump file or to import directly from a source database into a destination database over a database link. The ability to import without using a dump file makes full transportable import an indispensable tool for database migrations.

Invoking Full Transportable Import

If you are importing a dump file, Oracle Data Pump is able to determine whether a dump file was produced by a conventional or full transportable export. A file-based full transportable import thus requires only that you specify the dumpfile name and the list of user tablespace data files to be transported using the `TRANSPORT_DATAFILES=<datafile_names>` parameter. This parameter can use a comma-separated list of files, or can be specified multiple times for multiple data files. Oracle recommends the use of a parameter file to avoid issues such as proper use of quotation marks around filenames.

If you are migrating directly from the source database to the target database without using a dump file, then you must specify several parameters to start the operation in full transportable mode. This is because a network mode import actually performs the export from the source database and import to the target database as a single operation. A network-based full transportable import requires the following parameters:

- » `FULL=Y TRANSPORTABLE=ALWAYS` as specified for a full transportable export
- » `TRANSPORT_DATAFILES=<datafile_names>` as used in a file-based full transportable import
- » `VERSION=12` if the `COMPATIBLE` setting for the source database is lower than 12.0. Note that the source database must be Oracle Database 11g Release 2 (11.2.0.3) or higher for a full transportable import
- » `NETWORK_LINK=<dblink_name>` to specify the database link over which the data and metadata are transferred from the source database

Note that user tablespaces can be in a read/write state on the destination system during a full transportable import. It is important that no other user applications execute during the import job. At the successful completion of the import job, all user tablespaces will be set to a read/write state.

Endian Conversion

If both the source and target platforms have the same endian format, then data files can be transported as if they were on the same platform. If the source platform and the target platform are of different endianness, then the data files being transported must be converted to the target platform format using either the `RMAN CONVERT` command or the `GET_FILE` or `PUT_FILE` procedures in the `DBMS_FILE_TRANSFER` package.


Note that encrypted tablespaces cannot be transported to a platform with different endianness.

Example: Using Full Transportable to Move a Non-CDB into a CDB

In this example, we use full transportable to migrate an Oracle Database 11g Release 2 (11.2.0.3) database from an Oracle Solaris x86 server to an Oracle Database 12c PDB inside of a CDB running on Oracle Linux. This example uses a dump file, requiring a full transportable export from the source database followed by a full transportable import at the destination database. The source database has two user tablespaces: `hr_1` and `hr_2`.

TABLE 1. SOURCE DATABASE TABLESPACES

Tablespace Name	Encrypted?	Datafile Name
HR_1	Yes	/u01/app/oracle/oradata/hr_db/hr_101.dbf
HR_2	No	/u01/app/oracle/oradata/hr_db/hr_201.dbf



For this example we assume that the Oracle SID of the target database is HR_PDB, and that the data files for the target PDB are stored in the directory /u01/app/oracle/oradata/hr_pdb/ on the destination server.

Note that one of the tablespaces in this example is encrypted. The commands used to set up and administer Transparent Data Encryption (TDE) have changed between Oracle Database 11g Release 2 and Oracle Database 12c. Please see *Oracle Database Advanced Security Guide* for more information about implementing TDE for your database.

STEP 1: Check the endianness of both platforms

To check the endianness of a platform, run the following query on each platform.

```
SQL> SELECT d.PLATFORM_NAME, ENDIAN_FORMAT
       FROM V$TRANSPORTABLE_PLATFORM tp, V$DATABASE d
       WHERE tp.PLATFORM_ID = d.PLATFORM_ID;
```

In this case, both Oracle Solaris x86 and Oracle Enterprise Linux have little endian format, so no endian conversion is necessary.

STEP 2: Verify that the set of tablespaces to be transported is self-contained

Before transporting a set of tablespaces, you must verify that there are no logical or physical dependencies between objects stored in the tablespaces being transported and those that are not being transported. For example, you need to detect situations in which an object is stored in both user and administrative tablespaces. This is referred to as a containment check.

To determine whether our tablespaces hr_1 and hr_2 are self-contained, including verification that referential integrity constraints will be valid after the transport, execute the following command on the source database.

```
SQL> EXECUTE DBMS_TTS.TRANSPORT_SET_CHECK('hr_1,hr_2', TRUE);
```

Note that you must include all user tablespaces in the database when performing this check for a full transportable export/import.

After invoking this PL/SQL procedure, you can see all violations by selecting from the TRANSPORT_SET_VIOLATIONS view.

```
SQL> SELECT * FROM TRANSPORT_SET_VIOLATIONS;
```

If the set of tablespaces is self-contained, this view will be empty. If any violations are listed, then you must address these issues before proceeding with the full transportable operation.

STEP 3: Create a directory object in the source database

Before you start the full transportable export, create a directory object that points to the location to which the dump file will be written. Read and write access to this directory is automatically granted to the DBA role and to users SYS and SYSTEM.

```
SQL> CREATE DIRECTORY dp_dir AS '/u01/app/datafiles';
```

This directory object can be dropped from the database after the full transportable export has finished.

STEP 4: Place the hr_1 and hr_2 tablespaces in read-only mode

The tablespaces to be transported must be in read-only mode for the duration of the export. In this case we need to issue two commands on the source database.

```
SQL> ALTER TABLESPACE hr_1 READ ONLY;
```

```
SQL> ALTER TABLESPACE hr_2 READ ONLY;
```

The tablespaces can be returned to read-write status once the full transportable export has finished and the tablespace data files have been copied to the destination system..

STEP 5: Invoke full transportable export on the source database

The use of METRICS=Y and EXCLUDE of statistics are not required, but are generally good practice for data pump exports.

Invoke the Data Pump export utility as a user with the DATAPUMP_EXP_FULL_DATABASE role.

```
$ expdp system/manager full=y transportable=always version=12 \
    directory=dp_dir dumpfile=full_tts.dmp \
    metrics=y exclude=statistics \
    encryption_password=secret123word456 \
    logfile=full_tts_export.log
```

Note that the VERSION=12 parameter is required because the source database is Oracle Database 11g Release 2 (11.2.0.3). This is the only time that a version number greater than the current version is allowed by the expdp command. If the source database is Oracle Database 12c, with COMPATIBLE=12.0 or higher, then the VERSION parameter is not required.

After the export command completes, the export log file shows a list of all of the tablespace data files that need to be moved to the target.

STEP 6: Transport the tablespace data files and the export dump file from source to target


Because we are migrating this database to a new server, we must copy the tablespace data files and the export dump file to a location that is accessible from the destination database. For example, on the destination system you could issue the following commands.

```
$ cd /u01/app/oracle/oradata/hr_pdb/
$ cp /net/<source-server>/u01/app/oracle/oradata/hr_db/hr_101.dbf .
$ cp /net/<source-server>/u01/app/oracle/oradata/hr_db/hr_201.dbf .
$ cp /net/<source-server>/u01/app/datafiles/full_tts.dmp .
```

STEP 7: Create a directory object on the destination database

Because we copied the data pump dump file to the oradata directory for HR_PDB, we will create a directory object to point to that same directory for this import. This directory object must be created by a user connected to the PDB container.

```
SQL> CREATE DIRECTORY dp_dir AS '/u01/app/oracle/oradata/hr_pdb';
```



```
SQL> GRANT read, write on directory dp_dir to system;
```

This directory object can be dropped from the database after the full transportable import has finished.

STEP 8: Invoke full transportable import on the destination database

Invoke the Data Pump import utility as a user with `DATAPUMP_IMP_FULL_DATABASE` role.

You must explicitly specify the service name of the PDB in the connect string for the `impdp` command.

```
$ impdp system/manager@hr_pdb directory=dp_dir \  
  dumpfile=full_tts.dmp logfile=full_tts_imp.log \  
  metrics=y \  
  encryption_password=secret123word456 \  
  transport_datafiles='/u01/app/oracle/oradata/hr_pdb/hr_101.dbf',\  
                    '/u01/app/oracle/oradata/hr_pdb/hr_201.dbf'
```

Note that, while this example shows several parameters specified on the command line, in most cases use of a data pump parameter file is recommended to avoid problems with white space and quotation marks on the command line.

After this statement executes successfully, the user tablespaces are automatically placed in read/write mode on the destination database. Check the import log file to ensure that no unexpected error occurred, and perform your normal post-migration validation and testing.

STEP 9: (Optional) Restore user tablespaces to read-write mode on the source database

After the full transportable export has finished, you can return the user-defined tablespaces to read-write mode at the source database if desired.

```
SQL> ALTER TABLESPACE hr_101 READ WRITE;  
SQL> ALTER TABLESPACE hr_201 READ WRITE;
```


On the destination database, all tablespaces are set to read-write mode automatically upon completion of the full transportable import.

Conclusion

Full transportable export/import is a feature beginning with Oracle Database 12c that greatly simplifies the process of database migration. Combining the ease of use of Oracle Data Pump with the performance of transportable tablespaces, full transportable export/import gives you the ability to upgrade or migrate to Oracle Database 12c or later in a single operation if your source database is at least Oracle Database 11g Release 2 (11.2.0.3). Full transportable export/import is a valuable tool for migrating to pluggable databases, allowing you to take advantage of the cost savings and economies of scale inherent in moving to a multitenant architecture.

Appendix: Limitations on Full Transportable Export/Import

While full transportable export/import greatly enhances the usability of traditional transportable tablespaces, there are limitations of transporting data about which you should be aware.

- 
- » Like traditional transportable tablespaces, full transportable export/import requires that the user tablespaces to be transported be in read-only mode for the duration of the export. If this is undesirable (which may be particularly true when you are testing this method before using it in production), then you can use the RMAN capability to transport tablespaces from backups. See *Oracle Database Backup and Recovery User's Guide* for more information on transporting tablespaces from backups.
 - » All objects with storage that are selected for export must have all of their storage segments entirely within administrative, non-transportable tablespaces or entirely within user-defined transportable tablespaces. Storage for a single object cannot straddle the two kinds of tablespaces. Otherwise, transportable mode is inhibited and causes the export to fail. In this case, the export needs to be done conventionally or the object's storage redefined to meet this criteria.
 - » If the source and target platforms are of different endianness, then you must convert the data being transported so that it is in the format of the target platform.
 - » You can use the RMAN CONVERT DATAFILE command to convert the data.
 - » Alternatively data files can be converted to the target platform by transferring them with the GET_FILE or PUT_FILE procedures from the DBMS_FILE_TRANSFER package
 - » You cannot transport an encrypted tablespace to a platform with different endianness.
 - » To transport an encrypted tablespace to a platform with the same endianness, use the `expdp ENCRYPTION_PASSWORD` parameter during export. During import, specify the same value for the `impdp ENCRYPTION_PASSWORD` parameter.
 - » The XDB repository is not supported by full transportable export/import. However, user-defined XML schemas are supported by all forms of export/import: Data Pump export/import, transportable tablespaces, and full transportable export/import.
 - » The Automatic Workload Repository (AWR) is not supported by full transportable export/import. Use the `awrextr.sql` and `awrload.sql` scripts to move AWR data between Oracle databases.
 - » The source and target databases must use compatible database character sets. See *Oracle Database Administrator's Guide* for more information about this and other General Limitations on Transporting Data.







Oracle Corporation, World Headquarters

500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries

Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

-  blogs.oracle.com/oracle
-  facebook.com/oracle
-  twitter.com/oracle
-  oracle.com

Integrated Cloud Applications & Platform Services

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0116

Oracle Database Utilities: Full Transportable Export/Import
February 2018
Authors: Roy F. Swonger, George Claborn, Jim Stenois, William Beauregard