19

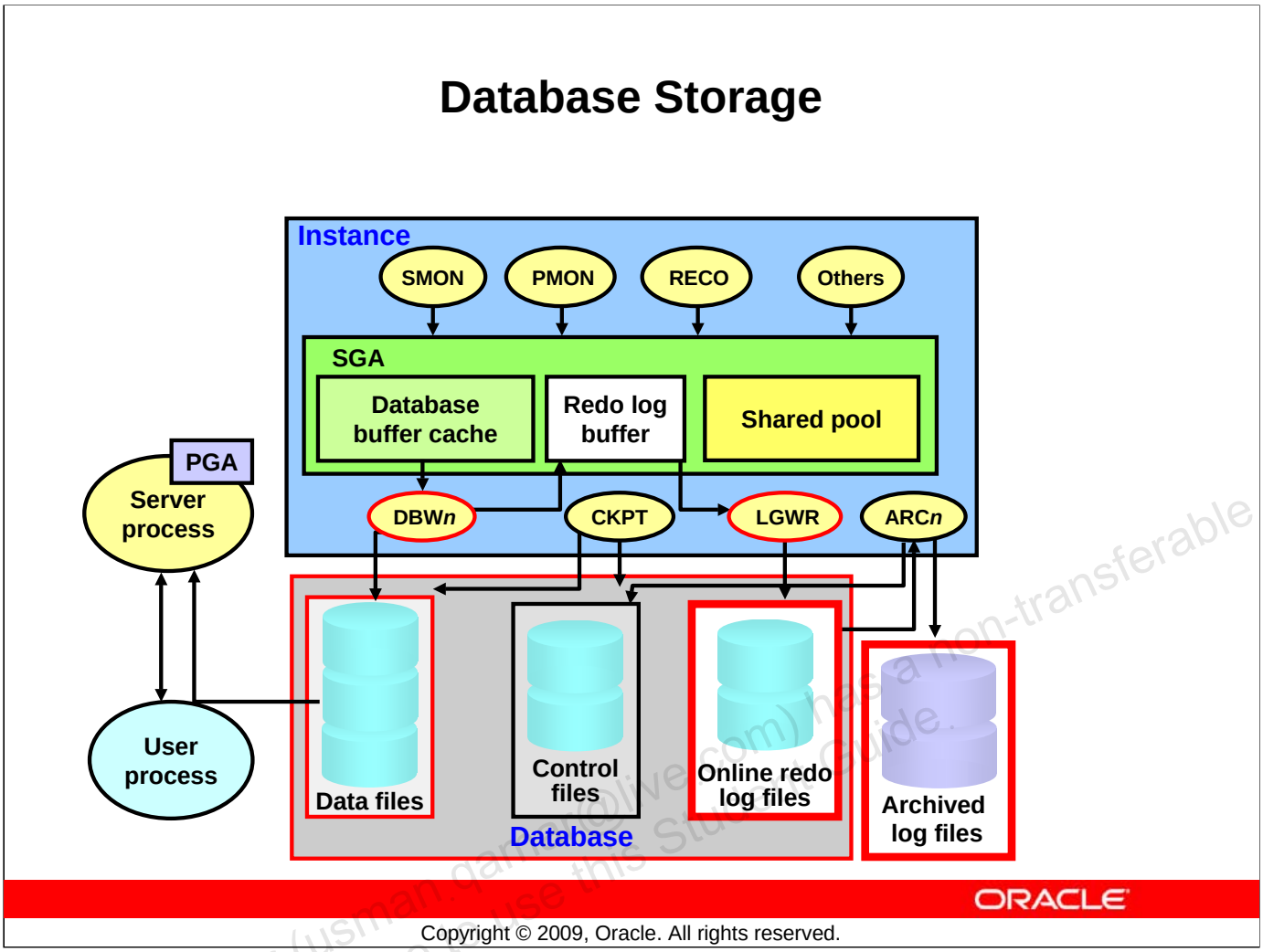
Managing Space for the Database

ORACLE

Objectives

After completing this lesson, you should be able to:

- Describe the concepts and use of 4 KB-sector disks
- Use transportable tablespaces
- Describe the concepts of transportable databases



Database Storage

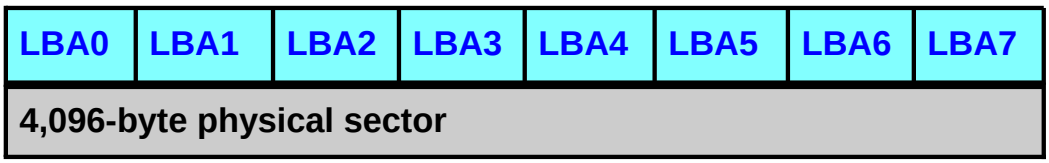
The database consists of both physical structures and logical structures. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting access to logical storage structures.

Disks, that are a primary storage medium for database, currently have predominantly a sector size of 512 bytes, but the larger, 4 KB-sector disks are beginning to appear on the market, which offer higher storage capacity with a lower overhead. Oracle databases access the hard disk via a platform-specific device driver. (The database writer and log writer [and ASM processes] can write directly to disk without going through the OS.)

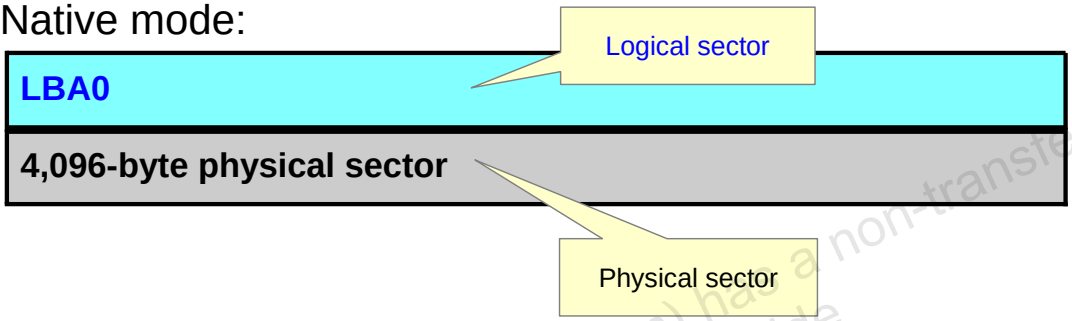
Oracle Database 11*g* Release2 detects the disk sector size and uses high-capacity disks without performance degradation (because of internal optimizations that reduce, for example, potential waste of redo space, which you might expect with applications such as an email system that has many short transactions).

Supporting 4-KB Sector Disks

- Emulation mode:



- Native mode:



ORACLE

Supporting 4-KB Sector Disks

4-KB sector disks have physical sectors (shown in gray) and logical sectors (shown in blue). There are two types of 4-KB sector disks: emulation mode and native mode.

- 4-KB sector disks in emulation mode have eight logical sectors per one physical sector (as shown in the slide). They maintain a 512-byte interface to their 4-KB physical sectors—that is, the logical block address (LBA) references 512 bytes on disk.
  Performance can be decreased in emulation mode because the disk drive reads the 4 KB sector into disk cache memory, changes the 512-byte section, and then writes the entire 4 KB sector back to disk.
- 4-KB sector disks in native mode have one logical sector per physical sector (as shown in the slide). So, there is only the 4-KB interface. That is, the LBA references 4,096 bytes on disk.

Using 4-KB Sector Disks

Emulation mode:

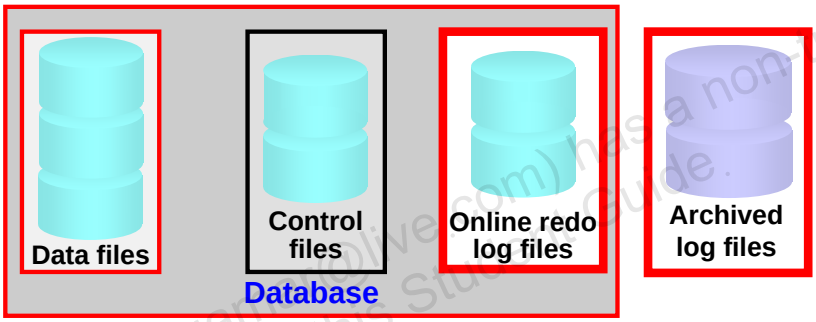
- Recommended 4-KB block size for logs
- Recommended 4-KB block size (or larger) for data files

Native mode:

- Mandatory 4-KB block size for logs
- Mandatory 4-KB block size (or larger) for data files

Not affected:

- Control file block size: 16 KB



ORACLE

Using 4-KB Sector Disks

In Oracle Database 11*g*R2, 4-KB sector disks mainly affect the redo log files. This includes online redo logs, standby redo logs, and archive logs. Oracle recommends that you create 4-KB block size logs on 4-KB emulation mode disks. On 4-KB native mode disks, you must create 4-KB block size logs.

That is, the redo block size must match the physical disk sector size (for 512-byte and for 4-KB native mode disks). Otherwise, you receive the ORA-1378 error. For 4-KB emulation mode disks, the redo block size could be 512 or 4,096 bytes. 4 KB is the preferred block size. When you are creating 512-byte blocks on a 4-KB emulation disk, a warning is printed to the alert log to indicate that the mismatched block size leads to degraded performance. This also applies to ASM disk groups.

The 4-KB sector disks also affect the Oracle data file. The Oracle database allows you the creation of 2-KB block size data files on 512-byte sector disks. With 4-KB sector disks, Oracle recommends that you create 4-KB (or larger) block size data files on the 4-KB emulation mode disks. On 4-KB native mode disks, you must create 4-KB block (or larger) size logs.

The control file block size is already 16 KB. Therefore, the 4-KB sector disks do not affect the control file.

Specifying the Disk Sector Size

Using the SECTOR_SIZE and BLOCKSIZE clauses of the following commands:

- CREATE DISKGROUP
- ALTER DATABASE
- CREATE DATABASE
- CREATE CONTROL FILE

Default sector size based on hardware (not the earlier 512-byte sectors)

```
CREATE DATABASE sample NORESETLOGS FORCE LOGGING
   ARCHIVELOG
   LOGFILE
     GROUP 1 '$ORACLE_BASE/oradata/sample/redo01.log'
             SIZE 100M BLOCKSIZE 4096,
     GROUP 2 '$ORACLE_BASE/oradata/sample/redo02.log'
             SIZE 100M BLOCKSIZE 4096
   DATAFILE
   ...
```

ORACLE

Specifying the Disk Sector Size

In an Automatic Storage Management (ASM) environment, you can set the SECTOR_SIZE attribute for disk groups. This attribute can be set only at disk group creation time (by using the CREATE DISKGROUP command).

You can specify the size of the log file with the new BLOCKSIZE clause for the following commands:

- ALTER DATABASE
- CREATE DATABASE
- CREATE CONTROL FILE

There is no additional work for you when you create a new database on 4-KB sector disks compared to creating a new database on 512-byte disks. There is no change in the GUI environments.

You have the option of using the BLOCKSIZE clause in the CREATE DATABASE command, as shown in the slide. When you do not specify a block size, the Oracle database discovers the underlying disk sector size and uses the disk sector size as the block size for the redo log creation. So by default, the redo log block size is the disk sector size, not the earlier 512-byte sector size.

Quiz

You must use 4-KB log files on 4-KB native mode disks.

1. True
2. False

Answer: 1

Quiz

Oracle recommends that you create 512-byte blocks on a 4-KB emulation disk for performance reasons.

1. True
2. False

Answer: 2

Quiz

Control files are not affected by 4-KB sector disks (because they are already larger).

1. True
2. False

Answer: 1

Transporting Tablespaces

- Cross-platform transportable tablespaces:
  - Simplify moving data between data warehouse and data marts
  - Allow database migration from one platform to another
- Supported platforms include:

Solaris[tm] OE (32-bit)	HP-UX (64-bit)	Microsoft Windows IA (64-bit)
Solaris[tm] OE (64-bit)	HP Tru64 UNIX	IBM zSeries Based Linux
Microsoft Windows IA (32-bit)	HP-UX IA (64-bit)	Linux 64-bit for AMD
Linux IA (32-bit)	Linux IA (64-bit)	Apple Mac OS
AIX-Based Systems (64-bit)	HP Open VMS	Microsoft Windows 64-bit AMD
IBM Power Based Linux	HP IA Open VMS	Solaris *x*86 and AMD64

ORACLE

Transporting Tablespaces

Transportable tablespace is the fastest way for moving large volumes of data between two Oracle databases. Using transportable tablespaces, Oracle data files (containing table data, indexes, and almost every other Oracle database object) can be directly transported from one database to another. Furthermore, like import and export, transportable tablespaces provide a mechanism for transporting metadata in addition to transporting data.

You can use the transportable tablespace feature to move data across platform boundaries. This simplifies the distribution of data from a data warehouse environment to data marts, which often run on smaller platforms. It also allows a database to be migrated from one platform to another by rebuilding the dictionary and transporting the user tablespaces.

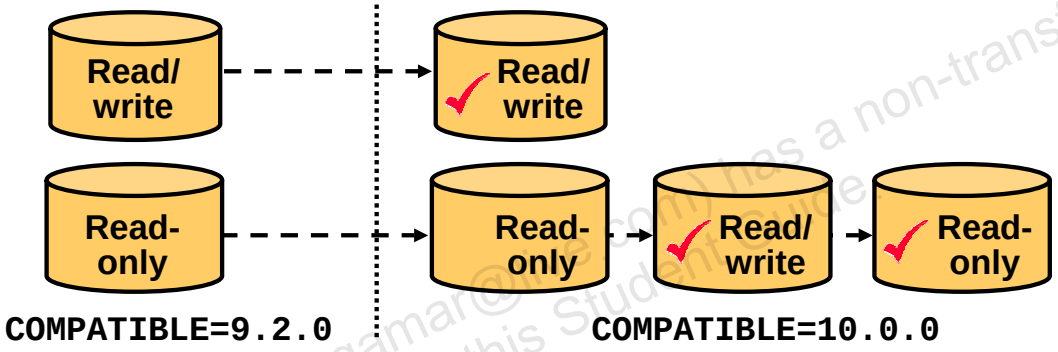
Moving data using transportable tablespaces is much faster than performing either an export/import or unload/load of the same data. This is because the data files containing all of the actual data are just copied to the destination location, and you use Data Pump to transfer only the metadata of the tablespace objects to the new database.

To be able to transport data files from one platform to another, you must ensure that both the source system and the target system are running on one of the supported platforms (see slide).

Note: The cross-platform transportable tablespace feature requires both platforms to be using the same character sets.

Concept: Minimum Compatibility Level

- Both source and target databases must have `COMPATIBLE` set to `10.0.0` or higher.
- Data file headers are platform-aware.
- Before transporting, make sure that all read-only and offline files are platform-aware.



Concept: Minimum Compatibility Level

Both source and target databases need to advance their database `COMPATIBLE` initialization parameter to `10.0.0` or greater before they can use the cross-platform transportable tablespace feature.

When data files are first opened under Oracle Database 10*g* or 11*g* with `COMPATIBLE` set to `10.0.0` (or greater), the files are made platform-aware. This is represented by the check marks in the diagram. Each file identifies the platform that it belongs to. These files have identical on-disk formats for file header blocks that are used for file identification and verification. Read-only and offline files get the compatibility advanced only after they are made read/write or are brought online. This implies that tablespaces that are read-only in databases before Oracle Database 10*g* must be made read/write at least once before they can use the cross-platform transportable feature.

Minimum Compatibility Level

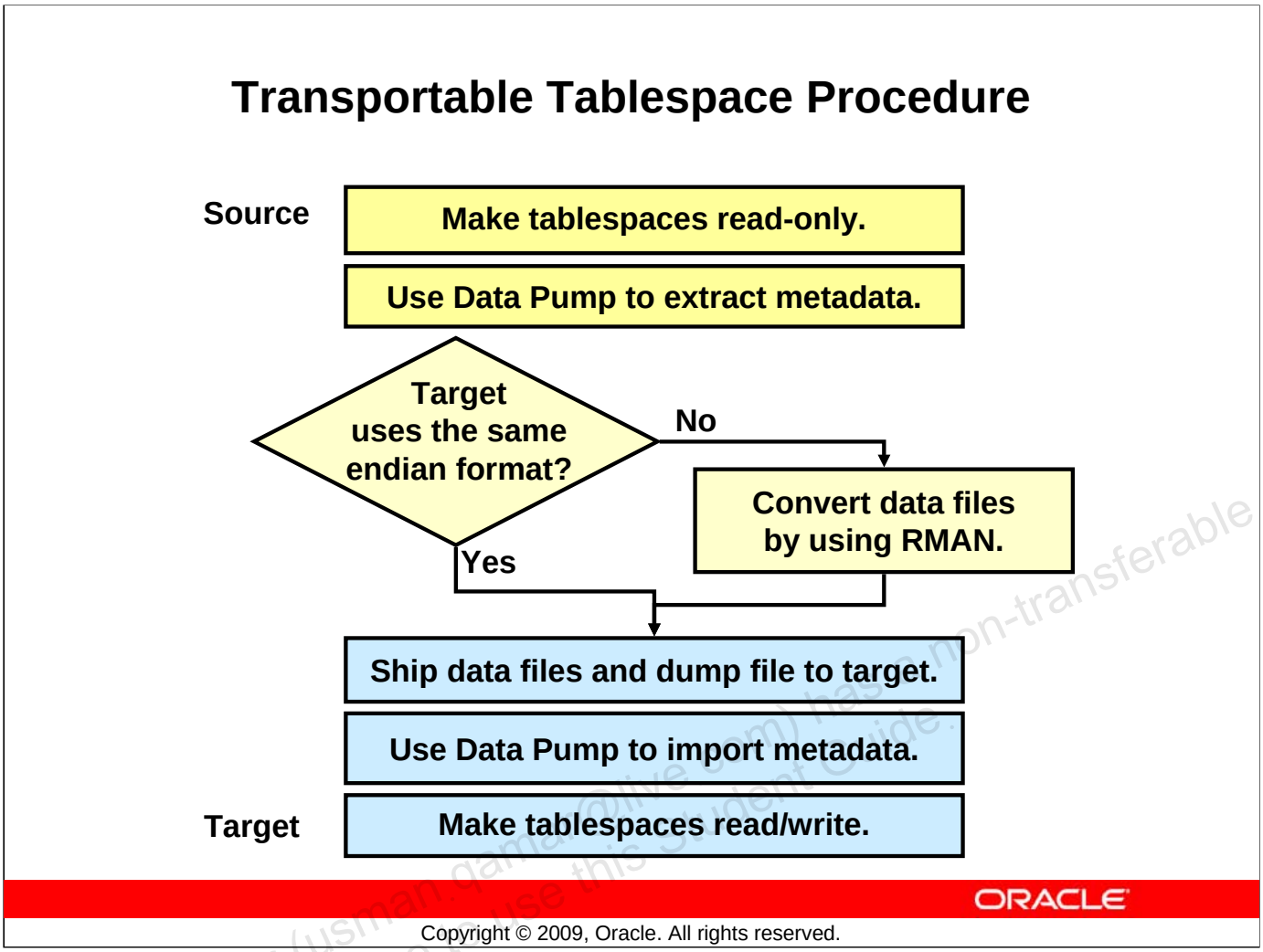
Transport Scenario	Minimum Compatibility Setting	
	Source Database	Target Database
Databases on the same platform	8.0	8.0
Tablespace with different database block size than the target database	9.0	9.0
Databases on different platforms	10.0	10.0

Minimum Compatibility Level

When you create a transportable tablespace set, Oracle Database computes the lowest compatibility level at which the target database must run. This is referred to as the compatibility level of the transportable set. Beginning with Oracle Database 11*g*, a tablespace can always be transported to a database with the same or higher compatibility setting, whether the target database is on the same or a different platform. The database signals an error if the compatibility level of the transportable set is higher than the compatibility level of the target database.

The above table shows the minimum compatibility requirements of the source and target tablespace in various scenarios. The source and target database need not have the same compatibility setting.

When data files are first opened, each file identifies the platform that it belongs to. These files have identical on-disk formats for file header blocks that are used for file identification and verification. Read-only and offline files get the compatibility advanced only after they are made read/write or are brought online.



Transportable Tablespace Procedure

To transport a tablespace from one platform to another (source to target), data files belonging to the tablespace set must be converted to a format that can be understood by the target or destination database. Although with Oracle Database, disk structures conform to a common format, it is possible for the source and target platforms to use different endian formats (byte ordering). When going to a different endian platform, you must use the `CONVERT` command of the RMAN utility to convert the byte ordering. This operation can be performed on either the source or the target platforms. For platforms that have the same endian format, no conversion is needed.

The slide graphic depicts the possible steps to transport tablespaces from a source platform to a target platform. However, it is possible to perform the conversion after shipping the files to the target platform. The last two steps must be executed on the target platform.

Basically, the procedure is the same as when using previous releases of the Oracle database server except when both platforms use different endian formats. It is assumed that both platforms are cross-transportable compliant.

Note: Byte ordering can affect the results when data is written and read. For example, the 2-byte integer value `1` is written as `0x0001` on a big-endian system (such as Sun SPARC Solaris) and as `0x0100` on a little-endian system (such as an Intel-compatible PC).

Determining the Endian Format of a Platform

```
SELECT tp.endian_format
FROM   v$transportable_platform tp, v$database d
WHERE  tp.platform_name = d.platform_name;
```

Determining the Endian Format of a Platform

You can query V$TRANSPORTABLE_PLATFORM to determine whether the endian ordering is the same on both platforms. V$DATABASE has two columns that can be used to determine your own platform name and platform identifier. Run the query below for a comprehensive list of supported platforms and their endian formats:

```
SQL> SELECT * FROM V$TRANSPORTABLE_PLATFORM;
PLATFORM_ID PLATFORM_NAME                      ENDIAN_FORMAT
----------- ---------------------------------- --------------
          1 Solaris[tm] OE (32-bit)            Big
          2 Solaris[tm] OE (64-bit)            Big
          7 Microsoft Windows IA (32-bit)      Little
         10 Linux IA (32-bit)                  Little
          6 AIX-Based Systems (64-bit)         Big
          3 HP-UX (64-bit)                     Big
          5 HP Tru64 UNIX                      Little
          4 HP-UX IA (64-bit)                  Big
         11 Linux IA (64-bit)                  Little
         15 HP Open VMS                        Little
```

Determining the Endian Format of a Platform (continued)

```
PLATFORM_ID PLATFORM_NAME                    ENDIAN_FORMAT
----------- -------------------------------- --------------
          8 Microsoft Windows IA (64-bit)    Little
          9 IBM zSeries Based Linux          Big
         13 Linux 64-bit for AMD             Little
         16 Apple Mac OS                     Big
         12 Microsoft Windows 64-bit for AMD Little
         17 Solaris Operating System (x86)   Little
         18 IBM Power Based Linux            Big
         19 HP IA Open VMS                   Little
         20 Solaris Operating System (AMD64) Little
```

Using the RMAN CONVERT Command

RMAN:

- Converts tablespaces, data files, or databases to the format of a destination platform
- Does not change input files
- Writes converted files to output destination

```
CONNECT TARGET SYS@orcl
RMAN>
SQL 'ALTER TABLESPACE hr READ ONLY';
CONVERT TABLESPACE hr
   TO PLATFORM 'Solaris[tm] OE (64-bit)'
   FORMAT '/tmp/transport_to_solaris/%U';;
```


Using the RMAN CONVERT Command

You use the RMAN CONVERT command to convert a tablespace, data file, or database to the format of a destination platform in preparation for transport across different platforms. Input files are not altered by CONVERT because the conversion is not performed in place. Instead, RMAN writes converted files to a specified output destination.

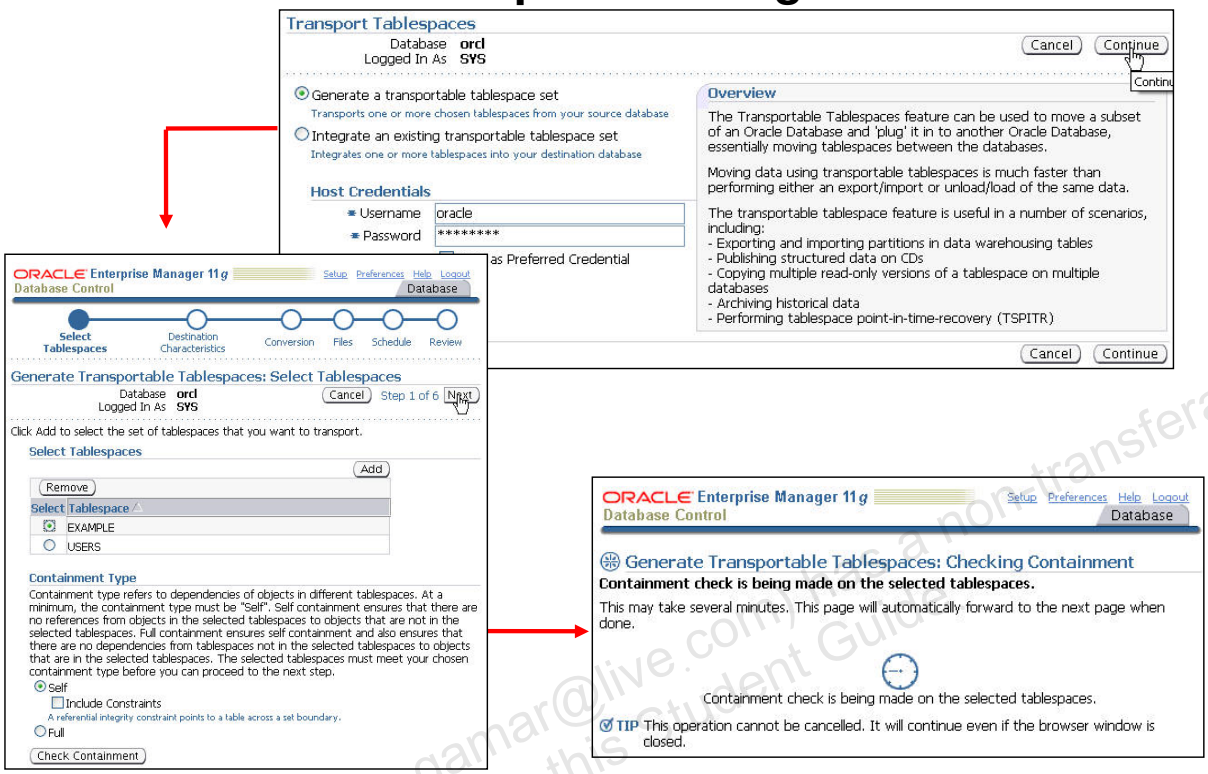
CONVERT TABLESPACE example:

- Assume that you have an ORCL database on a Linux 32-bit platform, which you want to transport to a Solaris 64-bit platform.
- Connect as TARGET to the *source* database (mounted or open).
- The tablespace must be read-only at the time of conversion.
- The result is a set of converted data files in the /tmp/transport_to_solaris/ directory, with data in the right endian-order for the Solaris 64-bit platform.

Restrictions: The CONVERT command does not process user data types that require endian conversions. To transport objects between databases that are built on underlying types that store data in a platform-specific format, use the Data Pump Import and Export utilities.

For detailed prerequisites, usage, restrictions, and syntax, see the *Oracle Database Backup and Recovery Reference*.

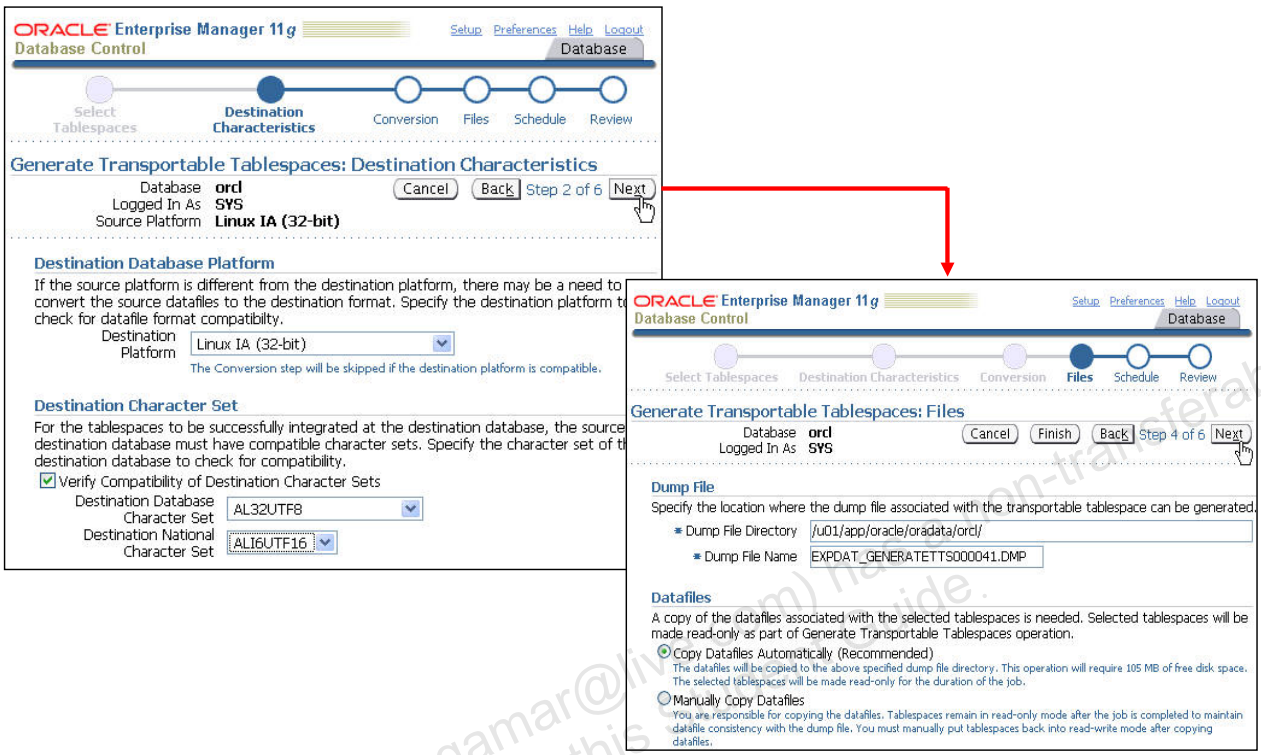
Transportable Tablespaces with Enterprise Manager



Transportable Tablespaces with Enterprise Manager

Enterprise Manager can be used to implement transportable tablespaces. From the Database home page, click the Data Movement folder tab, and then click Transport Tablespaces under the Move Database Files section. Select “Generate a transportable table set” and provide the login credentials for the `oracle` user, and then click Continue. On the Select Tablespaces page, add the tablespaces you want to transport from the displayed list by clicking the Tablespace button, and then clicking Add. Near the bottom of the page, you must select the level of containment checking to be done before the tablespaces are processed. The containment check looks for object dependencies within the tablespaces. When you have finished, click Next. Wait a few moments while the containment check runs. Address any issues found by the check before continuing.

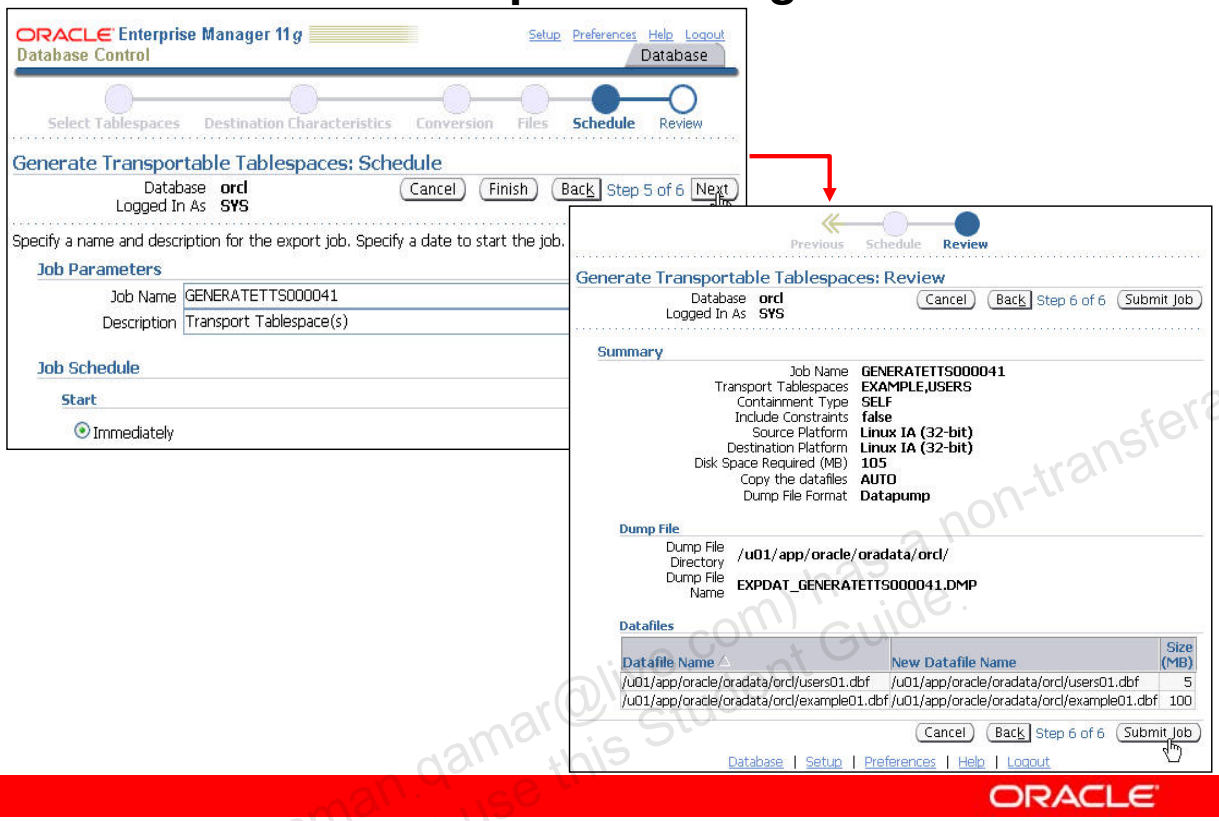
Transportable Tablespaces with Enterprise Manager



Transportable Tablespaces with Enterprise Manager (continued)

On the Destination Characteristics page, you must supply the destination platform and character sets. Under the Destination Database Platform section, select the operating system of the destination machine from the drop-down list. If the destination platform is different from the source platform, Enterprise Manger will perform a data conversion. Continue to the Destination Character Set section of the page and choose the destination character set and national character set from the drop-down lists. These character sets must be compatible with the source sets. When you click Next to continue, Enterprise Manager checks the compatibility of the character sets. If the chosen character sets are flagged as incompatible, you will be returned back to the Destination Characteristics page to correct your selections.

Transportable Tablespaces with Enterprise Manager

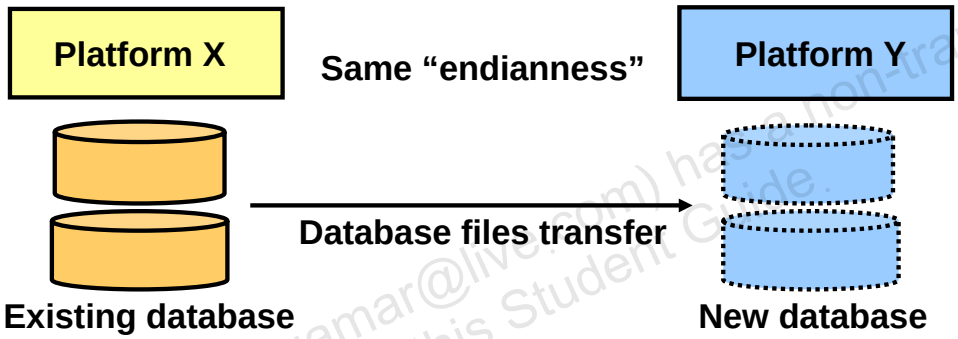


Transportable Tablespaces with Enterprise Manager (continued)

On the Schedule page, supply a meaningful description for the default job name. You can also choose to start the job immediately or schedule it for later execution. When you have made your selections, click Next to continue. On the review page, you can verify your choices before submitting the job for execution. Click the Submit Job button if the entries are correct. Click the Back button to correct any incorrect entries.

Transporting Databases

- Generalize the transportable tablespace feature.
- Data subsets can easily be distributed from a data warehousing environment to data marts, which are usually on smaller platforms.
- A database can be migrated from one platform to another very quickly.



Transporting Databases

You can use the transportable tablespace feature to migrate a database to a different platform by creating a new database on the destination platform and performing a transport of all the user tablespaces. You cannot transport the `SYSTEM` tablespace. Therefore, objects such as sequences, PL/SQL packages, and other objects that depend on the `SYSTEM` tablespace are not transported. You must either create these objects manually on the destination database, or use Data Pump to transport the objects that are not moved by transportable tablespace.

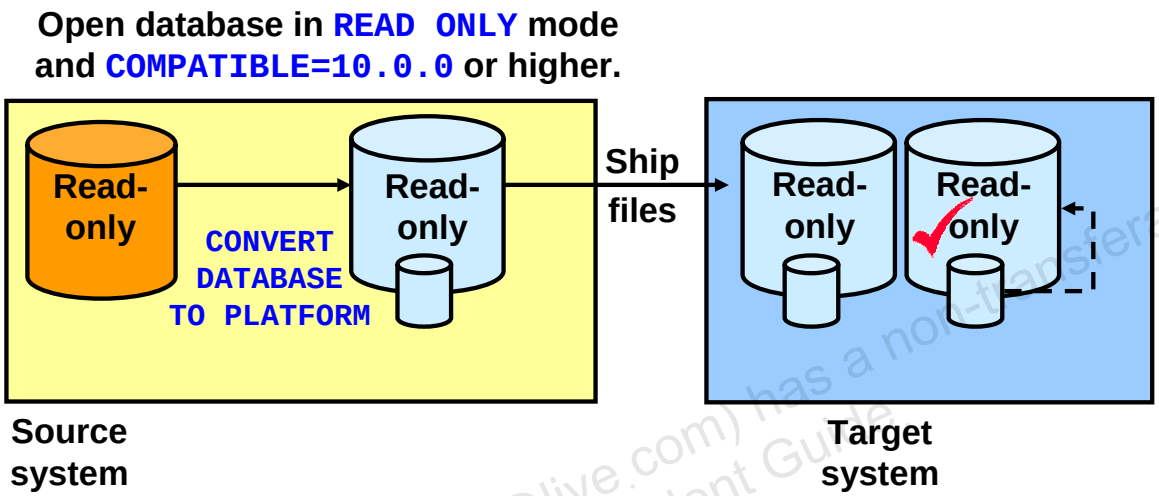
To transport databases from one platform to another, you must ensure that both the source system and the target system are running on one of the platforms that are listed in `V$TRANSPORTABLE_PLATFORM` and that both have the same endian format. For example, you can transport a database running on Linux IA (32-bit) to one of the Windows platforms.

If one or both of the databases uses Automatic Storage Management (ASM), you may need to use the `DBMS_FILE_TRANSFER` package to FTP the files.

Unlike transportable tablespace, where there is a target database to plug data into, this feature creates a new database on the target platform. The newly created database contains the same data as the source database. Except for things such as database name, instance name, and location of files, the new database also has the same settings as the source database.

Note: Transporting database is faster than using Data Pump to move data.

Database Transportation Procedure: Source System Conversion



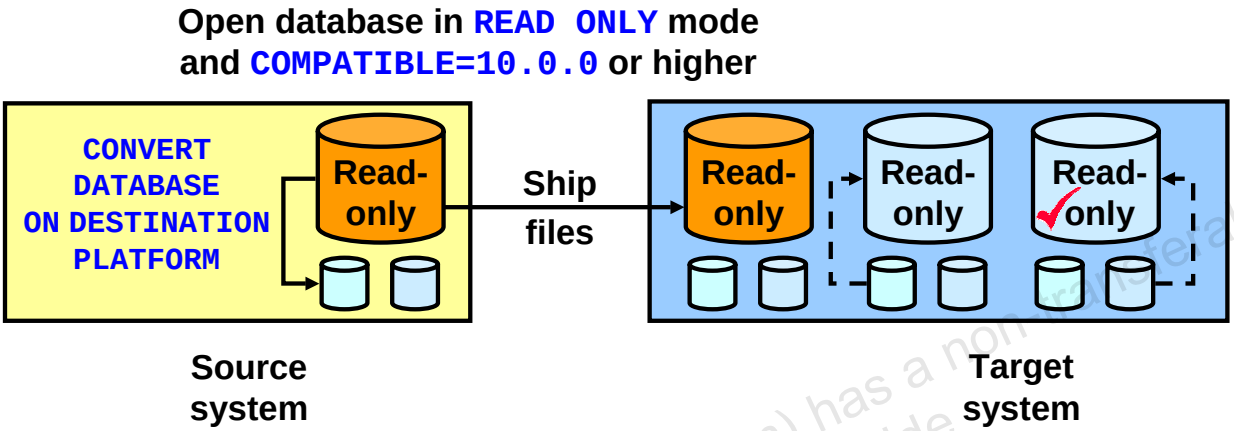
Database Transportation Procedure: Source System Conversion

Before you can transport your database, you must open it in `READ ONLY` mode. Then use RMAN to convert the necessary data files of the database.

When you do the conversion on the source platform, the RMAN `CONVERT DATABASE` command generates a script containing the correct `CREATE CONTROLFILE RESETLOGS` command that is used on the target system to create the new database. The `CONVERT DATABASE` command then converts all identified data files so that they can be used on the target system. You then ship the converted data files and the generated script to the target platform. By executing the generated script on the target platform, you create a new copy of your database.

Note: The source database must be running with the `COMPATIBLE` initialization parameter set to `10.0.0` or higher. All identified tablespaces must have been `READ WRITE` at least once since the time that `COMPATIBLE` was set to `10.0.0` or higher.

Database Transportation Procedure: Target System Conversion



Database Transportation Procedure: Target System Conversion

Before you can transport your database, you must open it in `READ ONLY` mode. Then use RMAN to convert the necessary data files of the database.

When you do the conversion on the target platform, the `CONVERT DATABASE` command (which is executed on the source system) generates only two scripts used on the target system to convert the data files, and to re-create the control files for the new database. Then, you ship the identified data files and both scripts to the target platform. After this is done, execute both scripts in the right order. The first one uses the existing RMAN `CONVERT DATAFILE` command to do the conversion, and the second issues the `CREATE CONTROLFILE RESETLOGS` SQL command with the converted data files to create the new database.

Note: The source database must be running with the `COMPATIBLE` initialization parameter set to `10.0.0` or higher. All identified tablespaces must have been `READ WRITE` at least once since `COMPATIBLE` was set to `10.0.0` or higher.

Database Transportation: Considerations

- Create the password file on the target platform.
- Transport the BFILEs used in the source database.
- The generated PFILE and transport script use OMF.
- Use DBNEWID to change the DBID.

Database Transportation: Considerations

Redo logs, control files, and tempfiles are not transported. They are re-created for the new database on the target platform. As a result, the new database on the target platform must be opened with the RESETLOGS option.

If a password file is used, it is not transported and you need to create it on the target platform. This is because the types of file names allowed for the password file are OS specific. However, the output of the CONVERT DATABASE command lists all the usernames and their system privileges, and advises to re-create the password file and add entries for these users on the target platform.

The CONVERT DATABASE command lists all the directory objects and objects that use BFILE data types or external tables in the source database. You may need to update these objects with new directory and file names. If BFILEs are used in the database, you have to transport the BFILEs.

The generated PFILE and transport script use Oracle Managed Files (OMF) for database files. If you do not want to use OMF, you must modify the PFILE and transport script.

The transported database has the same DBID as the source database. You can use the DBNEWID utility to change the DBID. In the transport script as well as the output of the CONVERT DATABASE command, you are prompted to use the DBNEWID utility to change the database ID.

Quiz

Select the statements that are true:

1. The RMAN CONVERT command performs an in-place conversion, so your input files are changed before they are transported to the destination.
2. Read/write tablespaces need to be in read-only mode at the time of an endian conversion.
3. You can use the RMAN CONVERT command for tables, tablespaces, and databases.
4. You can transport databases into a data warehouse environment.

Answer: 2, 4

Summary

In this lesson, you should have learned how to:

- Describe the concepts and use of 4 KB-sector disks
- Describe the concepts of transportable tablespaces and databases

Practice 19 Overview: Managing Space for the Database

This practice covers the following topic:

- Viewing a demonstration on “Using 4 KB-sector disks”