Platform Migration Using Transportable Database Oracle Database 11*g* and 10*g* Release 2

An Oracle White Paper
August 2008

Maximum
Availability
Architecture

Oracle Best Practices For High Availability

Platform Migration Using Transportable Database
Oracle Database 11*g* and 10*g* Release 2

EXECUTIVE OVERVIEW

Efficient and reliable methods of performing database maintenance, such as migrating a database to a new platform, have existed for many Oracle software versions. However, as maintenance windows continue to shrink and database sizes continue to grow, the importance placed on the time required to migrate a database to a more reliable or cost-effective platform has grown considerably.

Use Transportable Database to migrate to a new platform that is the same endian format as your existing database.

Platform migration is the process of moving a database from one operating-system platform to a different operating-system platform. The supported way to accomplish this prior to Oracle Database 10*g* Release 2 was to export the data from the database on the old platform, create a new database on the new platform, and import the data into the new database. This process could take a number of days for a large database. Starting with Oracle Database 10*g* Release 2, using Transportable Database and following the MAA best practices below, you can expect a 50% or greater reduction in the time it takes to complete a platform migration when compared to traditional unload/load methods.

Data Guard in Oracle Database 11*g* allows a heterogeneous mix of certain platform combinations, which can reduce platform migration time to the length of a switchover. See MetaLink Note 413484.1 for additional information.

This white paper explains how to use the Transportable Database feature in Oracle Database 10*g* Release 2 and Oracle Database 11*g* to migrate a database to a new platform that has the same endian format. Note that if the endian format is different between the source and target platforms, then you cannot use Transportable Database. Instead, use a different method, such as cross-platform Transportable Tablespaces. See the MAA white paper titled *Platform Migration using Transportable Tablespaces* for details about migrating a database to a platform that uses a different endian format.

This white paper complements the other MAA best practice papers that can be found on the Oracle Technology Network [1].

PLATFORM MIGRATION WITH TRANSPORTABLE DATABASE

Oracle Database 10*g* Release 2 added Transportable Database (TDB), which is the capability to transport an entire database (user data and the Oracle dictionary) to a platform with the same endian format. TDB uses Recovery Manager (RMAN) to convert the database files to the new target platform format and create scripts to facilitate in the migration process.

Overview

TDB requires that data files be converted to the target platform format. The data file conversion can occur on either the source system or the target system. When performing a **source system conversion**, TDB creates a second copy of all data files on the source system in the format of the target system. The converted data files must then be transferred to the proper location on the target system.

When performing a target system conversion, only data files that contain undo data require conversion. See MetaLink Note 732053.1 for details.

When performing TDB with **target system conversion**, the original data files on the source system are first transferred to the target system and placed in a staging area. RMAN is then run to convert the data files to the target system format and place them in their final location.

Using TDB to migrate a database to a new platform of the same endian format consists of the following high-level steps:

1. Check prerequisites
2. Prepare for the platform migration
3. Start the database in `READ ONLY` mode
4. Verify the database is ready for migration
5. Run the RMAN `CONVERT DATABASE` command
6. Move necessary files to the target system
7. Complete the migration

Oracle E-Business Suite Database Platform Migration

If you are using TDB to migrate an Oracle E-Business Suite database to a new platform that is using the same endian format, additional information can be found in the following Oracle MetaLink documents:

- E-Business Suite 11*i* - MetaLink Note 729309.1
- E-Business Suite R12 - MetaLink Note 734763.1

Best Practices

To keep the database outage as short as possible, the most important best practice is to properly prepare for and test the platform migration. Additional best practices are aimed at reducing the amount of work required during the platform migration and doing the work that must be done during downtime as efficiently as possible.

Plan the Migration and Validate in a Test Environment

The Oracle Upgrade Companion provides information and guidance on database upgrade. Much of the content provided also applies to platform migration using TDB. To visit the Oracle Upgrade Companion, go to MetaLink Note 466181.1.

As with most planned database maintenance activities, the probability of a successful platform migration is very high when a well-defined migration plan is validated in a suitable test environment before migrating the production database. The majority of time spent on a platform migration project should be spent in the planning, preparation, and testing phases. Review and follow the best practice guidance provided by the Oracle Upgrade Companion in MetaLink Note 466181.1, focusing specifically on the Planning, Prepare and Preserve, and Post sections.

Note: TDB requires the same Oracle Database software version on the source and target systems; hence, a database upgrade cannot be accomplished simultaneous with the platform migration. However, best practices for a database upgrade, as documented in the Oracle Upgrade Companion, also apply to any planned database maintenance activity, including platform migration.

Optimize File Conversion and Transfer

The largest time and resource requirements are for manipulating the data files – copying them to the new system and converting them to the format needed for the new platform. RMAN does not convert data files to the new platform format in place; hence an essential requirement is the ability to store an extra complete copy of the data files on either the source or the target system. Decide which conversion location – source system or target system – will provide the least downtime. Here are some things to consider:

- Data file conversion is a highly I/O intensive operation, so most systems will find this to be the limiting resource. If one system has a significantly better I/O subsystem, perform the data file conversion on that system.
- Investigate methods that will either eliminate the data file transfer step, or allow the data file transfer and the data file conversion to occur in the same step. Consider one of the following options:
    - Mount or replicate the source system storage device directly to the target system. The target system must be able to recognize and mount, in read-only mode, the source system volume manager disk group or file system. For example, a storage device connected to a Solaris system containing a UFS file system holding data files can be connected to a Linux system and the UFS file system mounted in read-only mode. During a target system conversion, RMAN reads from the read-only UFS file system and writes converted data files to a Linux EXT3 file system or to an ASM disk group.

- If performing a source system conversion, NFS mount on the source system the file system that contains the final data file location on the target system.

  ```
  source_host> mount target:/oradata /mnt/mydb
  ```

- If performing a target system conversion, NFS mount on the target system the file system that contains the data files on the source system.

  ```
  target_host> mount source:/oradata/PROD/datafile
  /mnt/mydb
  ```

- Only data files that contain undo data require conversion. However, the default behavior of CONVERT DATABASE is to perform a conversion on all data files. Eliminating unnecessary data file conversion, which is possible with target system conversion, can significantly reduce platform migration time. See MetaLink Note 732053.1 for details.

See MetaLink Note 417455.1 for additional information about using the RMAN PARALLEL option with Transportable Database in Oracle Database 10*g*.

- Use all available computer resources to do the CONVERT DATAFILE operations by running them in parallel. RMAN allows the PARALLEL option to be specified with the CONVERT DATABASE command. For additional information on RMAN parallelism and tuning RMAN, see the *Oracle Database Backup and Recovery Advanced User's Guide* [2].

- Mount storage containing the data files on the target system. TDB requires two operations on the data files: conversion to the target platform format, and transfer to the target system. Typically these operations occur as separate steps. For example, when performing a target system conversion, the data files are first transferred to the target system, and then converted using the CONVERT DATAFILE command in a separate step.

  These two operations can occur in a single step if the source data file location is mounted on the target system. An example is rezoning a storage area network so that the volumes containing the data files are available to the target system. When the file conversion is run, the output of the CONVERT DATAFILE commands place the data files in their final location without the need of a separate transfer step. The supportability of rezoning storage between platforms is operating system dependent. Contact your storage vendor for details.

- NFS mounting the source file system containing the data files on to the target system. When the file conversion is run, the output of the CONVERT DATAFILE commands place the data files in their final location without the need of a separate transfer step.

Leverage a Physical Standby Database

Utilizing a physical standby database to reduce downtime increases complexity of platform migration. The reduction in downtime must be weighed against the added complexity for your environment.

Transportable database commands can be run against a physical standby database that is open in READ ONLY mode. If the target database will reside at a remote location, a physical standby database running at the remote location on the source platform will eliminate the need to transfer the source data files across a wide area network during the outage window, thus reducing downtime.

If a physical standby database is to be used to eliminate a network transfer to a new data center during the outage, the source database must not process new transactions and the physical standby database must apply all changes that occurred on the source database prior to performing the TDB operations.

Refer to the *Oracle Data Guard Concepts and Administration* [3] and *Oracle Database High Availability Best Practices* [7] manuals for instructions on setting up and configuring a physical standby database.

PERFORMING A PLATFORM MIGRATION WITH TRANSPORTABLE DATABASE

To perform a platform migration using transportable database, follow these steps:

Step 1: Check Prerequisites

Before attempting a platform migration with TDB, check the following:

Verify TDB Support for Target Platform

Before attempting a platform migration with TDB, verify the target platform is supported for TDB by your source platform. Query the view `V$DB_TRANSPORTABLE_PLATFORM` for the target platform name. Here is example output from a database on 'Solaris Operating System (x86)' platform.

```
SQL> select platform_name from v$db_transportable_platform;

PLATFORM_NAME
---------------------------------------------
Microsoft Windows IA (32-bit)
Linux IA (32-bit)
HP Tru64 UNIX
Linux IA (64-bit)
HP Open VMS
Microsoft Windows IA (64-bit)
Linux 64-bit for AMD
Microsoft Windows 64-bit for AMD
Solaris Operating System (x86)

9 rows selected.
```

If the target platform does not appear in the output from `V$DB_TRANSPORTABLE_PLATFORM`, then the database cannot be migrated using TDB.

Check Documentation for TDB Restrictions

In addition to checking the target platform, check the Oracle documentation for restrictions and limitations.

See the *Oracle Database Backup and Recovery Advanced User's Guide* [2] for additional information.

Target System Software Version

The target system must have the same Oracle software version and patches installed as the source system. This includes the same patch set version, critical patch updates, and patch set exceptions. The software should be used to create a sample database to ensure it is operational. Refer to *Oracle Universal Installer and OPatch User's Guide* [4], for details on using the OPatch utility to determine the currently installed versions and patches.

Choose Source or Target System Conversion

Decide which platform will be used to perform the conversion of the data files. Choose the system that best optimizes the data file transfer and conversion process, per the best practices above.

Step 2: Prepare for the Platform Migration

Identify External Files and Directories

Identify directories external to the source database that need to be created on the target system, and external table files and BFILEs that will need to be moved to the target system.

The PL/SQL function `CHECK_EXTERNAL` identifies external tables, directories, and BFILEs that need to be moved during the migration so the target database is complete when the process is finished.

```
SQL> set serveroutput on
SQL> declare x boolean;
     begin x := dbms_tdb.check_external; end;

The following external tables exist in the database:
SH.SALES_TRANSACTIONS_EXT
The following directories exist in the database:
SYS.DATA_FILE_DIR, SYS.LOG_FILE_DIR, SYS.TTSDIR
The following BFILEs exist in the database:
PM.PRINT_MEDIA

PL/SQL procedure successfully completed.
```

- Directory objects must be created on the target system. Query `DBA_DIRECTORIES` on the source database to determine the file system locations that must exist on the target system for the directory objects to be usable.

  ```
  SQL> select directory_path from dba_directories;
  ```

```
DIRECTORY_PATH
-----------------------------------
/extdata/orcl/
/u01/app/oracle/admin/orcl/tts
/u01/app/oracle/admin/orcl/dpdump/
```

Ensure that each directory listed in the view DBA_DIRECTORIES points to a valid file system directory, or ASM disk group or directory on the target system. Accomplish this by either creating each directory on the target system, or altering the DIRECTORY_PATH to a valid directory when the target database is open after the migration process. For example:

```
$ mkdir -p /extdata/orcl
$ mkdir -p /u01/app/oracle/admin/orcl/tts
$ mkdir -p /u01/app/oracle/admin/orcl/dpdump/
```

Do not add additional external tables to the source database until the platform migration is complete.

- Identify external table files that will need to be transferred to the target system when indicated in a later step.

  To identify external table files, run the following query

  ```
  SQL> select directory_path||‘/’||location External_file_path
       from dba_directories a, dba_external_locations b
       where a.directory_name=b.directory_name;

  EXTERNAL_FILE_PATH
  --------------------------
  /extdata/orcl/sales1v3.dat
  ```

Do not initialize additional BFILEs in the source database until the platform migration is complete.

- Identify BFILE files that will need to be transferred to the target system when indicated in a later step.

  To identify directories that contain BFILEs, run the following SQL script from the appendix:

  ```
  SQL> @tdb_get_bfile_dirs.sql
  The following directories contain external files for BFILE
  columns
  Copy the files within these directories to the same path on
  the target system

  /extdata/bfiles/image/
  /extdata/bfiles/product/
  /extdata/bfiles/bdir/

  There are 3 directories, 417 total BFILEs
  ```

  If it is necessary to list all BFILE external files, then run the script tdb_get_bfiles.sql script from the appendix.

For additional information on the CHECK_EXTERNAL function of the DBMS_TDB PL/SQL package, refer to the *Oracle Database PL/SQL Packages and Types Reference* [5].

Shut Down the Application

Disconnect users and shutdown all application server processes. Users cannot use any application served by the database until the migration to the new platform is complete.

Application downtime begins with this step.

Export OLAP Analytic Workspaces

Oracle OLAP stores the OLAP `DECIMAL` data type in a hardware-dependent manner. Whenever changing platforms, all OLAP analytic workspaces (AWs) must be exported from the source database before the TDB process begins and imported into the target database after the TDB process is complete. AWs are exported and imported using the `DBMS_AW.EXECUTE` PL/SQL procedure. For additional information, see Oracle OLAP Reference [9] for Oracle Database 10*g*, or Oracle OLAP DML Reference for Oracle Database 11*g* [10].

The examples in MetaLink Note 352306.1 can be followed to perform OLAP AW export and import. While this document is primarily used for the migration of OLAP from 32 to 64-bit, this document properly describes how to migrate OLAP from one environment to another regardless of bit size.

Step 3: Start the database in READ ONLY mode

TDB requires that the source database be opened in `READ ONLY` mode. The source database will be unavailable from this step forward.

```
SQL> shutdown immediate;
SQL> startup mount;
SQL> alter database open read only;
```

Step 4: Verify the database is ready for migration

To ensure the database is ready for migration, run the `DBMS_TDB.CHECK_DB` function to verify that the database can be migrated to the target platform and that the database is in the proper state to be migrated. If using a physical standby database, this function is run on the standby database instead of on the primary.

The target platform name specified when running the DBMS_TDB.CHECK_DB function (‘Linux IA (32-bit)’ in the example provided) must match exactly the entry in the V$DB_TRANSPORTABLE_PLATFORM view.

```
SQL> set serveroutput on
SQL> declare
       retcode boolean;
     begin
       retcode := dbms_tdb.check_db(‘Linux IA (32-bit)',
        dbms_tdb.skip_none);
     end;
```

Any condition reported by `CHECK_DB` must be resolved before TDB can proceed. For details of the checks performed by `DBMS_TDB.CHECK_DB`, refer to the *Oracle Database PL/SQL Packages and Types Reference* [5] manual.

Step 5: Run the RMAN CONVERT DATABASE Command

Once the source database is ready for migration, run the `CONVERT DATABASE` command in RMAN. The steps differ slightly depending on the type of conversion chosen – source system or target system. If using a physical standby database, run this step on the physical standby database instead of on the source database.

For details about the `CONVERT DATABASE` command, see the *Oracle Database Backup and Recovery Reference* [6].

Source System Conversion

When performing a source system conversion, RMAN creates converted data files on the source system. The example provided shows a platform migration from 'Solaris Operating System (x86)’ to ‘Linux IA (32-bit)’.

The target platform name specified in the TO PLATFORM clause (‘Linux IA (32-bit)’ in the example provided) must match exactly the entry in the V$DB_TRANSPORTABLE_PLATFORM view.

```
% rman
RMAN> connect target /
RMAN> convert database
      transport script '/tmp/transport_mydb.sql'
      new database 'mydb'
      to platform 'Linux IA (32-bit)’
      parallelism 4
      format '/tmp/mydb'
      db_file_name_convert ‘/u01/oradata/PROD/datafile/’,’/stage/’;
```

See MetaLink Note 417455.1 for additional information about using the RMAN PARALLEL option with Transportable Database in Oracle Database 10*g*.

The `CONVERT DATABASE` command specified in the example creates a transport script named `/tmp/transport_mydb.sql`, a PFILE named `/tmp/init_mydb.ora`, and a copy of all data files in the `/stage` directory in the format of the target platform ‘Linux IA (32-bit)’. See the appendix for example source system conversion output.

To reduce overall outage time, NFS mount the target system’s final data file location and create the converted data files on the NFS mounted directory, directly in the final destination on the target system.

```
# mount target:/u01/oradata/MYDB/datafile /mnt/mydb
RMAN> convert database
      transport script '/tmp/transport_mydb.sql'
      new database 'mydb'
      to platform 'Linux IA (32-bit)’
      parallelism 4
      format '/tmp/mydb'
      db_file_name_convert ‘/u01/oradata/PROD/datafile/’,’/mnt/mydb/’;
```

Target System Conversion

When performing a target system conversion, RMAN creates an RMAN script to be used on the target system to convert all data files. The following example shows a platform migration from 'Solaris Operating System (x86)’ to ‘Linux IA (32-bit)’.

```
% rman
RMAN> connect target /
RMAN> convert database on target platform
      convert script ‘/tmp/convert_mydb.rman’
      transport script '/tmp/transport_mydb.sql'
      new database 'mydb'
      format '/tmp/mydb%U'
      db_file_name_convert '/u01/oradata/PROD/datafile','/tmp';
```

The `CONVERT DATABASE` command in the example creates a convert script named `/tmp/convert_mydb.rman`, a transport script named `/tmp/transport_mydb.sql`, and a PFILE named

/tmp/init_mydb.ora. See the appendix for example target system conversion output.

Step 6: Move necessary files to target system

Commands, like ftp, that can transfer files using ASCII or binary mode, must transfer Oracle data files using binary mode.

Once the CONVERT DATABASE command is complete, transfer the following files to the target system using operating system utilities (for example: FTP, SCP):

- **PFILE initialization file**—Place these files where the database initialization files are located (for example: $ORACLE_HOME/dbs).
- Transport SQL script
- **External table file system files**—Place these files on the target system in the file system that corresponds to the same directory object as the source system.

  For example:

  ```
  $ scp /extdata/orcl/sales1v3.dat \
  target_host:/extdata/orcl/sales1v3.dat
  ```

- **BFILE file system files**—Place these files on the target system in the file system that corresponds to the same directory object as the source system.

  For example:

  ```
  $ scp /oracle/product_media/monitor.jpg \
  target_host:/oracle/product_media/monitor.jpg
  $ scp /oracle/product_media/mousepad.jpg \
  target_host:/oracle/product_media/mousepad.jpg
  $ scp /oracle/product_media/keyboard.jpg \
  target_host:/oracle/product_media/keyboard.jpg
  $ scp /oracle/product_media/modem.jpg \
  target_host:/oracle/product_media/modem.jpg
  ```

Source System Conversion

In addition to the common files listed above, if performing a source system conversion, the following files must be transferred to the target system:

- **Converted data files**—Place these files in the final target system location, such as the location referenced by db_file_create_dest.

  ```
  $ scp /stage/* target:/u01/oradata/MYDB/datafile
  ```

  If, during the CONVERT DATABASE step, the target system data file location was NFS mounted and the converted data files were placed on the target system, then you do not need to transfer the converted data files again.

If using a physical standby database to assist in the migration, the converted data files will exist on the system where the physical standby database is running.

Target System Conversion

In addition to the common files listed above in step 6, if performing a target system conversion, transfer the following files to the target system:

Only data files that contain undo data require conversion. Any data files that will forego conversion are placed directly in their final location on the target system and are removed from the convert script. See MetaLink Note 732053.1 for details.

- **Unconverted data files**—Place these files in a staging area. The convert RMAN script reads data files from this location and writes the converted data files to their final location.

  To reduce overall outage time, instead of transferring the unconverted data files using a command like FTP, you should NFS mount the source system's data file location on the target system. During the conversion process, this allows the data files to be read from their original source location and written in the converted format to their final target location. See step 6 for details about modifying the convert RMAN script.

  ```
  # mount source:/u01/oradata/PROD/datafile /mnt/mydb
  ```

  If using a physical standby database to assist in the migration, the unconverted data files will exist on the system where the physical standby database is running.

  As discussed in the Executive Overview section, data files that do not contain UNDO segments require no conversion.

- Convert RMAN script

Step 7: Complete the migration

The final steps required to complete the migration differ slightly depending on the type of conversion chosen – source system or target system.

Review and Edit the PFILE

For all migrations, the PFILE created on the source system and transferred to the target system must be reviewed to ensure that directory paths and file locations are accurate for the target system. The parameters that need to be changed are grouped together at the top of the PFILE.

See MetaLink Note 417455.1 for additional information about using the RMAN PARALLEL option with Transportable Database in Oracle Database 10*g*.

Only data files that contain undo data require conversion. Any data files that will forego conversion are placed directly in their final location on the target system and are removed from the convert script. See MetaLink Note 732053.1 for details.

Target System Conversion Only –Review and Run the Convert RMAN Script

When performing a target system conversion, the data files are moved to the target system in the source system format. Review the convert RMAN script to ensure the file locations are accurate. The following should be reviewed and changed as necessary:

- The filename specified for `CONVERT DATAFILE` should be the unconverted data file that was transferred from the source system. If the source data file location is NFS mounted to the target system (instead of transferring the unconverted data files over the network), then specify the location of the NFS mount.
- The `FORMAT` specification should indicate the final location of the converted data file. This final location must match the location specified in the transport SQL script. See the Review and Run the Transport SQL Script section for more details.
- In Oracle Database 11*g*, ensure the PFILE clause in the `STARTUP NOMOUNT` command refers to the location where the PFILE was copied to the target system.

After reviewing and editing the convert script, as necessary, start the target instance in `NOMOUNT` mode using the PFILE transferred from the source system, and run the script. After running the convert script, shut down the target instance.

Note: Explicitly running `STARTUP NOMOUNT` is only necessary in Oracle Database 10*g*. In Oracle Database 11*g*, the `STARTUP NOMOUNT` command is included in the convert script.

```
RMAN> connect target /
RMAN> startup nomount pfile=<location of pfile> # 10g only
RMAN> @convert_mydb.rman
RMAN> shutdown
```

Review and Run the Transport SQL Script

For all migrations, the final step is to run the transport SQL script that was created by the RMAN `CONVERT DATABASE` command. First, edit the script on the target system to ensure that locations for all referenced files are correct. Review the following sections for accuracy and make changes, as necessary:

- Ensure that the PFILE referenced in the `STARTUP` command in the transport script points to the location where the PFILE was placed when it was transferred from the source system.
- Ensure the `DATAFILE` locations in the `CREATE CONTROLFILE` statement refer to the final location of the converted datafiles on the target system. If it is a target system conversion, then the `DATAFILE` locations should match where the `CONVERT DATAFILE` command placed its output.

- Ensure that the LOGFILE locations in the CREATE CONTROLFILE statement refer to the desired location of the online redo log files on the target system. If a location is not specified, then Oracle Managed Files is used to name and place the files.
- Ensure the TEMPFILE locations, if specified, are accurate for new tempfiles being created for temporary tablespaces.

After reviewing and editing the transport script (as necessary), run the script. For example:

```
SQL> connect / as sysdba;
SQL> @transport_mydb.sql
```

Once the transport script completes, review the output for errors.

Import OLAP Analytic Workspaces

Use the examples in Oracle MetaLink Note 352306.1 to perform OLAP AW export and import. While this MetaLink note is primarily used for the migration of OLAP from 32-bit to 64-bit, this document properly describes how to migrate OLAP from one environment to another regardless of bit size.

Import OLAP analytic workspaces (AWs) exported previously using the DBMS_AW.EXECUTE PL/SQL procedure. AWs are exported and imported using the DBMS_AW.EXECUTE PL/SQL procedure. For additional information, refer to Oracle OLAP Reference [9] for Oracle Database 10*g*, or Oracle OLAP DML Reference for Oracle Database 11*g* [10].

Start the Application

The final step is to start the application, directing connections to the database running on the new target platform.

CONCLUSION

Use Transportable Database to migrate a database between platforms that share the same endian format to incur less downtime than with the traditional method of using unload and load.

APPENDIX

The `tdb_get_bfile_dirs.sql` script is a sample script to get all file system directories that contain BFILEs referenced in the database.

The tdb_get_bfile_dirs.sql Script

```
REM
REM List all directories that contain BFILEs
REM
set serveroutput on format wrap;
set feedback off;

declare
  type cur_type is REF CURSOR;
  v_cur cur_type;
  v_sqlstmt varchar2(100);
  v_bfile_loc bfile;
  v_bfile_dir_name varchar2(30);
  v_bfile_filename varchar2(250);
  v_bfile_realpath varchar2(4000);
  type array_type is table of number index by varchar2(512);
  bfile_dirs array_type;
  mydir varchar2(512);
  total_bfiles number := 0;
begin
  -- loop through all columns that are BFILE type
  for bf in
    (select owner,table_name,column_name
     from dba_tab_cols
     where data_type='BFILE')
  loop
    v_sqlstmt:='select '||bf.column_name||' from '
                ||bf.owner||'.'||bf.table_name;
    open v_cur for v_sqlstmt;
    loop
      fetch v_cur into v_bfile_loc;
      exit when v_cur%notfound;

      -- get BFILE directory alias and filename
      dbms_lob.filegetname(v_bfile_loc, v_bfile_dir_name,
                           v_bfile_filename);

      if bfile_dirs.exists(v_bfile_dir_name) then
        bfile_dirs(v_bfile_dir_name) := bfile_dirs(v_bfile_dir_name) +
1;
      else
        bfile_dirs(v_bfile_dir_name) := 1;
      end if;

    end loop;
    close v_cur;
  end loop;

  dbms_output.put_line(' ');
  dbms_output.put_line('The following directories contain external
files for BFILE columns');
  dbms_output.put_line('Copy the files within these directories to the
same path on the target system');
  dbms_output.put_line(' ');

  -- loop through array of all directories
  mydir := bfile_dirs.first;
  while mydir is not null loop
```

```
    -- resolve the directory alias to a full path
    select directory_path
    into v_bfile_realpath
    from all_directories
    where directory_name = mydir;

    dbms_output.put_line(v_bfile_realpath);

    total_bfiles := total_bfiles + bfile_dirs(mydir);

    mydir := bfile_dirs.next(mydir);
  end loop;
  dbms_output.put_line(' ');
  dbms_output.put_line('There are ' || bfile_dirs.count
                       || ' directories, ' || total_bfiles
                       || ' total BFILEs');
  dbms_output.put_line(' ');
end;
/
```

tdb_get_bfiles.sql is a sample script to get the file system path of all BFILEs referenced in the database.

The tdb_get_bfiles.sql Script

```
REM
REM List all BFILE external files in database
REM
set serveroutput on;
set feedback off;

declare
  type cur_type is REF CURSOR;
  v_cur cur_type;
  v_sqlstmt varchar2(100);
  v_bfile_loc bfile;
  v_bfile_dir_name varchar2(30);
  v_bfile_filename varchar2(250);
  v_bfile_realpath varchar2(4000);
begin
  -- loop through all columns that are BFILE type
  for bf in
    (select owner,table_name,column_name
     from dba_tab_cols
     where data_type='BFILE')
  loop
    dbms_output.put_line('External files for BFILE column '
                         || bf.column_name || ' in table '
                         || bf.owner || '.' || bf.table_name);
    v_sqlstmt:='select ' || bf.column_name || ' from '
                || bf.owner || '.' || bf.table_name;
    open v_cur for v_sqlstmt;
    loop
      fetch v_cur into v_bfile_loc;
      exit when v_cur%notfound;

      -- get BFILE directory alias and filename
      dbms_lob.filegetname(v_bfile_loc, v_bfile_dir_name,
                           v_bfile_filename);

      -- resolve the directory alias to a full path
      select directory_path
      into v_bfile_realpath
      from all_directories
      where directory_name = v_bfile_dir_name;
      dbms_output.put_line(v_bfile_realpath || '/'
                           || v_bfile_filename);

    end loop;
    close v_cur;
  end loop;
end;
/
```

Example output of a source system conversion.

Example Source System Conversion

```
RMAN> convert database
2> transport script '/tmp/transport_mydb.sql'
3> new database 'mydb'
4> to platform 'Solaris Operating System (x86)'
5> format '/tmp/mydb'
6> db_file_name_convert '/u01/oradata/SAMPL2/datafile/','/tmp/rman/'
7> ;

Starting convert at 18-NOV-06
using channel ORA_DISK_1

External table SH.SALES_TRANSACTIONS_EXT found in the database

Directory SYS.MEDIA_DIR found in the database
Directory SYS.DATA_FILE_DIR found in the database
Directory SYS.LOG_FILE_DIR found in the database
Directory SYS.DM_PMML_DIR found in the database
Directory SYS.TTSDIR found in the database
Directory SYS.WORK_DIR found in the database
Directory SYS.ADMIN_DIR found in the database
Directory SYS.DATA_PUMP_DIR found in the database

BFILE PM.PRINT_MEDIA found in the database

User SYS with SYSDBA and SYSOPER privilege found in password file
channel ORA_DISK_1: starting datafile conversion
input datafile fno=00001
name=/u01/oradata/SAMPL2/datafile/o1_mf_system_2k5c82h1_.dbf
converted datafile=/tmp/rman/o1_mf_system_2k5c82h1_.dbf
channel ORA_DISK_1: datafile conversion complete, elapsed time:
00:01:05
channel ORA_DISK_1: starting datafile conversion
input datafile fno=00002
name=/u01/oradata/SAMPL2/datafile/o1_mf_undotbs1_2k5c8xmc_.dbf
converted datafile=/tmp/rman/o1_mf_undotbs1_2k5c8xmc_.dbf
channel ORA_DISK_1: datafile conversion complete, elapsed time:
00:01:15
channel ORA_DISK_1: starting datafile conversion
input datafile fno=00003
name=/u01/oradata/SAMPL2/datafile/o1_mf_sysaux_2k5c8zll_.dbf
converted datafile=/tmp/rman/o1_mf_sysaux_2k5c8zll_.dbf
channel ORA_DISK_1: datafile conversion complete, elapsed time:
00:00:45
channel ORA_DISK_1: starting datafile conversion
input datafile fno=00004
name=/u01/oradata/SAMPL2/datafile/o1_mf_example_2jm6wwvt_.dbf
converted datafile=/tmp/rman/o1_mf_example_2jm6wwvt_.dbf
channel ORA_DISK_1: datafile conversion complete, elapsed time:
00:00:15
channel ORA_DISK_1: starting datafile conversion
```

```
input datafile fno=00005
name=/u01/oradata/SAMPL2/datafile/o1_mf_users_2jm6trkj_.dbf
converted datafile=/tmp/rman/o1_mf_users_2jm6trkj_.dbf
channel ORA_DISK_1: datafile conversion complete, elapsed time:
00:00:01
Run SQL script /tmp/transport_mydb.sql on the target platform to create
database
Edit init.ora file /tmp/init_mydb.ora. This PFILE will be used to
create the database on the target platform
To recompile all PL/SQL modules, run utlirp.sql and utlrp.sql on the
target platform
To change the internal database identifier, use DBNEWID Utility
Finished backup at 18-NOV-06
```

Example output of a target system conversion.

Target System Conversion

```
RMAN> convert database on target platform
2> convert script '/tmp/convert_mydb.rman'
3> transport script '/tmp/transport_mydb.sql'
4> new database 'mydb'
5> format '/tmp/mydb%U'
6> db_file_name_convert '/u01/oradata/SAMPL2/datafile','/tmp/rman'
7> ;

Starting convert at 18-NOV-06
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=159 devtype=DISK

External table SH.SALES_TRANSACTIONS_EXT found in the database

Directory SYS.MEDIA_DIR found in the database
Directory SYS.DATA_FILE_DIR found in the database
Directory SYS.LOG_FILE_DIR found in the database
Directory SYS.DM_PMML_DIR found in the database
Directory SYS.TTSDIR found in the database
Directory SYS.WORK_DIR found in the database
Directory SYS.ADMIN_DIR found in the database
Directory SYS.DATA_PUMP_DIR found in the database

BFILE PM.PRINT_MEDIA found in the database

User SYS with SYSDBA and SYSOPER privilege found in password file
channel ORA_DISK_1: starting to check datafiles
input datafile fno=00001
name=/u01/oradata/SAMPL2/datafile/o1_mf_system_2k5c82h1_.dbf
channel ORA_DISK_1: datafile checking complete, elapsed time: 00:00:00
channel ORA_DISK_1: starting to check datafiles
input datafile fno=00002
name=/u01/oradata/SAMPL2/datafile/o1_mf_undotbs1_2k5c8xmc_.dbf
channel ORA_DISK_1: datafile checking complete, elapsed time: 00:00:00
channel ORA_DISK_1: starting to check datafiles
input datafile fno=00003
name=/u01/oradata/SAMPL2/datafile/o1_mf_sysaux_2k5c8zll_.dbf
channel ORA_DISK_1: datafile checking complete, elapsed time: 00:00:00
channel ORA_DISK_1: starting to check datafiles
input datafile fno=00004
name=/u01/oradata/SAMPL2/datafile/o1_mf_example_2jm6wwvt_.dbf
channel ORA_DISK_1: datafile checking complete, elapsed time: 00:00:00
channel ORA_DISK_1: starting to check datafiles
```

```
input datafile fno=00005
name=/u01/oradata/SAMPL2/datafile/o1_mf_users_2jm6trkj_.dbf
channel ORA_DISK_1: datafile checking complete, elapsed time: 00:00:00
Run SQL script /tmp/transport_mydb.sql on the target platform to create
database
Edit init.ora file /tmp/init_mydb 00i58vro_1_0.ora. This PFILE will be
used to create the database on the target platform
Run RMAN script /tmp/convert_mydb.rman on target platform to convert
datafiles
To recompile all PL/SQL modules, run utlirp.sql and utlrp.sql on the
target platform
To change the internal database identifier, use DBNEWID Utility
Finished backup at 18-NOV-06
```

Example transport script created by the CONVERT DATABASE command. This script is copied to the target system, modified to fix file pathnames, and run to complete the platform migration.

Transport Script

```
-- The following commands will create a new control file and use it
-- to open the database.
-- Data used by Recovery Manager will be lost.
-- The contents of online logs will be lost and all backups will
-- be invalidated. Use this only if online logs are damaged.

-- After mounting the created controlfile, the following SQL
-- statement will place the database in the appropriate
-- protection mode:
--  ALTER DATABASE SET STANDBY DATABASE TO MAXIMIZE PERFORMANCE

STARTUP NOMOUNT PFILE='/tmp/init_mydb.ora'
CREATE CONTROLFILE REUSE SET DATABASE "MYDB" RESETLOGS  ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 908
LOGFILE
  GROUP 1 SIZE 10M,
  GROUP 2 SIZE 10M,
  GROUP 3 SIZE 10M
DATAFILE
  '/tmp/rman/o1_mf_system_2k5c82h1_.dbf',
  '/tmp/rman/o1_mf_undotbs1_2k5c8xmc_.dbf',
  '/tmp/rman/o1_mf_sysaux_2k5c8zll_.dbf',
  '/tmp/rman/o1_mf_example_2jm6wwvt_.dbf',
  '/tmp/rman/o1_mf_users_2jm6trkj_.dbf'
CHARACTER SET WE8ISO8859P1
;

-- Database can now be opened zeroing the online logs.
ALTER DATABASE OPEN RESETLOGS;

-- Commands to add tempfiles to temporary tablespaces.
-- Online tempfiles have complete space information.
-- Other tempfiles may require adjustment.
ALTER TABLESPACE TEMP ADD TEMPFILE
     SIZE 28311552  AUTOEXTEND ON NEXT 655360  MAXSIZE 32767M;
-- End of tempfile additions.
--

set echo off
prompt
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt * Your database has been created successfully!
```

```
prompt * There are many things to think about for the new database.
Here
prompt * is a checklist to help you stay on track:
prompt * 1. You may want to redefine the location of the directory
objects.
prompt * 2. You may want to change the internal database identifier
(DBID)
prompt *    or the global database name for this database. Use the
prompt *    NEWDBID Utility (nid).
prompt
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

SHUTDOWN IMMEDIATE
STARTUP UPGRADE PFILE='/tmp/init_mydb.ora'
@@ ?/rdbms/admin/utlirp.sql
SHUTDOWN IMMEDIATE
STARTUP PFILE='/tmp/init_mydb.ora'
-- The following step will recompile all PL/SQL modules.
-- It may take serveral hours to complete.
@@ ?/rdbms/admin/utlrp.sql
set feedback 6;
```

Example convert script created by the CONVERT DATABASE command for a target system conversion using Oracle Database 10*g*. This script is copied to the target system, modified to fix file pathnames, and run to convert the datafiles.

Convert Script—Oracle Database 10*g*

Here is the convert script created by the example Target System Conversion command.

```
RUN {

  CONVERT DATAFILE
'/u01/oradata/SAMPL2/datafile/o1_mf_system_2k5c82h1_.dbf'
  FROM PLATFORM 'Solaris Operating System (x86)'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-SYSTEM_FNO-
1_0ji58vro';


  CONVERT DATAFILE
'/u01/oradata/SAMPL2/datafile/o1_mf_undotbs1_2k5c8xmc_.dbf'
  FROM PLATFORM 'Solaris Operating System (x86)'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-UNDOTBS1_FNO-
2_0ki58vro';


  CONVERT DATAFILE
'/u01/oradata/SAMPL2/datafile/o1_mf_sysaux_2k5c8zll_.dbf'
  FROM PLATFORM 'Solaris Operating System (x86)'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-SYSAUX_FNO-
3_0li58vro';


  CONVERT DATAFILE
'/u01/oradata/SAMPL2/datafile/o1_mf_example_2jm6wwvt_.dbf'
  FROM PLATFORM 'Solaris Operating System (x86)'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-EXAMPLE_FNO-
4_0mi58vro';
```

```
  CONVERT DATAFILE
'/u01/oradata/SAMPL2/datafile/o1_mf_users_2jm6trkj_.dbf'
  FROM PLATFORM 'Solaris Operating System (x86)'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-USERS_FNO-5_0ni58vro';

}
```

Convert Script with Parallelism Fix—Oracle Database 10*g*

This is the same convert script with workarounds for which:

- The resolutions to the issues that are documented in MetaLink Note 417455.1 have been applied.
- The script has been edited to match the proper location of the existing unconverted files (staging area /stage) and the proper target location for the converted files (ASM diskgroup +DATA) on the target system.

Note: These changes are necessary only for Oracle Database 10*g*.

```
RUN {
  CONVERT DATAFILE
'/stage/SAMPL2/datafile/o1_mf_system_2k5c82h1_.dbf',
'/stage/SAMPL2/datafile/o1_mf_undotbs1_2k5c8xmc_.dbf',
'/stage/SAMPL2/datafile/o1_mf_sysaux_2k5c8zll_.dbf',
'/stage/SAMPL2/datafile/o1_mf_example_2jm6wwvt_.dbf',
'/stage/SAMPL2/datafile/o1_mf_users_2jm6trkj_.dbf'
  FROM PLATFORM 'Solaris Operating System (x86)'
  DB_FILE_NAME_CONVERT ‘/stage/SAMPL2/datafile’,’+DATA’;
}
```

Convert Script—Oracle Database 11*g*

Example convert script created by the CONVERT DATABASE command for a target system conversion using Oracle Database 11*g*. This script is copied to the target system, modified to fix file pathnames, and run to convert the data files.

```
STARTUP NOMOUNT PFILE = '/tmp/init_mydb.ora';
RUN {
  CONVERT
  FROM PLATFORM 'Solaris Operating System (x86)'
  PARALLELISM 1
DATAFILE '/u01/oradata/SAMPL2/datafile/o1_mf_system_2k5c82h1_.dbf'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-SYSTEM_FNO-1_0ji58vro'
DATAFILE '/u01/oradata/SAMPL2/datafile/o1_mf_undotbs1_2k5c8xmc_.dbf'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-UNDOTBS1_FNO-
2_0ki58vro'
DATAFILE '/u01/oradata/SAMPL2/datafile/o1_mf_sysaux_2k5c8zll_.dbf'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-SYSAUX_FNO-3_0li58vro'
DATAFILE '/u01/oradata/SAMPL2/datafile/o1_mf_example_2jm6wwvt_.dbf'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-EXAMPLE_FNO-
4_0mi58vro'
DATAFILE '/u01/oradata/SAMPL2/datafile/o1_mf_users_2jm6trkj_.dbf'
  FORMAT '/tmp/mydbdata_D-SAMPL2_I-4155901966_TS-USERS_FNO-5_0ni58vro'
; }
```

REFERENCES

1. Oracle Maximum Availability Architecture
   http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm

2. *Oracle Database Backup and Recovery Advanced User's Guide* (Part B14191)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14191

3. *Oracle Data Guard Concepts and Administration* (Part B14239)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14239

4. *Oracle Universal Installer and OPatch User's Guide* (Part B16227)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b16227

5. *Oracle Database PL/SQL Packages and Types Reference* (Part B14258)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14258

6. *Oracle Database Backup and Recovery Reference* (Part B14194)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14194

7. *Oracle Database High Availability Best Practices* (Part B25159)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b25159

8. *Oracle Data Guard Concepts and Administration* (Part B28294)
   http://otn.oracle.com/pls/db111/db111.to_toc?partno=b28294

9. *Oracle OLAP Reference* (Part B14350)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14350

10. *Oracle OLAP DML Reference* (Part B28126)
    http://otn.oracle.com/pls/db111/db111.to_toc?partno=b28126

11. *Oracle Database Backup and Recovery User's Guide* (Part B28270)
    http://otn.oracle.com/pls/db111/db111.to_toc?partno=b28270

Platform Migration Using Transportable Database: Oracle Database 11*g* and 10*g* Release 2
August 2008
Author: Douglas Utzig

Oracle USA, Inc.
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com