**Performing a 32 bit to 64 bit migration using the Transportable Database RMAN feature**

This note describes the procedure used to perform a 32 bit to 64 bit conversion of an 11.2.0.3 database on the Linux platform.

The RMAN CONVERT DATABASE command is used to automate the movement of an entire database from one platform (the source platform) to another (the destination platform).

This is provided that the source and destination platforms are of the same endian format.

For example between Linux X86 32 bit and Linux X86 64 bit.

Note the following:

* Certain types of blocks, such as blocks in undo segments, need to be reformatted to ensure compatibility with the destination platform.
* Redo log files and control files from the source database are not transported. New control files and redo log files are created for the new database during the transport process, and an OPEN RESETLOGS is performed once the new database is created
* BFILEs are not transported. RMAN provides a list of objects using the BFILE datatype in the output for the CONVERT DATABASE command, but users must copy the BFILEs themselves and fix their locations on the destination database.
* Tempfiles belonging to locally managed temporary tablespaces are not transported. The temporary tablespace will be re-created on the target platform when the transport script is run.
* External tables and directories are not transported
* Password files are not transported. If a password file was used with the source database, the output of CONVERT DATABASE includes a list of all usernames and their associated privileges. Create a new password file on the destination database using this information

**Pre-Migration Checks**

**Check if transportable database can be used**

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

SQL> select platform\_name from v$db\_transportable\_platform where platform\_name like ‘Linux%’;

PLATFORM\_NAME

--------------------------------------------------------------------------------

Linux IA (32-bit)

Linux IA (64-bit)

Linux x86 64-bit

**Check for external tables and directories**

SQL> set serveroutput on

SQL> declare x boolean;

begin x := dbms\_tdb.check\_external; end; 2

3 /

The following directories exist in the database:

SYS.APPS\_DATA\_FILE\_DIR, SYS.ECX\_UTL\_LOG\_DIR\_OBJ, SYS.ECX\_UTL\_XSLT\_DIR\_OBJ,

SYS.LYCO\_AP\_REMIT, SYS.DATA\_PUMP\_DIR, SYS.ORACLE\_OCM\_CONFIG\_DIR

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\_name,directory\_path from dba\_directories;

DIRECTORY\_NAME

------------------------------

DIRECTORY\_PATH

--------------------------------------------------------------------------------

DATA\_PUMP\_DIR

/oracle/DEV/admin/DEV/dpdump/

ORACLE\_OCM\_CONFIG\_DIR

/oracle/DEV/devdb/11.2.0.3/ccr/state

LYCO\_AP\_REMIT

/home/filxfer/PROD/ENG/AP\_REM

DIRECTORY\_NAME

------------------------------

DIRECTORY\_PATH

--------------------------------------------------------------------------------

ECX\_UTL\_XSLT\_DIR\_OBJ

/usr/tmp

APPS\_DATA\_FILE\_DIR

/oracle/DEV/devdb/11.2.0.3/appsutil/outbound/DEV\_oradevint

ECX\_UTL\_LOG\_DIR\_OBJ

/usr/tmp

6 rows selected.

**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.**

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;

no rows selected

**Check for BFILES**

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;

/

The following directories contain external files for BFILE columns

Copy the files within these directories to the same path on the target system

There are 0 directories, 0 total BFILEs

**Export OLAP Analytic Workspaces**

Create a directory to save OLAP analytical workspace migration files.

mkdir /oracle/stage/aw\_migrate

**As Sys:**

Create a directory to save OLAP analytical workspace migration files.

SQL> create directory aw\_migrate as '/oracle/stage/aw\_migrate';

Directory created.

Obtain a list of OLAP analytical workspaces to be migrated.

SQL> col owner format a15

col aw\_name format a15

select OWNER, AW\_NAME ,PAGESPACES from dba\_aws

  where owner != 'SYS' order by 1,2;

SQL>

OWNER           AW\_NAME         PAGESPACES

--------------- --------------- ----------

FPA             FPAPJP                 308

ZPB             ZPBANNOT                 7

ZPB             ZPBCODE                292

ZPB             ZPBDATA                  7

For each OLAP analytical workspace from the above query (as per the table below), run the following package procedures to export the workspace.

exec dbms\_aw.execute( 'aw attach <owner>.<aw\_name> rw' );  
exec dbms\_aw.execute( 'allstat' );  
exec dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/<schema>\_<aw\_name>.eif ''' );  
exec dbms\_aw.execute( 'aw detach <owner>.<aw\_name>' );

Verify that each export file has been successfully created and copy them all across to the same location on the 64-bit server.

Delete each of the OLAP analytical workspaces.

exec dbms\_aw.execute( 'aw delete <owner>.<aw\_name>' );

As Sys:

Execute scripts to remove OLAP from the database.

@?/olap/admin/catnoamd.sql  
@?/olap/admin/olapidrp.plb  
@?/olap/admin/catnoaps.sql  
@?/olap/admin/catnoxoq.sql  
@?/rdbms/admin/utlrp.sql

Manually remove additional remaining OLAP database objects.

drop procedure sys.xoq\_validate;  
drop view sys.olap\_oledb\_reg\_attrs\_pvt;  
drop package body sys.cwm2\_olap\_installer;

set linesize 120 pagesize 0 feedback off echo off

spool /oracle/stage/aw\_migrate/drop\_olap\_syns.sql

select 'drop public synonym ' || object\_name || ';'  
  from dba\_objects  
  where owner = 'PUBLIC'  
  and object\_type = 'SYNONYM'  
  and status <> 'VALID'  
  and ( object\_name like '%OLAP%' or object\_name like '%AW' )  
  and object\_name not like '%RAW%';

spool off

run script /oracle/stage/aw\_migrate/drop\_olap\_syns.sql

SQL> exec dbms\_aw.execute( 'aw attach FPA.FPAPJP rw' );

SQL> exec dbms\_aw.execute( 'allstat' );

SQL> exec dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/FPA\_FPAPJP.eif ''' );

SQL> exec dbms\_aw.execute( 'aw detach FPA.FPAPJP' );

SQL> exec dbms\_aw.execute( 'aw attach ZPB.ZPBANNOT rw' );

SQL> exec dbms\_aw.execute( 'allstat' );

SQL> exec dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/ZPB\_ZPBANNOT.eif ''' );

SQL> exec dbms\_aw.execute( 'aw detach ZPB.ZPBANNOT' );

SQL> SQL> BEGIN dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/ZPB\_ZPBANNOT.eif ''' ); END;

\*

ERROR at line 1:

ORA-33390: There are no objects to export.

ORA-06512: at "SYS.DBMS\_AW", line 93

ORA-06512: at "SYS.DBMS\_AW", line 122

ORA-06512: at line 1

SQL> exec dbms\_aw.execute( 'aw attach ZPB.ZPBCODE rw' );

SQL> exec dbms\_aw.execute( 'allstat' );

SQL> exec dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/ZPB\_ZPBCODE.eif ''' );

SQL> exec dbms\_aw.execute( 'aw detach ZPB.ZPBCODE' );

SQL> exec dbms\_aw.execute( 'aw attach ZPB.ZPBDATA rw' );

SQL> exec dbms\_aw.execute( 'allstat' );

SQL> exec dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/ZPB\_ZPBDATA.eif ''' );

BEGIN dbms\_aw.execute( 'export all to eif file ''AW\_MIGRATE/ZPB\_ZPBDATA.eif ''' ); END;

\*

ERROR at line 1:

ORA-33390: There are no objects to export.

ORA-06512: at "SYS.DBMS\_AW", line 93

ORA-06512: at "SYS.DBMS\_AW", line 122

ORA-06512: at line 1

SQL> exec dbms\_aw.execute( 'aw detach ZPB.ZPBDATA' );

SQL>exec dbms\_aw.execute( 'aw delete FPA.FPAPJP' );

SQL> exec dbms\_aw.execute( 'aw delete ZPB.ZPBCODE' );

SQL> exec dbms\_aw.execute( 'aw delete ZPB.ZPBDATA' );

SQL> exec dbms\_aw.execute( 'aw delete ZPB.ZPBANNOT' );

Follow MetaLink Note 352306.1

**Prepare the database for Transportable Database**

**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.

Shut down and 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;

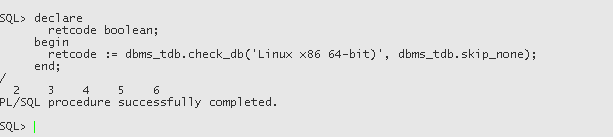
SQL> startup mount;

SQL> alter database open read only;

**Run the DBMS\_TDB.CHECK\_DB to Check Database State**

Checks to see if:

* Unrecognized target platform name
* Target platform has a different endian format.
* Database is not open read-only
* There are active or in-doubt transactions in the database.
* Database compatibility version is below 10



**CONVERT DATABASE**

In this case we are converting the datafiles on the destination source as opposed to converting them on the source host.

The source database (32 bit) data files are stored in a file system while the destination database (64 bit) will be using ASM disk groups instead.

Performing the conversion on the destination system will also avoid any performance overhead on the source system while the conversion process is underway.

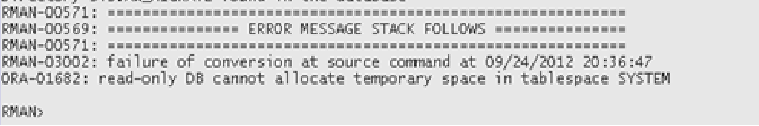
We use the **CONVERT DATABASE ON TARGET PLATFORM** RMAN command.

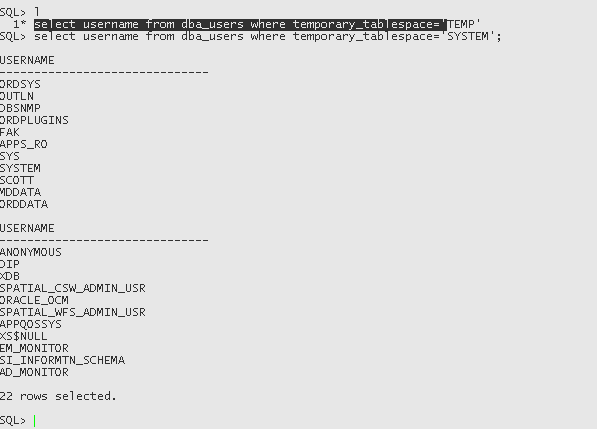
This command will generate a convert script which contains the CONVERT DATAFILE commands to perform the conversion from the source to target format

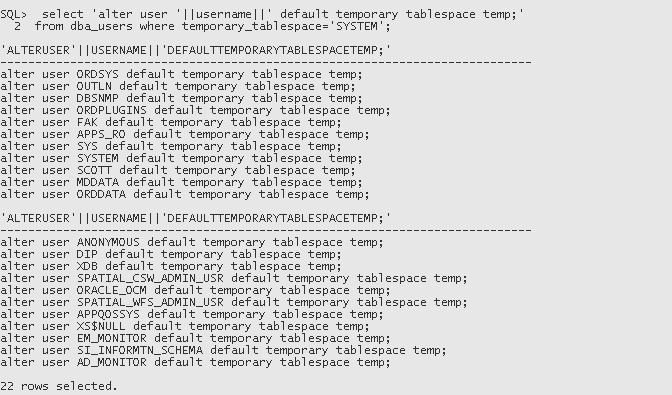
The transport script contains the actual commands used to recreate the control files on the target database after the data file conversion is completed.

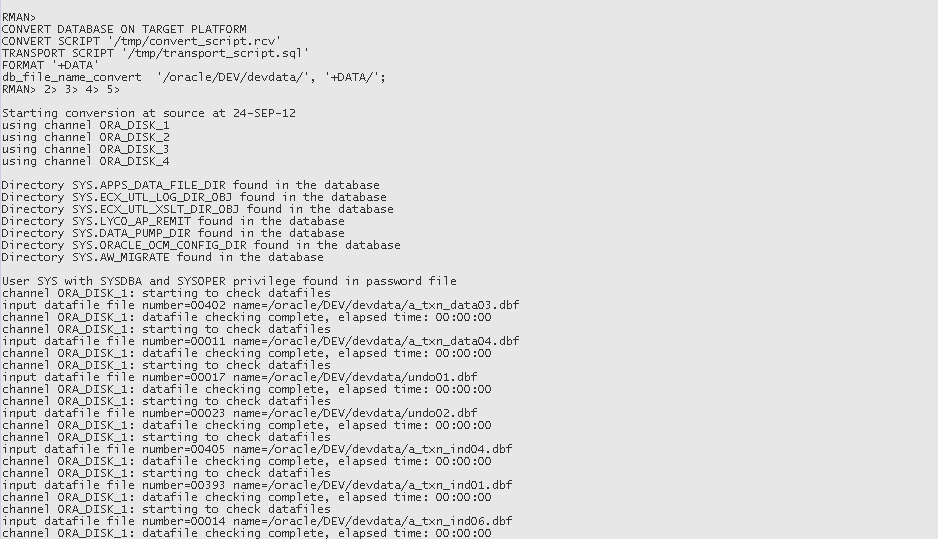


In our case we encountered an error because a number of users had been allocated the SYSTEM tablespace as their default tablespace.





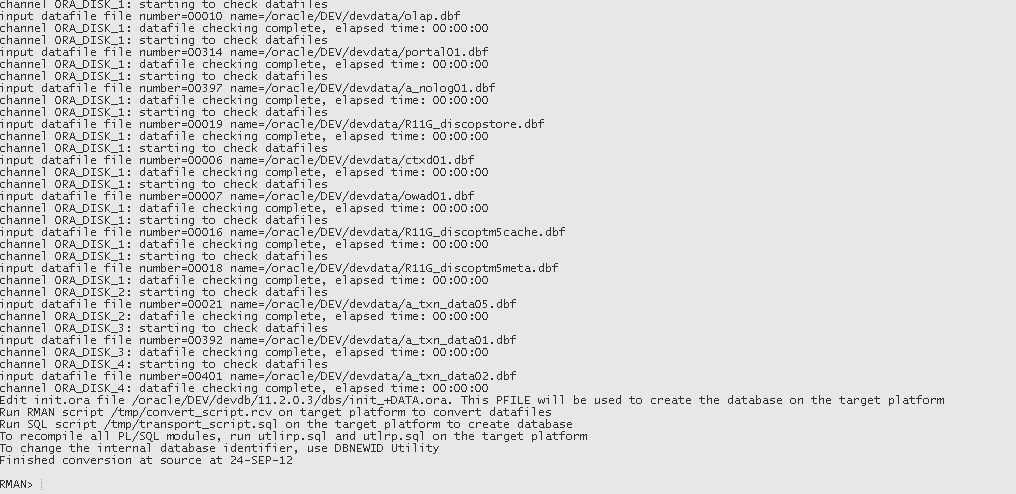




…

…..

………



Copy the /tmp/convert\_script.rcv and /tmp/transport\_script.sql files from source to appropriate destination on target machine.

Copy init.ora parameter file from source to target $ORACLE\_HOME/dbs

Make changes to init.ora

Change control\_files location to point to ASM disk group

Control\_files=’+DATA’,’+DATA’,’+DATA’

Also change location of adump in the init.ora datafile.

**Mount the source (read only) file system on target server**

Add an entry in /etc/exports

$ cat /etc/exports

/oracle oradevdb.mycorp.com.au (ro,sync)

Followed by

/etc/init.d/nfs restart

On target

mount -t nfs 192.168.xxx.xx:/oracle /oracle

**Run the CONVERT script**

convert\_script.rcv contents:

RUN {

CONVERT

FROM PLATFORM 'Linux IA (32-bit)'

PARALLELISM 4

DATAFILE '/oracle/DEV/devdata/undo02.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/undo01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/undo04.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/undo03.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system06.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system07.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system11.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system02.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system03.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system04.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system05.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system10.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system09.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/system08.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_data03.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_data04.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_ind04.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_ind01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_ind06.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_ind02.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_ind03.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_ind05.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/sysaux01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_archive01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_media01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_int01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_data06.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_ref01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_summ01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_ref02.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/lyco\_data01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/xxpic01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_data07.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_queue01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_queue02.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/discoverer01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/odm.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/olap.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/portal01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_nolog01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/R11G\_discopstore.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/ctxd01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/owad01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/R11G\_discoptm5cache.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/R11G\_discoptm5meta.dbf' FORMAT '+DATA'

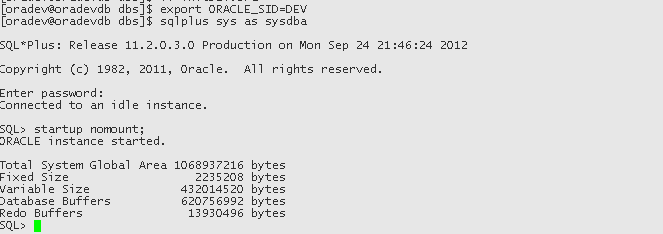
DATAFILE '/oracle/DEV/devdata/a\_txn\_data05.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_data01.dbf' FORMAT '+DATA'

DATAFILE '/oracle/DEV/devdata/a\_txn\_data02.dbf' FORMAT '+DATA'

; }

Next, start the database with NOMOUNT option and run the RMAN script convert\_script.rcv



RMAN**>@convert\_script.rcv**



Note – what I found that while it mentions the converted datafile as “+DATA/olap.dbf”, actually the file is created as “+DATA/dev/datafile/olap.277.794979681”

We had to take lines out of transport\_script.sql and edit to include the correct ASM OMF file name.

We then created a file called crectl.sql which had the following contents:

crecctl.sql

CREATE CONTROLFILE REUSE SET DATABASE "DEV" RESETLOGS FORCE LOGGING ARCHIVELOG

MAXLOGFILES 32

MAXLOGMEMBERS 5

MAXDATAFILES 512

MAXINSTANCES 8

MAXLOGHISTORY 7260

LOGFILE

GROUP 1 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 2 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 3 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 4 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 5 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 6 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 7 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 8 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 9 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512,

GROUP 10 (

'+DATA',

'+DATA'

) SIZE 100M BLOCKSIZE 512

DATAFILE

'+DATA/dev/datafile/apps\_ts\_tx\_data.355.794965469',

'+DATA/dev/datafile/apps\_ts\_tx\_data.361.794965469',

'+DATA/dev/datafile/apps\_ts\_tx\_data.360.794965469',

'+DATA/dev/datafile/apps\_ts\_tx\_data.356.794965469',

'+DATA/dev/datafile/apps\_ts\_tx\_idx.351.794969911',

'+DATA/dev/datafile/apps\_undots1.353.794969911',

'+DATA/dev/datafile/apps\_undots1.352.794969911',

'+DATA/dev/datafile/apps\_ts\_tx\_data.354.794969895',

'+DATA/dev/datafile/apps\_ts\_tx\_idx.350.794973207',

'+DATA/dev/datafile/apps\_ts\_tx\_idx.348.794973643',

'+DATA/dev/datafile/apps\_ts\_tx\_idx.349.794973643',

'+DATA/dev/datafile/apps\_ts\_tx\_idx.347.794973797',

'+DATA/dev/datafile/sysaux.343.794976769',

'+DATA/dev/datafile/apps\_undots1.344.794976663',

'+DATA/dev/datafile/apps\_undots1.345.794976619',

'+DATA/dev/datafile/system.342.794977855',

'+DATA/dev/datafile/apps\_ts\_tx\_data.337.794978951',

'+DATA/dev/datafile/apps\_ts\_seed.336.794979017',

'+DATA/dev/datafile/apps\_ts\_summary.335.794979051',

'+DATA/dev/datafile/apps\_ts\_seed.334.794979059',

'+DATA/dev/datafile/lyco.316.794979229',

'+DATA/dev/datafile/system.320.794979215',

'+DATA/dev/datafile/xxpic.315.794979255',

'+DATA/dev/datafile/apps\_ts\_tx\_data.319.794979261',

'+DATA/dev/datafile/system.318.794979377',

'+DATA/dev/datafile/system.317.794979377',

'+DATA/dev/datafile/system.307.794979377',

'+DATA/dev/datafile/system.268.794979377',

'+DATA/dev/datafile/system.271.794979523',

'+DATA/dev/datafile/system.267.794979523',

'+DATA/dev/datafile/system.269.794979523',

'+DATA/dev/datafile/apps\_ts\_queues.270.794979523',

'+DATA/dev/datafile/odm.276.794979665',

'+DATA/dev/datafile/olap.277.794979681',

'+DATA/dev/datafile/portal.278.794979695',

'+DATA/dev/datafile/discoverer.275.794979665',

'+DATA/dev/datafile/apps\_ts\_queues.273.794979665',

'+DATA/dev/datafile/r11g\_disco\_pstore.281.794979715',

'+DATA/dev/datafile/apps\_ts\_nologging.279.794979711',

'+DATA/dev/datafile/owapub.283.794979731',

'+DATA/dev/datafile/r11g\_disco\_ptm5\_cache.284.794979731',

'+DATA/dev/datafile/ctxd.282.794979729',

'+DATA/dev/datafile/r11g\_disco\_ptm5\_meta.285.794979731',

'+DATA/dev/datafile/system.272.794979657'

CHARACTER SET US7ASCII

;

Now startup nomount the database and run the script crectl.sql which will create the control files in the ASM disk group +DATA

After the control files have been created we open the database with RESETLOGS option.

SQL> ALTER DATABASE OPEN RESETLOGS;

The generated transport\_script.sql has the commands to add the tempfiles to the database and then run utlirp.sql.

But we found that when we tried to add the tempfiles, we were getting an error as shown below.

SQL> ALTER TABLESPACE TEMP ADD TEMPFILE '+DATA'

     SIZE 9000M AUTOEXTEND ON NEXT 8192  MAXSIZE 15000M;  2

ALTER TABLESPACE TEMP ADD TEMPFILE '+DATA'

\*

ERROR at line 1:

ORA-00604: error occurred at recursive SQL level 1

ORA-06553: PLS-801: internal error [56327]

Next shutdown the database using the IMMEDIATE option.

Change the location for CONTROL\_FILES from +DATA to actual OMF file name

\*.control\_files='+DATA/DEV/CONTROLFILE/current.323.795010677','+DATA/dev/controlfile/current.322.795010679','+DATA/dev/controlfile/current.321.795010679'

Then start the database using the UPGRADE option

SQL> startup upgrade pfile=' /u01/DEV/db/tech\_st/11.2.0.3/dbs/initDEV.ora’

Run the utlirp.sql script

@ ?/rdbms/admin/utlirp.sql

Add the tempfiles to the database

ALTER TABLESPACE TEMP ADD TEMPFILE '+DATA'

SIZE 9000M AUTOEXTEND ON NEXT 8192 MAXSIZE 15000M;

ALTER TABLESPACE TEMP ADD TEMPFILE '+DATA'

SIZE 2000M AUTOEXTEND ON NEXT 8192 MAXSIZE 15000M;

ALTER TABLESPACE TEMP ADD TEMPFILE '+DATA'

SIZE 32000M AUTOEXTEND ON NEXT 8192 MAXSIZE 32000M;

ALTER TABLESPACE R11G\_IAS\_TEMP ADD TEMPFILE '+DATA'

SIZE 104857600 AUTOEXTEND ON NEXT 8192 MAXSIZE 1048576000 ;

Shutdown the database

SQL> shutdown immediate

Next, start the database and run utlrp.sql to recompile all PL/SQL modules

SQL> startup pfile=' /u01/DEV/db/tech\_st/11.2.0.3/dbs/initDEV.ora’

SQL>@ ?/rdbms/admin/utlrp.sql

***Note – recompilation took close to 3 hours in this case***

**Add OLAP back into the database.**

@?/olap/admin/olap.sql SYSAUX TEMP

**Import OLAP Analytic Workspaces**

For each OLAP analytical workspace , run the following package procedures to import the workspace.

exec dbms\_aw.execute( 'aw create <owner>.<aw\_name>' );  
exec dbms\_aw.execute( 'import all from eif file ''AW\_MIGRATE/<schema>\_<aw\_name>.eif'' data dfns' );  
exec dbms\_aw.execute( 'update' );  
commit;  
exec dbms\_aw.execute( 'aw detach <owner>.<aw\_name>' );

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

Drop the directory used to import the OLAP analytical workspace migration files.

SQL> drop directory aw\_migrate;

Recompile invalid objects.

SQL> @?/rdbms/admin/utlrp.sql

SQL> select OWNER, AW\_NAME ,PAGESPACES from dba\_aws

where owner != 'SYS' order by 1,2;

OWNER AW\_NAME PAGESPACES

------------------------------ ------------------------------ ----------

FPA FPAPJP 532

ZPB ZPBANNOT 8

ZPB ZPBCODE 480

ZPB ZPBDATA 8

Note – further details can be found in MetaLink Note 352306.1