Reduce Your Disk Footprint by Sharing Read-Only Tablespaces

Iordan K. Iotzov

Senior Database Administrator
News America Marketing (NewsCorp)

iiotzov@newsamerica.com
Blog: http://iiotzov.wordpress.com/

About me

- 10+ years of database administration and development experience
- MS in Computer Science, BS in Electrical Engineering
- Presented at Hotsos, NYOUG and Virta-Thon
- Active blogger and OTN participant
- Senior DBA at News America Marketing (NewsCorp)

Check out our new SmartSource Xpress for iPad® app!

Check out SmartSource Xpress, our new iPad app!

Follow us on Twitter | Like us on Facebook

Agenda

- Overview of methods for sharing data among databases
- Transportable tablespaces method for sharing RO tablespaces
- Universal method /non-supported/ for sharing RO tablespaces
- Implementation strategies and tips

Overview

Ways to save space

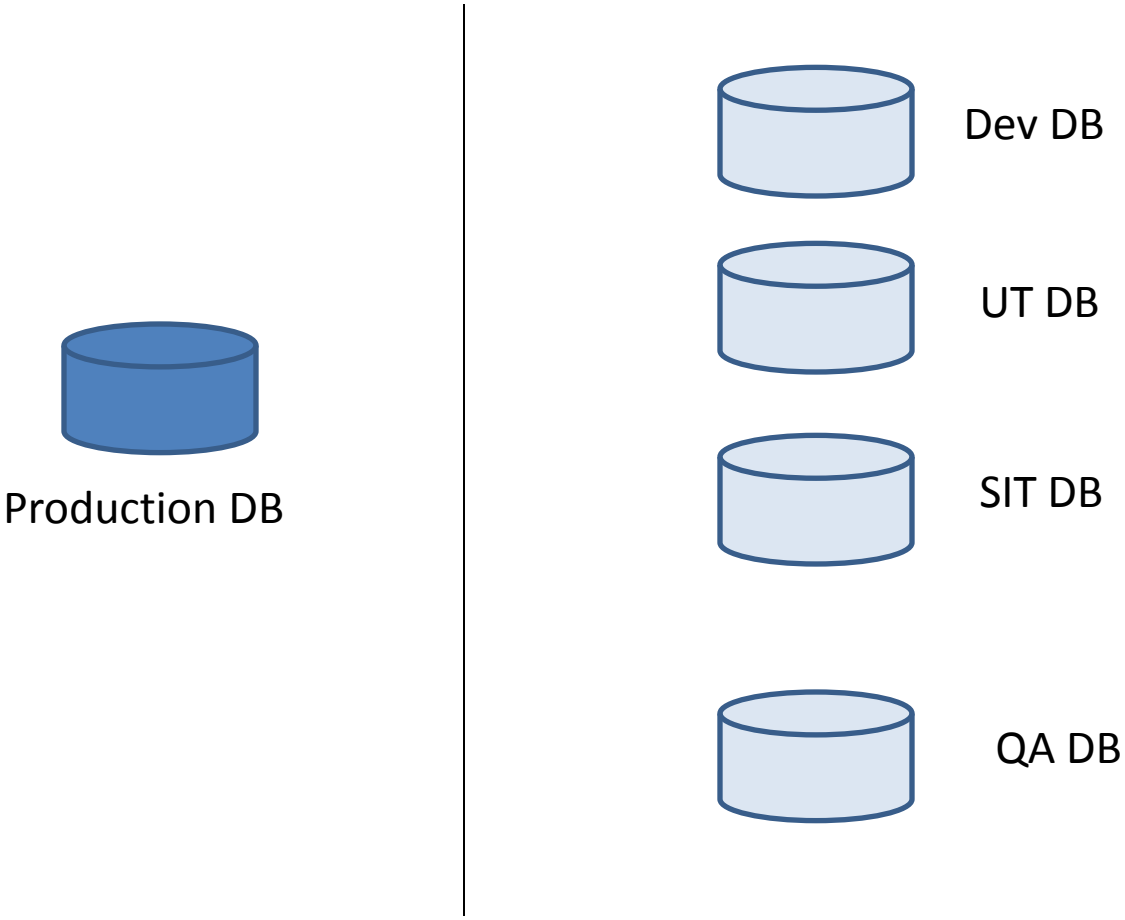
Strategies for reducing disk footprint for Oracle DB systems

- Compression
- Sensible database design
- Strict enforcement of retention policies
- Sharing data among multiple DBs
  - Database virtualization (Delphix®)
  - CloneDB, Direct NFS (11gR2)
  - Sharing RO tablespaces

Overview

The long tail

Where does most of the disk space go?



Overview

Challenge your assumptions

A simple word problem:

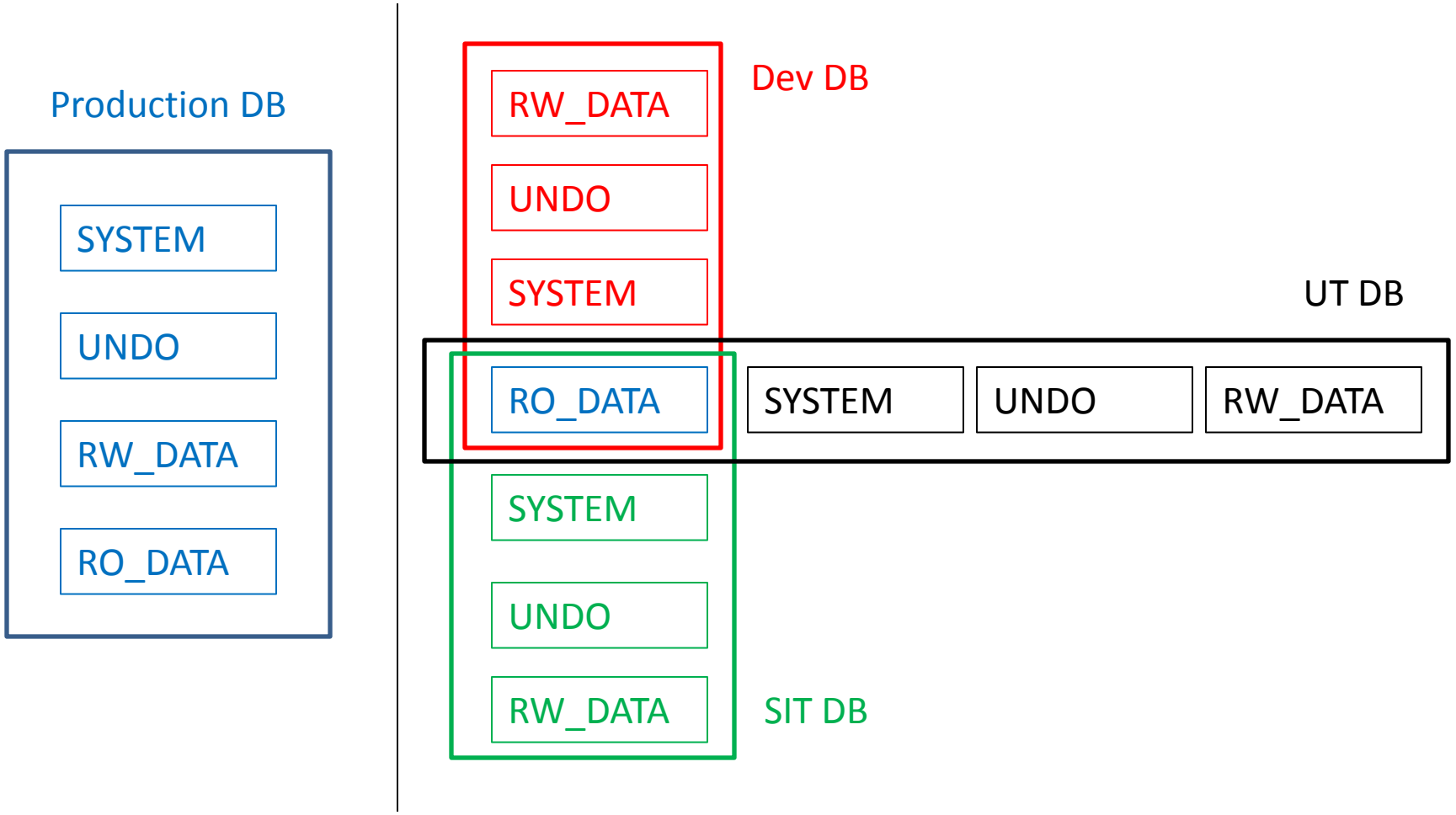
Sara administers a 1T production database. For the purposes of development, testing, and troubleshooting , she is asked to deliver eight (8) full non-production copies of that database. How much disk space does she needs for the eight non-production databases?

a) 8T

b) It depends

Overview

The big picture



Overview

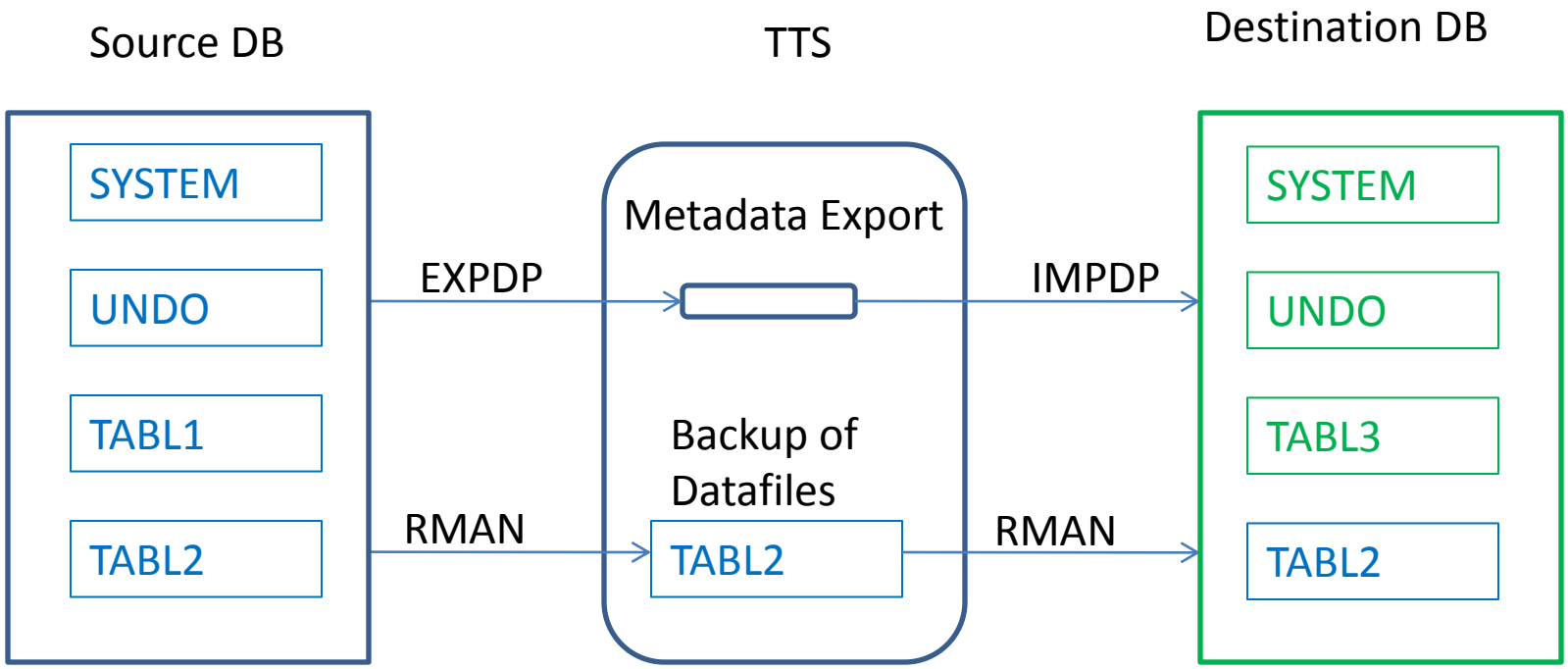
Would this approach get practical results?

In a typical database system, is there enough read-only data to get us a meaningful benefit from sharing read-only tablespaces?

- Very few entities are read only, however
- For many temporal entities, once inserted, the data is immutable
  - Transactions
  - Audit and log entries

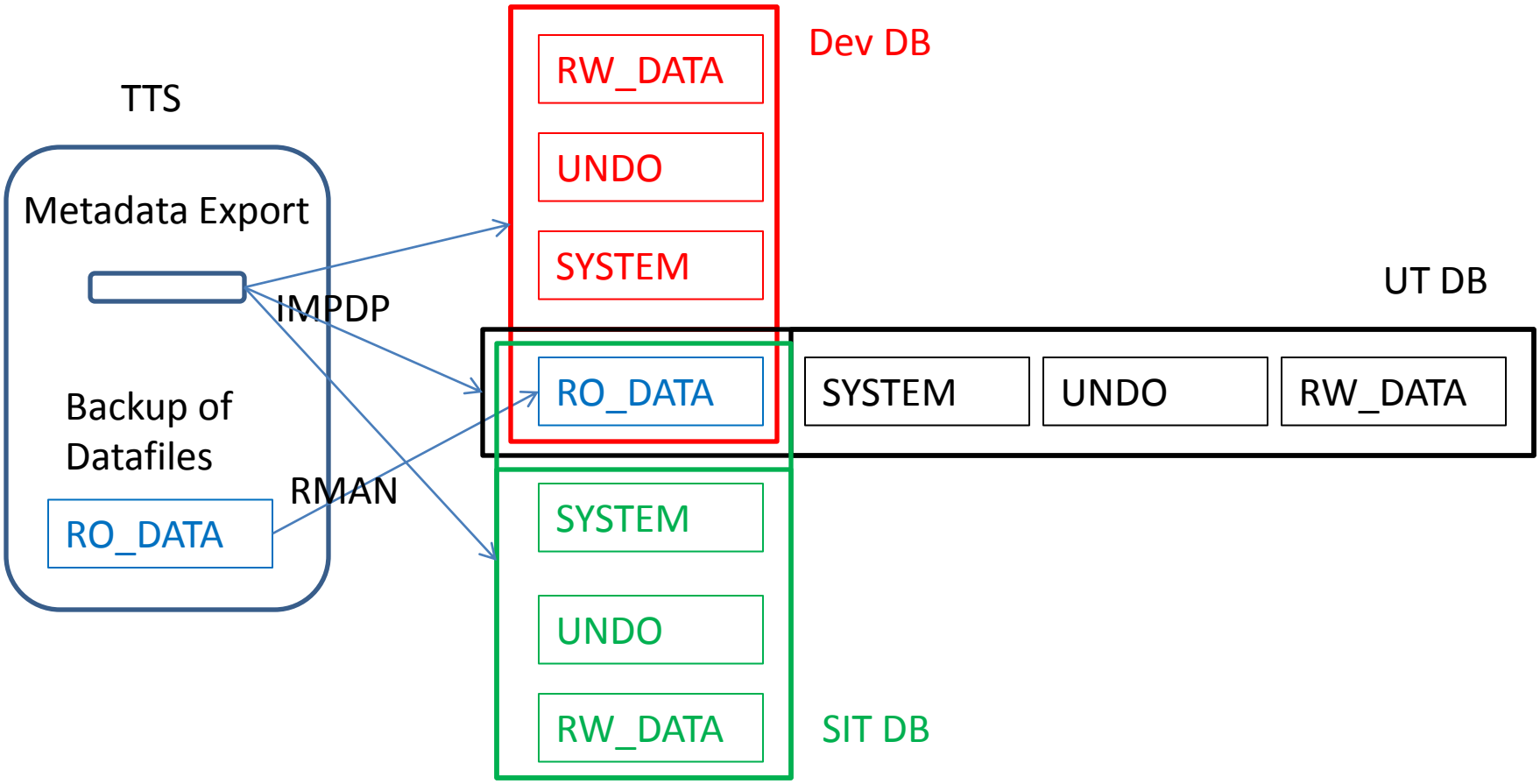
Sharing RO Tablespaces with TTS

Overview of transportable tablespaces



Sharing RO Tablespaces with TTS

Sharing in action



Sharing RO Tablespaces with TTS

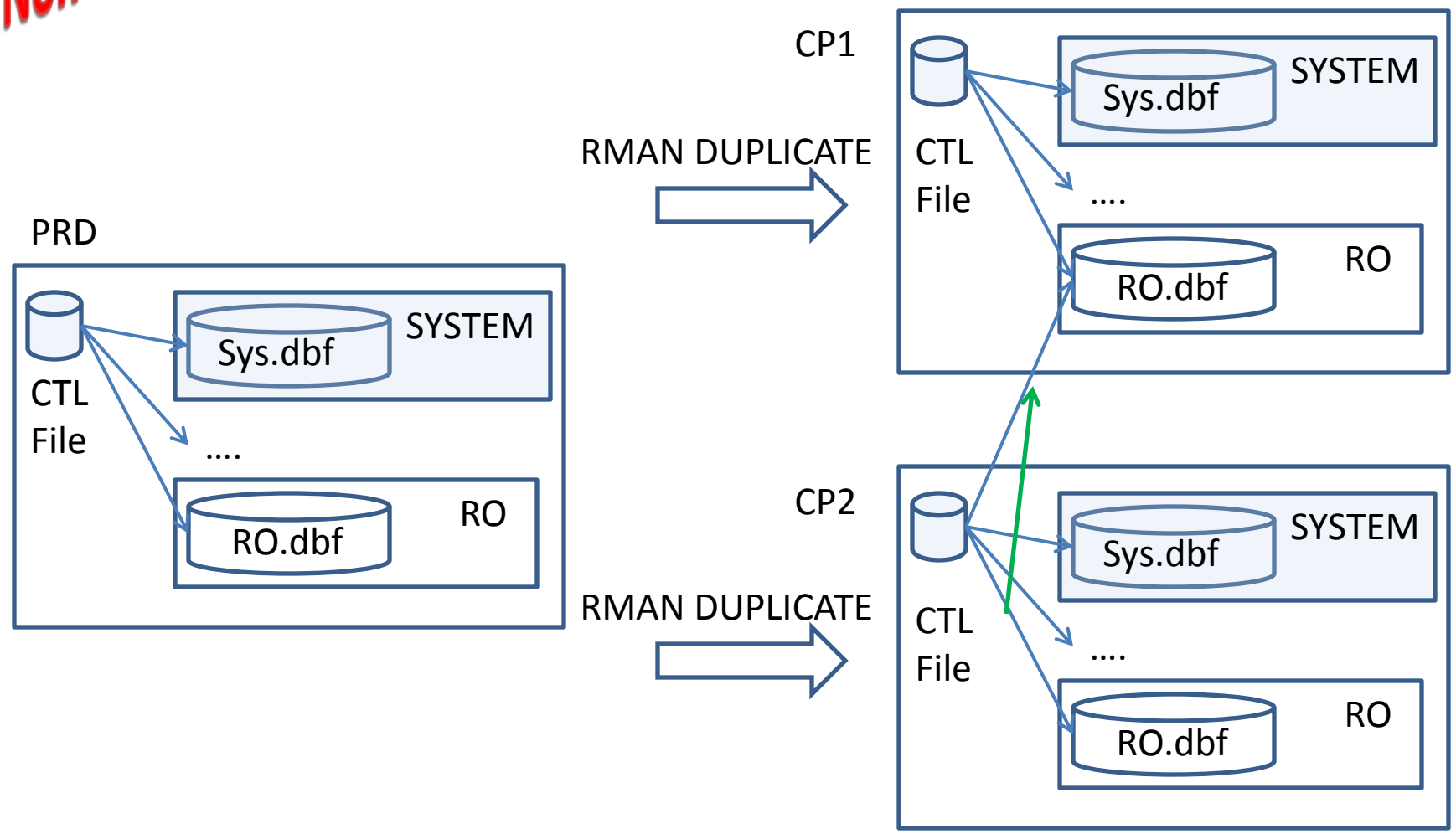
The challenge

- The tablespaces in a TTS set must be self-contained:
  - Objects with underlying objects (such as materialized views) or contained objects (such as partitioned tables) are not transportable unless all of the underlying or contained objects are in the tablespace set.

Universal Method for Sharing RO Tablespaces

How does it work?

Non-Supported



Universal Method for Sharing RO Tablespaces

Setup scripts RO tablespace

Non-Supported

```
PRD:
SQL> connect tst_user

SQL> create table ro_tab ( txt varchar2(100)) tablespace ro ;

SQL> insert into  ro_tab values ('resides in RO tablespace');

SQL> connect system

SQL> alter tablespace ro read only ;

SQL> connect tst_user

SQL> delete ro_tab ;
delete ro_tab
       *
ERROR at line 1:
ORA-00372: file 6 cannot be modified at this time
ORA-01110: data file 6: '+DATA/prd/datafile/ro.8187.792687981'
```

Universal Method for Sharing RO Tablespaces

Creation of non-production copies

Non-Supported

PRD

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/prd/datafile/system.4998.792687427
2 +DATA/prd/datafile/sysaux.5092.792687427
3 +DATA/prd/datafile/undotbs1.5289.792687427
4 +DATA/prd/datafile/users.5345.792687427
5 +DATA/prd/datafile/rw.8204.792687947
6 +DATA/prd/datafile/ro.8187.792687981
```

RMAN DUPLICATE

CP1

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/cp1/datafile/system.7993.792762069
2 +DATA/cp1/datafile/sysaux.8369.792762069
3 +DATA/cp1/datafile/undotbs1.8774.792762069
4 +DATA/cp1/datafile/users.9058.792762069
5 +DATA/cp1/datafile/rw.8715.792762069
6 +DATA/cp1/datafile/ro.9064.792762069
```

RMAN DUPLICATE

CP2

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/cp2/datafile/system.9530.792762689
2 +DATA/cp2/datafile/sysaux.9644.792762689
3 +DATA/cp2/datafile/undotbs1.9288.792762689
4 +DATA/cp2/datafile/users.10518.792762689
5 +DATA/cp2/datafile/rw.10462.792762689
6 +DATA/cp2/datafile/ro.9354.792762689
```

Universal Method for Sharing RO Tablespaces

Non-Supported

Datafile switch – from CP2 RO to CP1 RO

```
CP2:

SQL> alter tablespace ro offline ;

Tablespace altered.

SQL>  alter database rename file
'+DATA/cp2/datafile/ro.9354.792762689'
to '+DATA/cp1/datafile/ro.9064.792762069';

Database altered.

SQL>  alter tablespace ro online ;

Tablespace altered.

SQL> select * from tst_user.ro_tab ;

resides in RO tablespace
```

Universal Method for Sharing RO Tablespaces

Non-Supported

Datafiles after the switch

CP1

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/cp1/datafile/system.7993.792762069
2 +DATA/cp1/datafile/sysaux.8369.792762069
3 +DATA/cp1/datafile/undotbs1.8774.792762069
4 +DATA/cp1/datafile/users.9058.792762069
5 +DATA/cp1/datafile/rw.8715.792762069
6 +DATA/cp1/datafile/ro.9064.792762069
```

PRD

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/prd/datafile/system.4998.792687427
2 +DATA/prd/datafile/sysaux.5092.792687427
3 +DATA/prd/datafile/undotbs1.5289.792687427
4 +DATA/prd/datafile/users.5345.792687427
5 +DATA/prd/datafile/rw.8204.792687947
6 +DATA/prd/datafile/ro.8187.792687981
```

CP2

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/cp2/datafile/system.9530.792762689
2 +DATA/cp2/datafile/sysaux.9644.792762689
3 +DATA/cp2/datafile/undotbs1.9288.792762689
4 +DATA/cp2/datafile/users.10518.792762689
5 +DATA/cp2/datafile/rw.10462.792762689
6 +DATA/cp1/datafile/ro.9064.792762069
```

Universal Method for Sharing RO Tablespaces

Major issues after the switch?

Non-Supported

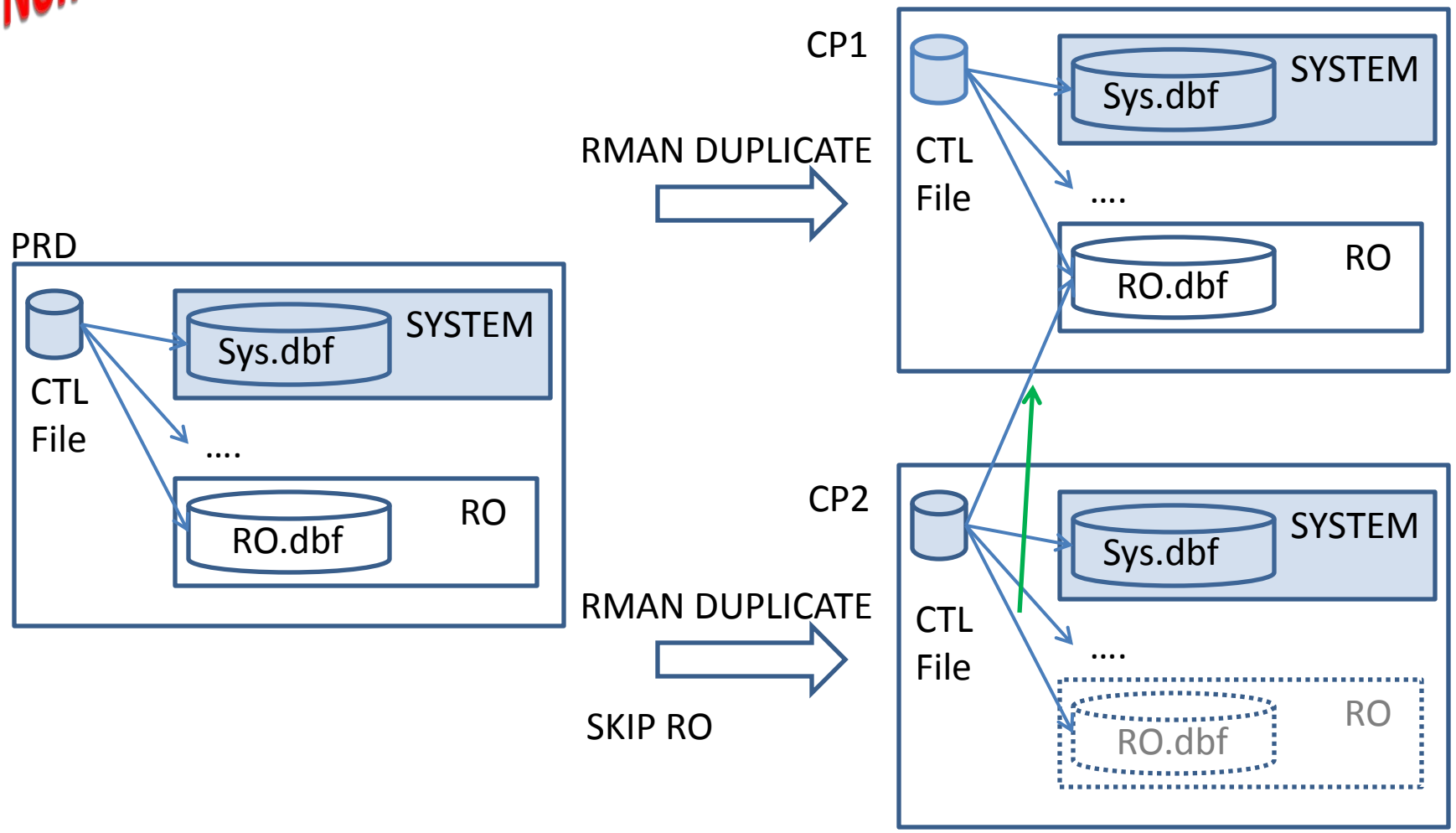
```
CP1:
SQL> shutdown immediate ;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> startup ;
ORACLE instance started.
Database mounted.
Database opened.
SQL> select * from tst_user.ro_tab;

resides in RO tablespace




SQL> select * from tst_user.ro_tab;

resides in RO tablespace
```

```
CP2:
SQL> shutdown immediate ;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> startup mount ;
ORACLE instance started.
Database mounted.
SQL> recover database until cancel ;
Media recovery complete.
SQL> alter database open resetlogs ;

Database altered.

SQL> select * from tst_user.ro_tab ;

resides in RO tablespace
```

Universal Method for Sharing RO Tablespaces

Making it more efficient

Non-Supported



Universal Method for Sharing RO Tablespaces

Creation of non-production copies – the efficient approach

Non-Supported

CP1

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/cp1/datafile/system.11298.793034355
2 +DATA/cp1/datafile/sysaux.10270.793034355
3 +DATA/cp1/datafile/undotbs1.11091.793034355
4 +DATA/cp1/datafile/users.11972.793034355
5 +DATA/cp1/datafile/rw.11601.793034355
6 +DATA/cp1/datafile/ro.10462.793034355
```

RMAN DUPLICATE

PRD

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/prd/datafile/system.4998.792687427
2 +DATA/prd/datafile/sysaux.5092.792687427
3 +DATA/prd/datafile/undotbs1.5289.792687427
4 +DATA/prd/datafile/users.5345.792687427
5 +DATA/prd/datafile/rw.8204.792687947
6 +DATA/prd/datafile/ro.8187.792687981
```

CP2

```
SQL> select  FILE# , NAME from v$datafile ;
1 +DATA/dbacp2/datafile/system.374.793035061
2 +DATA/dbacp2/datafile/sysaux.375.793035061
3 +DATA/dbacp2/datafile/undotbs1.376.793035061
4 +DATA/dbacp2/datafile/users.10026.793035061
5 +DATA/dbacp2/datafile/rw.10808.793035061
6 /u01/app/oracle/product/11.2.0/dbhome_1
/dbs/MISSING00006
```

RMAN DUPLICATE SKIP RO

Universal Method for Sharing RO Tablespaces

Datafile switch – from CP2 RO (non-present) to CP1 RO

Non-Supported

```
CP2:

SQL> alter tablespace ro offline ;

Tablespace altered.

SQL> alter database rename file
'/u01/app/oracle/product/11.2.0/dbhome_1/dbs/MISSING00006'
to   '+DATA/cp1/datafile/ro.10462.793034355' ;

Database altered.

SQL> alter tablespace ro online ;

Tablespace altered.

SQL> select * from tst_user.ro_tab ;

resides in RO tablespace
```

Universal Method for Sharing RO Tablespaces

But what about the DBID?

Non-Supported

As per MOS”
Difference between DB name, DB global name, DBID and SID [ID 1277854.1]

What's a DBID?

DBID is an internal, uniquely generated number that differentiates databases.
Oracle creates this number automatically when you create the database.
It's used to identify the database a file belongs to, and
It's used in recovery operations to determine that a certain redo log/archived redo log actually belong to the database being recovered;
Hence changing DBID requires opening the database with "resetlogs" option,
and invalidates all previous archived logs.

My opinion:
Some of those restrictions do not apply to datafiles that are part of read-only tablespaces

Universal Method for Sharing RO Tablespaces

Non-Supported

Thoughts what about the DBID

- Sharing read-only tablespaces between databases using TTS is a documented and supported option. Since the shared data file was never modified and has only one DBID then the DBID stored in the shared data file is not used to identify which database it belongs to.

- Dumping the file headers indicates so
  ALTER SESSION SET EVENTS 'IMMEDIATE trace name file_hdrs level 10';
  As per MOS [ID 218105.1] Introduction to ORACLE Diagnostic EVENTS

PRD (1864694135)

ID	DBID
1	1864694135
2	1864694135
3	1864694135
4	1864694135
5	1864694135
6	1864694135

CP1 (1054141155)

ID	DBID
1	1054141155
2	1054141155
3	1054141155
4	1054141155
5	1054141155
6	1864694135

CP2 (1701148445)

ID	DBID
1	1701148445
2	1701148445
3	1701148445
4	1701148445
5	1701148445
6	1864694135

Implementation Details and Tips

General observations – how to set the RO tablespace back to RW

Problem:

- Putting the shared RO tablespace in READ-WRITE mode

Solution:

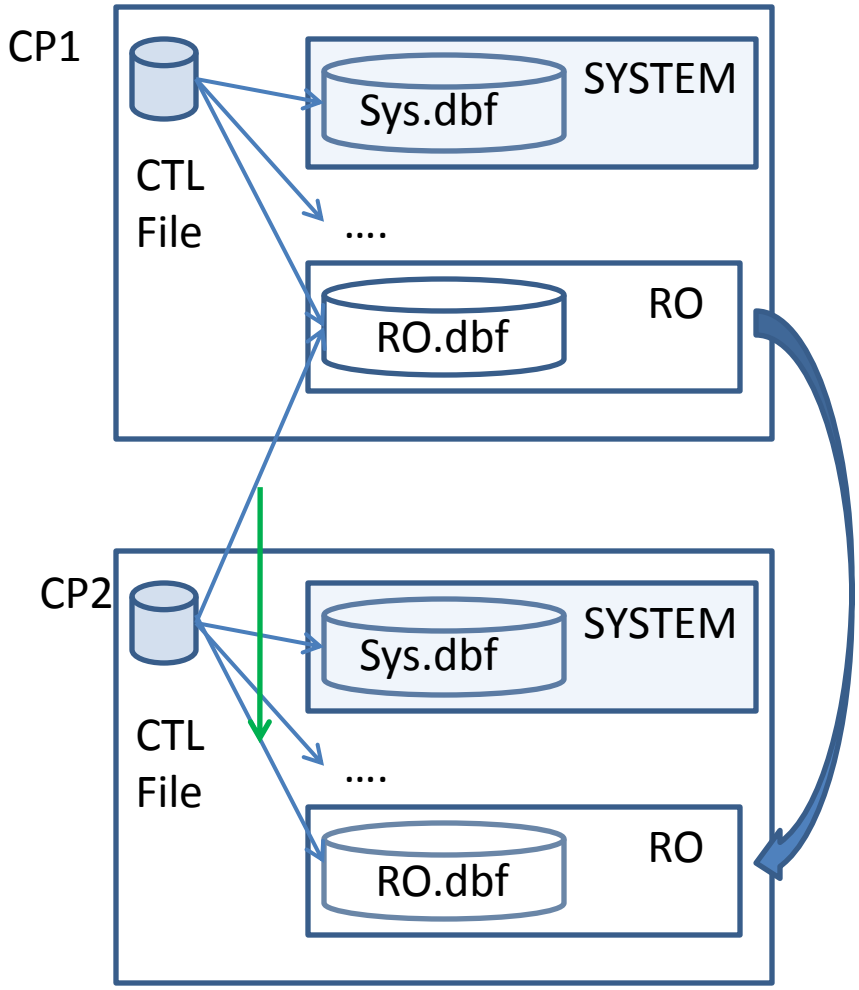
- Make a instance specific copy of the shared RO tablespace
- Re-point controlfile to the newly created instance specific copy
- Put the tablespace in Read-Write mode

N.B.

Never put a shared read-only tablespace in read-write mode!

Implementation Details and Tips

General observations – how to set the RO tablespace back to RW



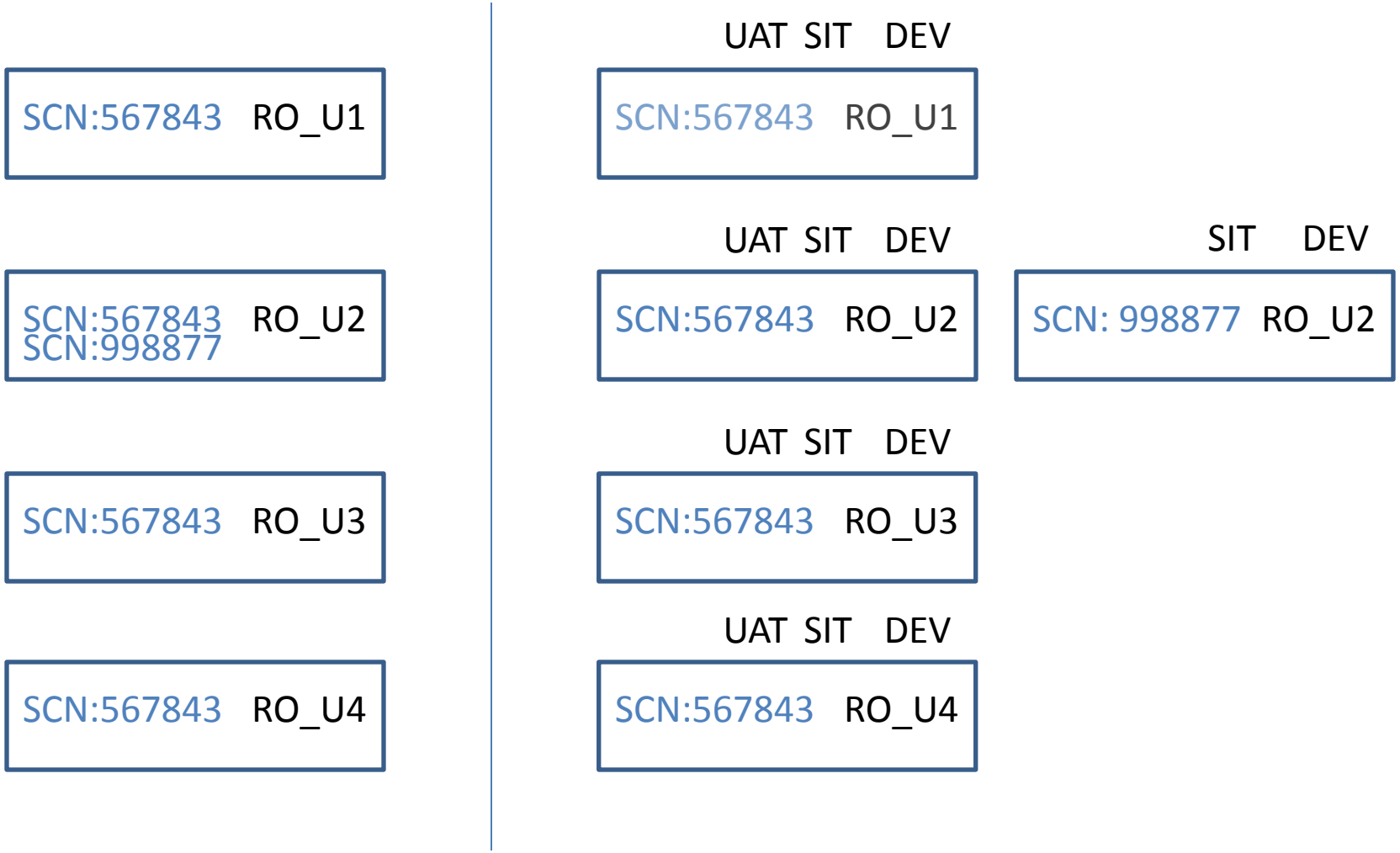
Implementation Details and Tips

General observations – basic rules

- Do not share read-only tablespaces that you intent to make read-write, even for a moment, at any time in the future
- If a shared read-only tablespace is to be temporarily converted to read-write, reduce the frequency of such conversions
- To minimize the impact of updating data originally slated as read-only, you can create multiple read-only tablespaces related to different unit. If data for a single unit it to be revised, only the tablespace for that specific unit would be affected.

Implementation Details and Tips

General observations – basic rules



Implementation Details and Tips

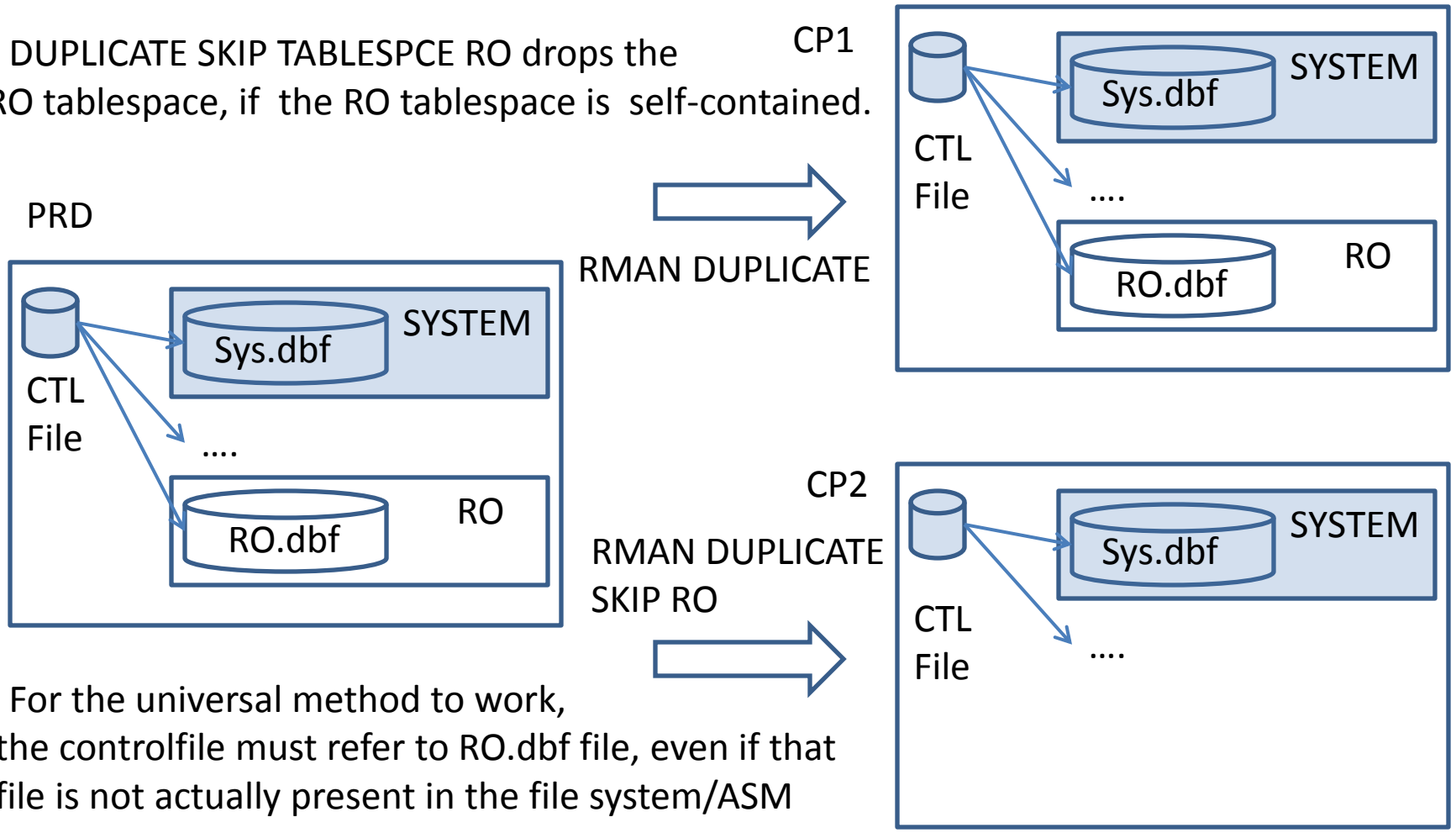
Universal method considerations - ways to implement

- Major implementation methods
  - RMAN DUPLICATE (preferred)
    - SKIP TABLESPACE
  - RMAN RESTORE/RECOVER
    - SKIP TABLESPACE
  - Manual
    - UNIX cp,
    - create controflie,
    - alter database resetlogs
    - etc

Implementation Details and Tips

Universal method considerations – preventing the removal of the RO tablespace

- DUPLICATE SKIP TABLESPCE RO drops the RO tablespace, if the RO tablespace is self-contained.



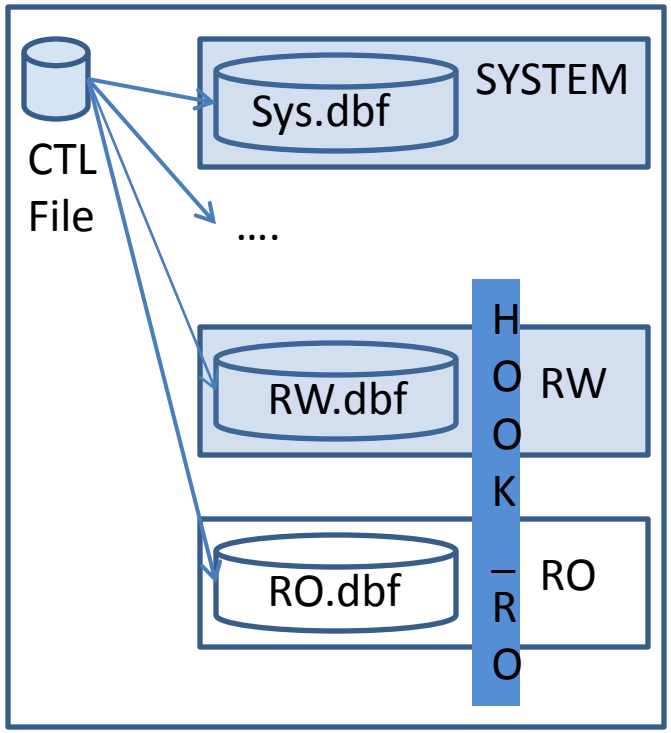
- For the universal method to work, the controlfile must refer to RO.dbf file, even if that file is not actually present in the file system/ASM

Implementation Details and Tips

Universal method considerations – preventing the removal of the RO tablespace

A simple way to guarantee that the RO tablespace is not dropped after DUPLCIATE is to make it not self-contained.
A dummy partitioned table would do the trick:

```
CREATE TABLE HOOK_RO
(
  DUMMY NUMBER
)
PARTITION BY RANGE (DUMMY)
(
  PARTITION PART_RW VALUES
  LESS THAN (1000)
  TABLESPACE "RW"
, PARTITION PART_RO VALUES
  LESS THAN (2000)
  TABLESPACE "RO"
NOCOMPRESS
);
SQL> insert into HOOK_RO values (500);
SQL> insert into HOOK_RO values (1500);
```



Implementation Details and Tips

Universal method considerations – 11gR2 DUPLICATE behavior

Problem:

Starting in 11gR2, Oracle performs self-contained check before starting DUPLICATE, preventing us from skipping non self-contained tablespaces

Solution/Workaround:

MOS Document “RMAN-05548 # skip self-contained check in rman duplicate [ID 1355120.1]”

Implementation Details and Tips

Universal method considerations – backup

- RMAN DUPLICATE generates new DBID

  no restrictions

- RMAN RESTORE and manual implementation keep the production DBID.

  Nid (generate new DBID)cannot be run

  Cannot use CATALOG when backing up and restoring

Implementation Details and Tips

Universal method considerations – restore

- RMAN RESTORE/RECOVER:

Need to exempt the RO tablespace when issuing RMAN RECOVER

If needed, change the controlfile to indicate the valid location of the datafile

```
RMAN> RUN
 {
 set until time "to_date(…')";
 recover database skip tablespace ro;
 }
SQL> alter tablespace ro offline;
alter database rename file '<CTL_FILE_LOCATION>' to
'<ACTUAL_LOCATION>'
SQL> alter tablespace ro online;
```

- RMAN DUPLCIATE
  - Connected to target – OK
  - Not connected - FAIL

Thank you