



Internals of the Oracle Database 12c Multitenant Architecture

UGF4133

Vit Spinka

Agenda

- Changes in data dictionary
- Changes in redo log
- A lot of things are no longer unique
- Redo, undo, datafiles
- Backup and recovery
- Q&A

Vit Spinka

- Working with Oracle Database since 8i
- Oracle Certified Master
- Principal developer of Dbvisit Replicate
- ... which gets its data by parsing Oracle redo logs

- @vitspinka
- vit.spinka@dbvisit.com
- This presentation for download at <http://vitspinka.cz/download.html>



Dbvisit



- HQ in New Zealand, US subsidiary, partners throughout the world
- Used in 110+ Countries
- Database Replication is our playground
- Worldwide leader in DR solutions for Oracle Standard Edition
- Product Engineers with “real world” DBA Experience
- Regular presenters at Oracle events such as OOW, Collaborate and NZOUG
- Passionate about Oracle Technology



Trusted in 110+ countries. . . By 1000+ companies

The image features a central world map with a blue grid pattern. Surrounding the map are logos for a wide variety of companies and organizations, demonstrating global reach. The logos include:

- North America:** CIANBRO, SHADOW PHYSICAL SYSTEMS, ALDO, NIC, Model N, J.B. HUNT, ROAMWARE, GRANTSOLUTIONS.GOV, fiserv., TELEDYNE OIL & GAS, ComplianceEase, Celgene, Boehringer Ingelheim, PAREXEL, First Citizens Bank, verizon, deckers OUTDOOR CORPORATION, ADVANTEX, SCHNEIDER, imation.
- Europe:** FX INNOVATION, G A L I L E O, Capgemini, Bell, Bell, LOGISTICS, univision, AEGON, MercuryGate, engaging NETWORKS, Loylogic, ORION, Government of the Republic of Trinidad and Tobago.
- Asia:** Mercedes-Benz, Solium CAPITAL, FloridaRealtors, vizrt, Pythian, AGFA HealthCare, IBM, TechHealth, AURUS, North Carolina Health IT, Alcatel-Lucent, CBS, FBE.
- Australia:** senozza, Sansio, Radio-Canada, amazon web services.
- Other:** Dbvisit THE SMART ALTERNATIVE, at&t, VW, CBC, AUSA.

What is multitenant, after all



- A kind of new naming schema
- Separate datafiles, separate schemas, separate logins
- But same instance

- When playing around with this, don't forget PDBs don't open on instance startup
- `SQL> alter session set container=UNITPDB1;`
- (12.1.0.2 adds `ALTER PLUGGABLE DATABASE ... SAVE STATE`)

CON_ID, CON_UID

```
SQL> select CON_ID, CON_UID, GUID, NAME from v$pdb;
```

CON_ID	CON_UID	GUID	NAME
2	4078450203	EEFF5A1B3E781A03E045000000000001	PDB\$SEED
3	2392015286	EEFFBB158E172CB0E045000000000001	UNITPDB1
4	3061124827	EEFFC1111B122D59E045000000000001	UNITPDB2

- CON_ID – one byte, unique in the database
 - 0 – non-CDB database
 - 1 – CDB level (visible in v\$containers)
 - 2 – PDB\$SEED
- CON_UID – (locally?) unique, similar as DBID (CON_UID of CDB *is* DBID)
- GUID – globally unique

Datafiles

F#	TS#	R#	NAME	CON_ID
1	0	1	.../DATAFILE/system.444.835805791	1
3	1	3	.../DATAFILE/sysaux.463.835805697	1
4	2	4	.../DATAFILE/undotbs1.382.835805887	1
6	4	6	.../DATAFILE/users.361.835805887	1
5	0	1	.../DD7C48AA5A4404A2E04325AAE80A403C/DATAFILE/system.424.835806017	2
7	1	4	.../DD7C48AA5A4404A2E04325AAE80A403C/DATAFILE/sysaux.372.835806017	2
8	0	1	.../EEFFBB158E172CB0E04500000000001/DATAFILE/system.462.835807737	3
9	1	4	.../EEFFBB158E172CB0E04500000000001/DATAFILE/sysaux.396.835807735	3

- RFILE# is relative file; dates back to TTS, now used per-PDB; TS#, too
- OMF uses GUID for PDB files
- Only CON_ID in the v\$datafile view (and virtually all others)

SYS.OBJ\$

- In non-CDB:
- SYS.OBJ\$ is table in SYSTEM (id = 18, in my db); not part of any cluster
- Contains info about all objects in database

```
create table obj$                                     /* object table */
( obj#          number not null,                      /* object number */
  /* DO NOT CREATE INDEX ON DATAOBJ# AS IT WILL BE UPDATED IN A SPACE
   * TRANSACTION DURING TRUNCATE */
  dataobj#      number,                               /* data layer object number */
  owner#        number not null,                     /* owner user number */
  name          varchar2("M_IDEN") not null,         /* object name */
  namespace     number not null,                     /* namespace of object (see KQD.H): */
/* 1 = TABLE/PROCEDURE/TYPER, 2 = BODY, 3 = TRIGGER, 4 = INDEX, 5 = CLUSTER, */
/* 6 = SYNONYM, 7 = SEQUENCE, 8 = LOB, 9 = DIRECTORY, */
/* 10 = QUEUE, 11 = REPLICATION OBJECT GROUP, 12 = REPLICATION PROPAGATOR, */
/* 13 = JAVA SOURCE, 14 = JAVA RESOURCE */
  subname       varchar2("M_IDEN"),                  /* subordinate to the name */
  type#         number not null,                     /* object type (see KQD.H): */
/* 1 = INDEX, 2 = TABLE, 3 = CLUSTER, 4 = VIEW, 5 = SYNONYM, 6 = SEQUENCE, */
/* 7 = PROCEDURE, 8 = FUNCTION, 9 = PACKAGE, 10 = NON-EXISTENT, */
/* 11 = PACKAGE BODY, 12 = TRIGGER, 13 = TYPE, 14 = TYPE BODY, */
```

DBA_OBJECTS

```
select u.name, o.name, o.subname, o.obj#, o.dataobj#,
       decode(o.type#, 0, 'NEXT OBJECT', 1, 'INDEX', 2, 'TABLE', ... 'UNDEFINED'),
       o.ctime, o.mtime,
       to_char(o.stime, 'YYYY-MM-DD:HH24:MI:SS'),
       decode(o.status, 0, 'N/A', 1, 'VALID', 'INVALID'),
       decode(bitand(o.flags, 2), 0, 'N', 2, 'Y', 'N'),
       o.namespace,
       decode(bitand(o.flags, 4194304), 4194304, 'Y', 'N')
from sys."_CURRENT_EDITION_OBJ" o, sys.user$ u
where o.owner# = u.user#
and o.type# != 10 /* NON-EXISTENT */
   and o.name != '_NEXT_OBJECT'
   and o.name != '_default_auditing_options_'
```

- `_CURRENT_EDITION_OBJ` is a view on `SYS.OBJ$`, handling Editions

Multitenant: CDB_OBJECTS

- The DDL for SYS.OBJ\$ and DBA_OBJECTS is the same
- There is new view: CDB_OBJECTS

12.1.0.1

```
SELECT "OWNER", "OBJECT_NAME", "SUBOBJECT_NAME", "OBJECT_ID",  
"DATA_OBJECT_ID", "OBJECT_TYPE", "CREATED", "LAST_DDL_TIME",  
"TIMESTAMP", "STATUS", "TEMPORARY", "GENERATED", "SECONDARY",  
"NAMESPACE", "EDITION_NAME", "SHARING", "EDITIONABLE",  
"ORACLE_MAINTAINED", "CON_ID" FROM CDB$VIEW("SYS"."DBA_OBJECTS")
```

- CDB\$VIEW: Internal, undocumented
- Gathers tables from all PDBs into X\$ table, then runs PX slaves (a bit similar to GV\$)

CDB\$VIEW

- But don't try that with your tables*:

```
SQL> select * from cdb$view("SCOTT"."EMP");  
select * from cdb$view("SCOTT"."EMP")  
*
```

ERROR at line 1:

ORA-03113: end-of-file on communication channel

Exception [type: SIGSEGV, Address not mapped to object] [ADDR:0xA] [PC:
0xB83E308, kglhdgn()+120] [flags: 0x0, count: 1]

```
kglhdgn()+120      signal    __sighandler()          7F5B34766C40 ? kglrdti()  
+238             call      kglhdgn()                7F5B34766C40 ? qcdlgbo()+5243  
call             kglrdti()                7F5B34766C40 ?
```

Multitenant: CDB_OBJECTS

- 12.1.0.2 introduces CONTAINERS clause

12.1.0.2

```
SELECT "OWNER", "OBJECT_NAME", "SUBOBJECT_NAME", "OBJECT_ID",  
"DATA_OBJECT_ID", "OBJECT_TYPE", "CREATED", "LAST_DDL_TIME",  
"TIMESTAMP", "STATUS", "TEMPORARY", "GENERATED", "SECONDARY",  
"NAMESPACE", "EDITION_NAME", "SHARING", "EDITIONABLE",  
"ORACLE_MAINTAINED", "CON_ID" FROM CONTAINERS ("SYS"."DBA_OBJECTS")
```

- Now documented and supported
- But beware: by default it excludes PDB\$SEED and this affects even views like CDB_DATA_FILES(!)

Links

- Metadata link – the table is in CDB as well as in PDBs
- PDB has pointers to records in CDB
- (you can check that the rows are in PDB, too, really – cf. rowids)

```
SQL> select rowid, name from sys.col$ where obj#=18 and name='OBJ#';
```

```
ROWID          NAME
-----
AAAAACAABAAAACSABE OBJ#
-----
AAAAACAABAAAACRAAT OBJ#
```

- Object link – the table in CDB only, but visible in PDBs, too
(and query with rowid crashes in PDB 😊)

```
select rowid, name from dba_hist_sga where ...
```

Metadata Links cont'ed

- PDB has pointers to records in CDB
- And you can update the different rows in PDBs

```
SQL> update sys.col$ set name='LENGTH_' where obj#=13 and col#=7;
```

...

```
SQL> select con_id, name from cdb$view(sys.col$) where obj#=13 and col#=7
```

```
CON_ID NAME
```

```
-----
```

```
1 LENGTH#
```

```
2 LENGTH
```

```
3 LENGTH_
```


SYS.OBJ\$

- So how many OBJ\$s are there, actually?

```
SQL> select con_id, owner, object_name, object_id, data_object_id, sharing
from cdb_objects where object_name='OBJ$'
```

CON_ID	OWN	OBJE	OBJECT_ID	DATA_OBJECT_ID	SHARING
3	SYS	OBJ\$	18	18	METADATA LINK
2	SYS	OBJ\$	18	18	METADATA LINK
1	SYS	OBJ\$	18	18	METADATA LINK
4	SYS	OBJ\$	18	18	METADATA LINK

User-created links

- User/DBA can create linked objects, too
<http://www.dbi-services.com/index.php/blog/entry/oracle-12c-cdb-metadata-a-object-links-internals> (Franck Pachot, dbi services)
- Undocumented, unsupported (`_ORACLE_SCRIPT=TRUE`)
- SHARING=NONE, METADATA, OBJECT
- COMMON_DATA
- Shows that some further magic is happening

What is redo, after all

- Basic goal: recover database
- Online redo: instance/crash recovery
- Archive redo: media recovery
- Every change is written to redo before data files
- Allows to replay changes
- Shows what actually Oracle stores on disk

Redo dump

- MOS Note 103181.6
- ALTER SYSTEM DUMP LOGFILE 'filename';
- The resulting trace file is huge and contains most (but not all!) interesting information from the redo in a text format
- We will see these dumps further in this session
- Database can dump redo from a different database, as long as the endian (big/little) match and version is the same or higher than the source db

Basic structure of redo

- Redo record
- Spans one or more redo blocks
- Each redo record has an RBA, pointing to start of the redo record
- Header – SCN, time, size; trace will lie and show these even if they are not in the header
- Change records – actual data

```
REDO RECORD - Thread:1 RBA: 0x000068.00000015.0010 LEN: 0x007c VLD: 0x05 CON_UID: 3345156736  
SCN: 0x0000.002d22c3 SUBSCN: 1 07/08/2014 15:58:58  
(LWN RBA: 0x000068.00000015.0010 LEN: 0002 NST: 0001 SCN: 0x0000.002d22c0)
```

REDO RECORD - Thread:1 RBA: 0x000068.00000015.0010 LEN: 0x007c VLD: 0x05
CON_UID: 3345156736

SCN: 0x0000.002d22c3 SUBSCN: 1 07/08/2014 15:58:58

(LWN RBA: 0x000068.00000015.0010 LEN: 0002 NST: 0001 SCN: 0x0000.002d22c0)

Basic structure of redo



- Change record
- Describes actual change
- Data changes: change to undo, change to data (tables, indexes)
- Transactions: changes to undo (allocate undo entry), commit/rollback
- Maintenance: space allocation, ASSM bitmaps...
- All is referenced by ids valid on the source: object id, data object id, datafile Absolute File Number, DBA, undo segment id...

Change record in detail



- Example: insert into hr.jobs values ('TE_TEST', 'Test job', 1, 100)
- Redo

```
CHANGE #1 CON_ID:3 TYP:0 CLS:1 AFN:13 DBA:0x034000bb OBJ:95424 SCN:0x0000.00285536 SEQ:2 OP:11.2 ENC:0 RBL:0
KTB Redo
op: 0x01 ver: 0x01
compat bit: 4 (post-11) padding: 1
op: F xid: 0x0002.013.000008d3 uba: 0x01003327.0187.04
KDO Op code: IRP row dependencies Disabled
  xtype: XA flags: 0x00000000 bdba: 0x034000bb hdba: 0x034000ba
itli: 1 ispac: 0 maxfr: 4858
tabn: 0 slot: 0(0x0) size/delt: 26
fb: --H-FL-- lb: 0x1 cc: 4
null: ----
col 0: [ 7] 54 45 5f 54 45 53 54
col 1: [ 8] 54 65 73 74 20 6a 6f 62
col 2: [ 2] c1 02
col 3: [ 2] c2 02
```


CHANGE #1 CON_ID:3 TYP:0 CLS:1 AFN:13 DBA:0x034000bb OBJ:95424 SCN:
0x0000.00285536 SEQ:2 OP:11.2 ENC:0 RBL:0

KTB Redo

op: 0x01 ver: 0x01

compat bit: 4 (post-11) padding: 1

op: F xid: 0x0002.013.000008d3 uba: 0x01003327.0187.04

KDO Op code: IRP row dependencies Disabled

xtype: XA flags: 0x00000000 bdba: 0x034000bb hdba: 0x034000ba

itli: 1 ispac: 0 maxfr: 4858

tabn: 0 slot: 0(0x0) size/delt: 26

fb: --H-FL-- lb: 0x1 cc: 4

null: ----

col 0: [7] 54 45 5f 54 45 53 54

col 1: [8] 54 65 73 74 20 6a 6f 62

col 2: [2] c1 02

col 3: [2] c2 02

Change record in detail



```
CHANGE #5 CON_ID:1 TYP:0 CLS:20 AFN:4 DBA:0x01003327 OBJ:4294967295 SCN:0x0000.002d228c SEQ:1 OP:5.1 ENC:0 RBL:0
ktudb redo: siz: 112 spc: 7776 flg: 0x0012 seq: 0x0187 rec: 0x04
          xid: 0x0002.013.000008d3
ktubl redo: slt: 19 rci: 0 opc: 11.1 [objn: 95424 objd: 95424 tsn: 3]
Undo type: Regular undo          Begin trans    Last buffer split: No
Temp Object: No
Tablespace Undo: No
          0x00000000 prev ctl uba: 0x01003327.0187.03
prev ctl max cmt scn: 0x0000.002d1e27 prev tx cmt scn: 0x0000.002d1e39
txn start scn: 0x0000.002d22c3 logon user: 114 prev brb: 16790299 prev bcl: 0 BuExt idx: 0 flg2: 0
KDO undo record:
KTB Redo
op: 0x03 ver: 0x01
compat bit: 4 (post-11) padding: 1
op: Z
KDO Op code: DRP row dependencies Disabled
      xtype: XA flags: 0x00000000 bdba: 0x034000bb hdba: 0x034000ba
itli: 1 ispac: 0 maxfr: 4858
tabn: 0 slot: 0(0x0)
```

CHANGE #5 CON_ID:1 TYP:0 CLS:20 AFN:4 DBA:0x01003327 OBJ:4294967295 SCN:0x0000.002d228c SEQ:
1 OP:5.1 ENC:0 RBL:0

ktudb redo: siz: 112 spc: 7776 flg: 0x0012 seq: 0x0187 rec: 0x04
xid: 0x0002.013.000008d3

ktubl redo: slt: 19 rci: 0 opc: 11.1 [objn: 95424 objd: 95424 tsn: 3]

Undo type: Regular undo Begin trans Last buffer split: No

Temp Object: No

Tablespace Undo: No
0x00000000 prev ctl uba: 0x01003327.0187.03

prev ctl max cmt scn: 0x0000.002d1e27 prev tx cmt scn: 0x0000.002d1e39

txn start scn: 0x0000.002d22c3 logon user: 114 prev brb: 16790299 prev bcl: 0 BuExt idx:
0 flg2: 0

KDO undo record:

KTB Redo

op: 0x03 ver: 0x01

compat bit: 4 (post-11) padding: 1

op: Z

KDO Op code: DRP row dependencies Disabled

xtype: XA flags: 0x00000000 bdba: 0x034000bb hdba: 0x034000ba

itli: 1 ispac: 0 maxfr: 4858

tabn: 0 slot: 0(0x0)

Redo in multitenant



- Still one redo for a CDB
- Contains changes for all PDBs

Redo dump



- Each redo record and each change now has CON_ID/CON_UID

```
REDO RECORD - Thread:1 RBA: 0x000066.0000000b.0010 LEN: 0x0334 VLD: 0x0d
```

```
    CON_UID: 3345156736
```

```
SCN: 0x0000.002d14ca SUBSCN: 1 07/08/2014 15:29:41
```

```
(LWN RBA: 0x000066.0000000b.0010 LEN: 0002 NST: 0001 SCN: 0x0000.002d14c9)
```

```
CHANGE #1 CON_ID:3 TYP:0 CLS:1 AFN:13 DBA:0x034000bf OBJ:95424 SCN:  
0x0000.00285536 SEQ:2 OP:11.2 ENC:0 RBL:0
```

- But some of the stuff is global

```
CHANGE #2 CON_ID:1 TYP:0 CLS:29 AFN:4 DBA:0x010000e0 OBJ:4294967295 SCN:  
0x0000.002d1473 SEQ:1 OP:5.2 ENC:0 RBL:0
```

Redo dump (non-CDB)



- Each redo record and each change now has CON_ID/CON_UID

```
REDO RECORD - Thread:1 RBA: 0x0008e6.00000002.0118 LEN: 0x023c VLD: 0x01 CON_UID: 0
SCN: 0x0000.00f51e47 SUBSCN: 4 07/13/2014 19:17:13
CHANGE #3 CON_ID:0 TYP:0 CLS:1 AFN:2 DBA:0x028000bf OBJ:91537 SCN:0x0000.00f51e47 SEQ:1
      OP:11.2 ENC:0 RBL:0
```

- And the global stuff:

```
CHANGE #1 CON_ID:0 TYP:0 CLS:37 AFN:4 DBA:0x010001c8 OBJ:4294967295 SCN:0x0000.00f51bff
SEQ:1 OP:5.2 ENC:0 RBL:0
```

- So all id's are 0
- Note: Multi-tenant does a bit of reordering of the changes, but that's immaterial

CON_ID in redo

- Changes in one redo record can be in different order (some 11g, much 12c)
- One redo record, one logical change will often span CDB and one PDB

```
REDO RECORD - Thread:1 RBA: 0x000066.0000000b.0010 LEN: 0x0334 VLD: 0x0d CON_UID:
3345156736
CHANGE #1 CON_ID:3 TYP:0 CLS:1 AFN:13 DBA:0x034000bb OBJ:95424 SCN:0x0000.00285536
SEQ:2 OP:11.2 ENC:0 RBL:0
CHANGE #2 CON_ID:1 TYP:0 CLS:19 AFN:4 DBA:0x01000090 OBJ:4294967295 SCN:
0x0000.002d228d SEQ:1 OP:5.2 ENC:0 RBL:0
CHANGE #3 CON_ID:3 TYP:0 CLS:1 AFN:13 DBA:0x034000c3 OBJ:95425 SCN:0x0000.002d22c3
SEQ:1 OP:10.2 ENC:0 RBL:0
CHANGE #4 CON_ID:1 TYP:0 CLS:19 AFN:4 DBA:0x01000090 OBJ:4294967295 SCN:
0x0000.002d22c4 SEQ:1 OP:5.4 ENC:0 RBL:0
CHANGE #5 CON_ID:1 TYP:0 CLS:20 AFN:4 DBA:0x01003327 OBJ:4294967295 SCN:
0x0000.002d228c SEQ:1 OP:5.1 ENC:0 RBL:0
CHANGE #6 CON_ID:1 TYP:0 CLS:20 AFN:4 DBA:0x01003327 OBJ:4294967295 SCN:
0x0000.002d22c4 SEQ:1 OP:5.1 ENC:0 RBL:0
```

CON_ID in redo



- Oracle does not allow transactions across PDBs

```
alter session set container=UNITPDB1;
```

```
Session altered.
```

```
SQL> update ...
```

```
update ...
```

```
      *
```

```
ERROR at line 1:
```

```
ORA-65023: active transaction exists in container CDB$ROOT
```


Many things are not unique

- Tablespace names (but note that there is only CDB undo)

```
CON_ID TABLESPACE_NAME
```

```
-----  
1 SYSAUX  
1 SYSTEM  
1 TEMP  
1 UNDOTBS1  
1 USERS  
2 SYSAUX  
2 SYSTEM  
2 TEMP  
3 EXAMPLE  
3 SOE  
3 SYSAUX  
3 SYSTEM  
and so on...
```

Example – clone PDB

- `CREATE PLUGGABLE DATABASE unitpdb2 FROM unitpdb1;`

```
SQL> select owner, object_name, con_id from cdb_objects where  
data_object_id=95424;
```

OWNER	OBJECT_NAM	CON_ID
HR	JOBS	3
HR	JOBS	4

- The same object_id is a coincidence, thanks to the clone; from now on, the object_ids are generated from independent sequences
- Now on boths PDBs:
`insert into hr.jobs values ('TE_TEST', 'Test job', 1, 100);`

REDO RECORD - Thread:1 RBA: 0x000068.00000015.008c LEN: 0x02e0 VLD: 0x09 CON_UID: **3345156736**
SCN: 0x0000.002d22c4 SUBSCN: 1 07/08/2014 15:58:58
CHANGE #1 **CON_ID:3** TYP:0 CLS:1 **AFN:13** DBA:0x034000bb **OBJ:95424** SCN:0x0000.00285536 SEQ:2 OP:
11.2 ENC:0 RBL:0
fb: --H-FL-- lb: 0x1 cc: 4
col 0: [7] 54 45 5f 54 45 53 54
col 1: [8] 54 65 73 74 20 6a 6f 62
CHANGE #2 CON_ID:1 TYP:0 CLS:19 AFN:4 DBA:0x01000090 OBJ:4294967295 SCN:0x0000.002d228d SEQ:
1 OP:5.2 ENC:0 RBL:0
ktudh redo: slt: 0x0013 sqn: 0x000008d3 flg: 0x0052 siz: 112 fbi: 0
uba: 0x01003327.0187.04 pxid: 0x0000.000.00000000 pdbid:3345156736
CHANGE #4 CON_ID:1 TYP:0 CLS:19 AFN:4 DBA:0x01000090 OBJ:4294967295 SCN:0x0000.002d22c4 SEQ:
1 OP:5.4 ENC:0 RBL:0
ktucm redo: slt: 0x0013 sqn: 0x000008d3 srt: 0 sta: 9 flg: 0x2
CHANGE #5 CON_ID:1 TYP:0 CLS:20 AFN:4 DBA:0x01003327 OBJ:4294967295 SCN:0x0000.002d228c SEQ:
1 OP:5.1 ENC:0 RBL:0
ktudb redo: siz: 112 spc: 7776 flg: 0x0012 seq: 0x0187 rec: 0x04
xid: 0x0002.013.000008d3
ktubl redo: slt: 19 rci: 0 opc: 11.1 [**objn: 95424 objd: 95424** tsn: 3]
Undo type: Regular undo Begin trans Last buffer split: No
Temp Object: No

```
REDO RECORD - Thread:1 RBA: 0x000068.00000017.0118 LEN: 0x01a0 VLD: 0x01 CON_UID: 3918633952
SCN: 0x0000.002d22ca SUBSCN: 1 07/08/2014 15:59:07
CHANGE #1 CON_ID:4 TYP:0 CLS:1 AFN:17 DBA:0x034000bd OBJ:95424 SCN:0x0000.002d22ca SEQ:1 OP:
11.2 ENC:0 RBL:0
fb: --H-FL-- lb: 0x1 cc: 4
col 0: [ 7] 54 45 5f 54 45 53 54
col 1: [ 8] 54 65 73 74 20 6a 6f 62
CHANGE #2 CON_ID:1 TYP:0 CLS:29 AFN:4 DBA:0x010000e0 OBJ:4294967295 SCN:0x0000.002d228f SEQ:
1 OP:5.2 ENC:0 RBL:0
ktudh redo: slt: 0x0011 sqn: 0x000007df flg: 0x0052 siz: 112 fbi: 0
uba: 0x01000d7c.01ca.05 pxid: 0x0000.000.00000000 pdbid:3918633952
CHANGE #4 CON_ID:1 TYP:0 CLS:29 AFN:4 DBA:0x010000e0 OBJ:4294967295 SCN:0x0000.002d228f SEQ:
1 OP:5.4 ENC:0 RBL:0
ktucm redo: slt: 0x0011 sqn: 0x000007df srt: 0 sta: 9 flg: 0x2
CHANGE #5 CON_ID:1 TYP:0 CLS:30 AFN:4 DBA:0x01000d7c OBJ:4294967295 SCN:0x0000.002d228e SEQ:
1 OP:5.1 ENC:0 RBL:0
ktudb redo: siz: 112 spc: 7638 flg: 0x0012 seq: 0x01ca rec: 0x05
xid: 0x0007.011.000007df
ktubl redo: slt: 17 rci: 0 opc: 11.1 [objn: 95424 objd: 95424 tsn: 3]
Undo type: Regular undo Begin trans Last buffer split: No
Temp Object: No
```

Backup

- The CDB has metadata about PDBs, so it must be backed up, too
- The archivelogs are global, but they are needed for recovery of any PDB
- Backup at root level can include archivelogs
 - backup of whole database
 - backup of CDB container only
 - backup of any PDB(s) (or datafile or tablespace)
- Backup at PDB level cannot access archivelogs
 - backup of this PDB only (or datafile or tablespace)
- So think twice about your backup strategy and your archive log backups
- Backup optimization may be your friend here

Backup whole database



- Easiest approach, treat your database as one (same SLA, application...)
- Note each PDB gets own backup set

```
RMAN> backup database plus archivelog;  
current log archived
```

...

```
Starting backup at 20-JUL-14  
using channel ORA_DISK_1  
name=+.../EEFFBB158E172CB0E04500000000001/DATAFILE/system.462.835807737  
name=+.../EEFFBB158E172CB0E04500000000001/DATAFILE/users.352.835807833  
channel ORA_DISK_1: backup set complete, elapsed time: 00:05:06  
name=+.../DATAFILE/sysaux.463.835805697  
name=+.../DATAFILE/system.444.835805791
```

Backup just PDB



- Or backup each PDB, but then you have to manage the archlogs

```
RMAN> backup pluggable database unitpdb1 plus archivelog;  
current log archived  
input archived log thread=1 sequence=139 RECID=129 STAMP=853427615
```

```
Starting backup at 20-JUL-14  
input datafile file number=00008 name=+.../EEFFB158E172CB0E045000000000001/  
DATAFILE/system.462.835807737  
input datafile file number=00010 name=+.../EEFFB158E172CB0E045000000000001/  
DATAFILE/users.352.835807833  
channel ORA_DISK_1: backup set complete, elapsed time: 00:02:46  
Finished backup at 20-JUL-14
```

Backup a PDB tablespace from root



- Documentation says it cannot be done, due to inability to unambiguously identify the tablespace

```
RMAN> backup tablespace UNITPDB1:SYSTEM;
Starting backup at 20-JUL-14
channel ORA_DISK_1: starting full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
input datafile file number=00008 name=+DATA/UNITCDB/
EEFFBB158E172CB0E045000000000001/DATAFILE/system.462.835807737
piece handle = +DATA/UNITCDB/EEFFBB158E172CB0E04500000000001/BACKUPSET/
2014_07_20/nnndf0_tag20140720t143449_0.333.853425291 tag=TAG20140720T143449
comment=NONE
Finished backup at 20-JUL-14
```


And restore...



```
RMAN> alter pluggable database unitpdb1 close;
```

```
RMAN> restore tablespace UNITPDB1:SYSTEM;
```

```
channel ORA_DISK_1: restoring datafile 00008 to +.../  
EEFFBB158E172CB0E04500000000001/DATAFILE/system.462.835807737
```

```
channel ORA_DISK_1: restore complete, elapsed time: 00:01:35
```

```
RMAN> recover tablespace UNITPDB1:SYSTEM;
```

```
starting media recovery
```

```
media recovery complete, elapsed time: 00:00:02
```

```
RMAN> alter pluggable database unitpdb1 open;
```

DataGuard



- DG is almost oblivious to existence of CDB/PDB
- DG works at CDB level, protecting datafiles
- Target is always a CDB (because we ship redo, thus again, at CDB level)
- Can skip PDB by offlining datafiles (12.1.0.2: STANDBYS clause of the SQL CREATE PLUGGABLE DATABASE)
- Plugging in new PDB – need to copy files to standby, too (ADG does a copy itself if it's a clone from another PDB)

Patching

- Most/all changes in DB are at CDB container (thanks to metadata and object links) or just binaries
- Create a new OH and CDB, patch it
- Replug PDBs into the new CDB (*might* need some further scripts – see patch docs, e.g. 12.1.0.2 needs catupgrd.sql), no copy of datafiles needed
- Or do it in one go (but with more downtime) – use catctl.pl to run scripts on all PDBs

One last trick and two traps



- You can use `TWO_TASK` (instead of `ORACLE_SID`) for local sqlplus
- Beware that RMAN in PDB does not see any archive logs; check and test your scripts
- `DROP PLUGGABLE DATABASE` will deregister backups, too

Thank you - Q&A

Visit us at **Booth 1931**

Challenge our team of tech specialists by answering tricky questions!

Enter our competition and be sure to follow #moveituseit for clues!

RepAttack: Children Creativity Museum:
Tuesday 27th 1-5pm

info@dbvisit.com
[@dbvisit](#)

