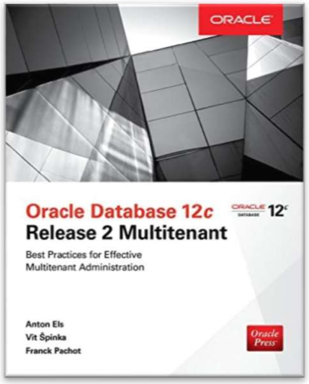
Infrastructure at your Service.

12*c* Single-Tenant:
Multitenant Features for All Editions

dbi services

About me



ORACLE

Oracle Database 12*c* Release 2 Multitenant

ORACLE DATABASE 12c

Best Practices for Effective Multitenant Administration

Anton Els
Vít Špinka
Franck Pachot

Oracle Press

Franck Pachot
Principal consultant
Oracle Technology Leader

Mobile +41 79 963 27 22
franck.pachot@dbi-services.com
www.dbi-services.com

ORACLE
Certified Master
Oracle Database 12*c* Administrator

OakTable®

ORACLE ACE Director

Follow @FranckPachot

Who we are
dbi services

Experts At Your Service

- Over 50 specialists in IT infrastructure
- Certified, experienced, passionate

Based In Switzerland

- 100% self-financed Swiss company
- Over CHF 8.4 mio. turnover

Leading In Infrastructure Services

- More than 150 customers in CH, D, & F
- Over 50 SLAs dbi FlexService contracted

Certified Management System SQS ISO / IEC 20000-1

GREAT PLACE TO WORK® Best Companies to Work for 2017 Switzerland

12*c* Single-Tenant
Agenda

1. Multitenant architecture
2. Data movement: pluggable databases
3. Security: lockdown profiles
4. Fast upgrade: plug to new version
5. Agility: local undo, flashback PDB
6. Conclusion: Multitenant is not an option

Multitenant architecture
Definition of multitenant

The feature was originally called Oracle Pluggable Databases

- Dictionary separation (CDB / PDB)
- Transportable Tablespace for SYSTEM, SYSAUX and UNDO
- Unplug/plug of PDBs, clone, relocate, etc.

It goes beyond with Multitenant Option

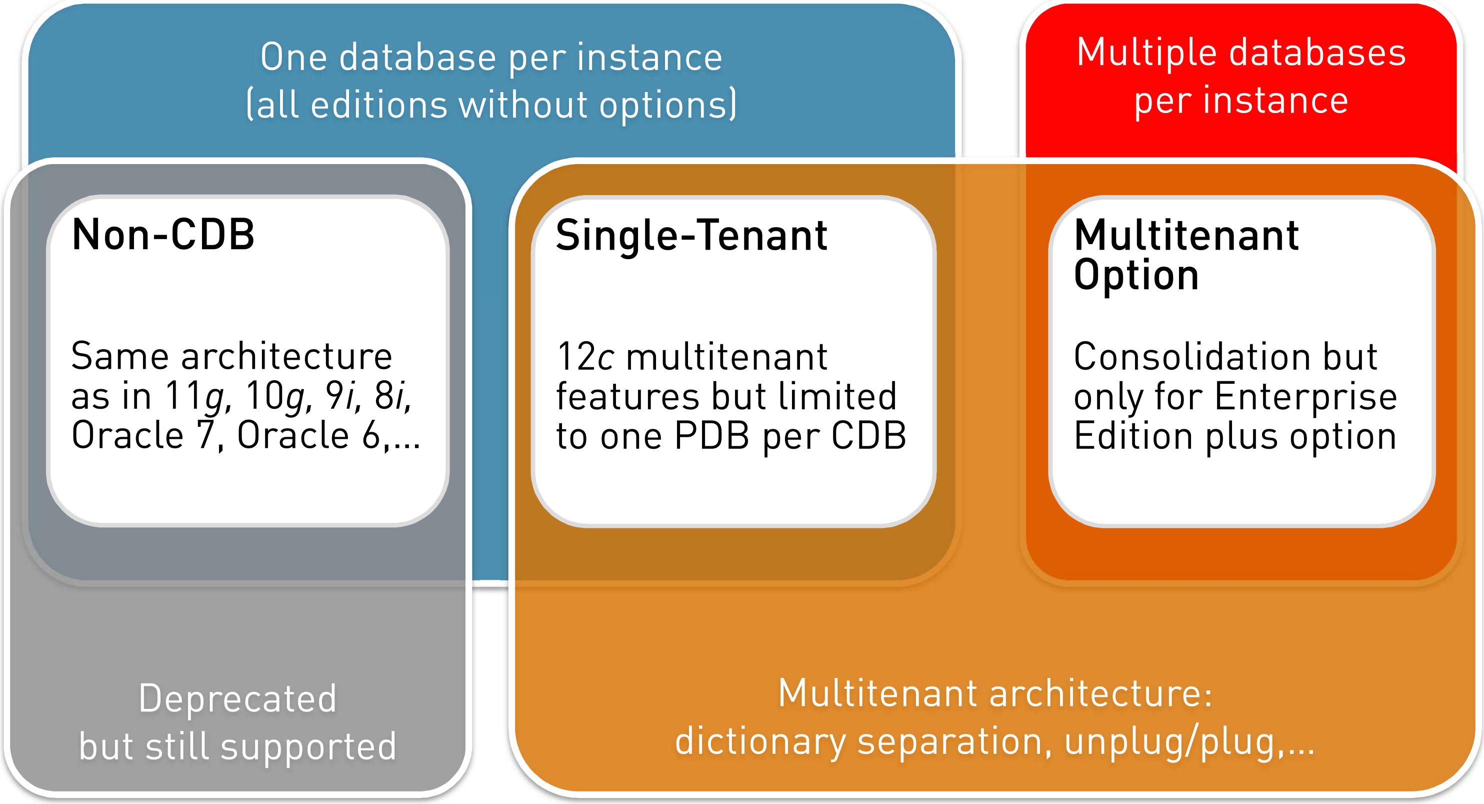
- One CDB can manage multiple PDBs
- Ready for consolidation into the Cloud

Finally the feature was released as Oracle Multitenant

- Ready for consolidation into the Cloud

Multitenant architecture

Definition of multitenant



One database per instance
(all editions without options)

Multiple databases
per instance

Non-CDB

Same architecture as in 11*g*, 10*g*, 9*i*, 8*i*, Oracle 7, Oracle 6,...

Single-Tenant

12*c* multitenant features but limited to one PDB per CDB

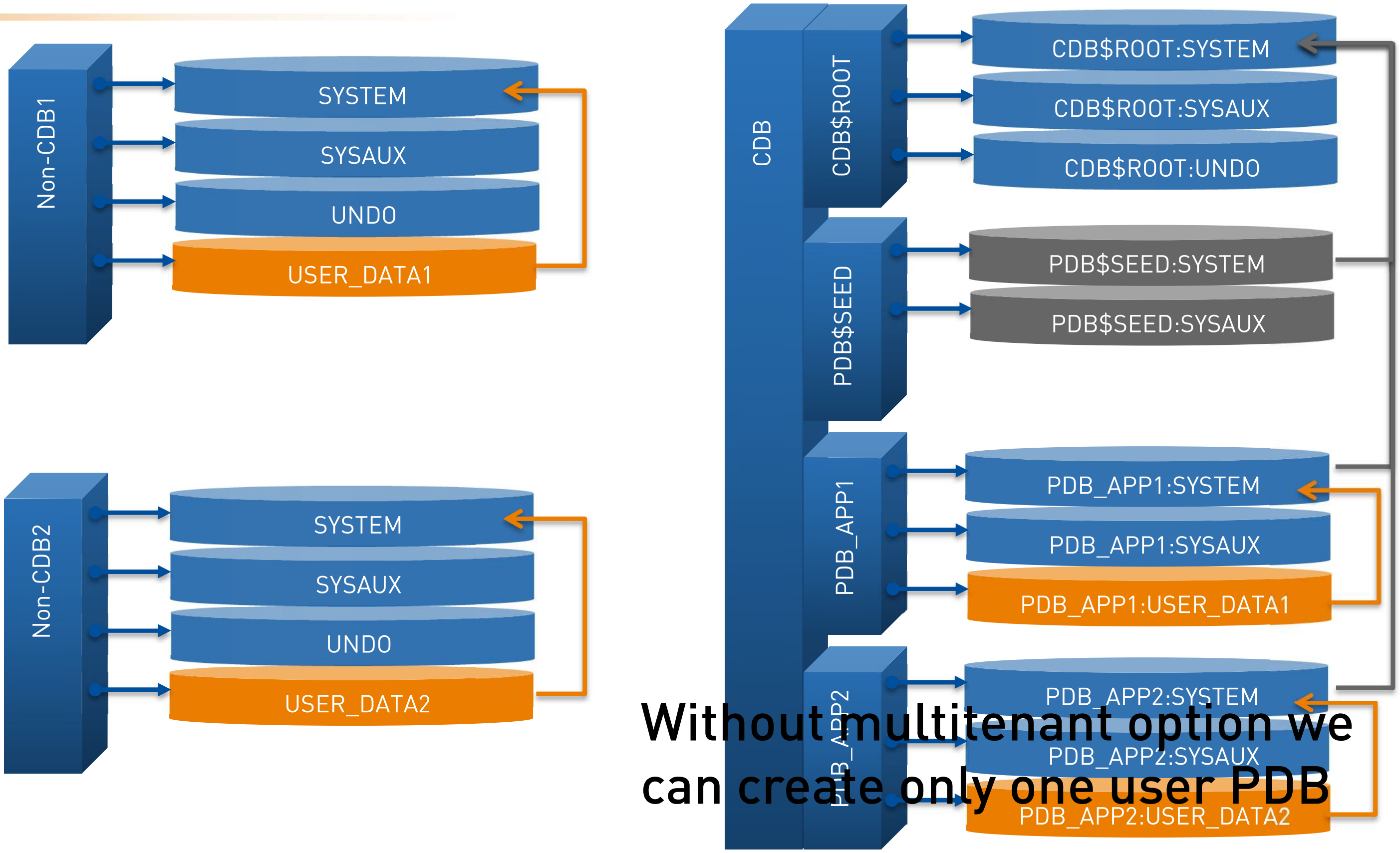
Multitenant Option

Consolidation but only for Enterprise Edition plus option

Deprecated
but still supported

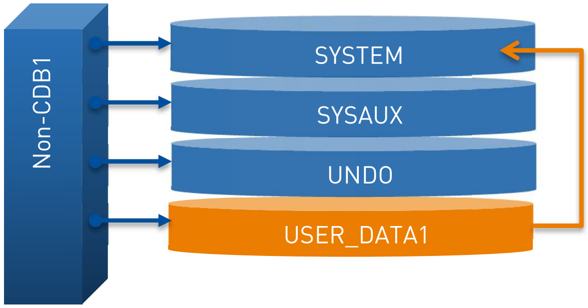
Multitenant architecture:
dictionary separation, unplug/plug,...

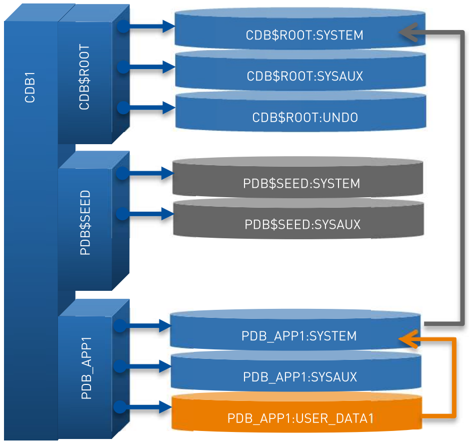
Multitenant architecture
PDBs are tablespaces

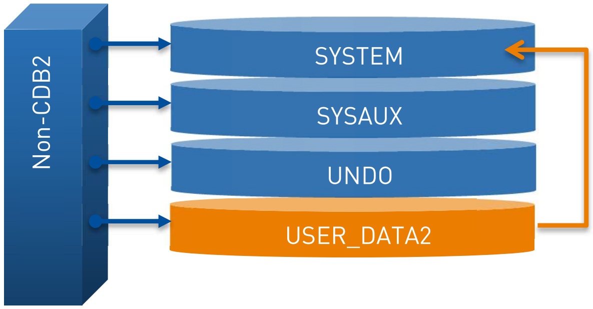


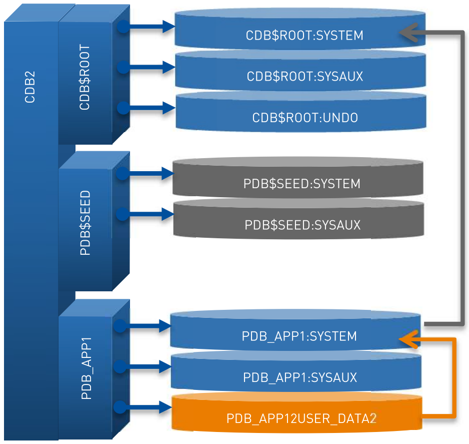
Without multitenant option we can create only one user PDB

Multitenant architecture
Non-CDB vs. single-tenant

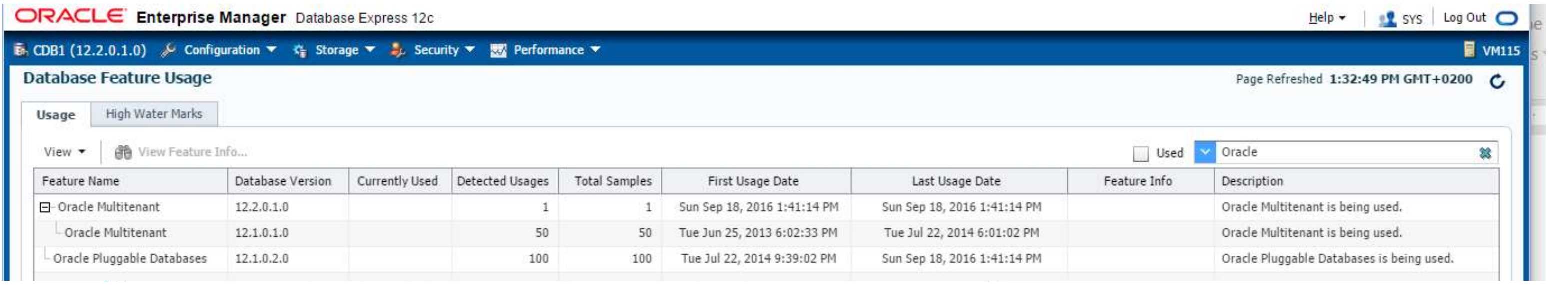








Multitenant architecture
Database Feature usage



Database Feature Usage

Page Refreshed 1:32:49 PM GMT+0200

Feature Name	Database Version	Currently Used	Detected Usages	Total Samples	First Usage Date	Last Usage Date	Feature Info	Description
⊟ Oracle Multitenant	12.2.0.1.0		1	1	Sun Sep 18, 2016 1:41:14 PM	Sun Sep 18, 2016 1:41:14 PM		Oracle Multitenant is being used.
Oracle Multitenant	12.1.0.1.0		50	50	Tue Jun 25, 2013 6:02:33 PM	Tue Jul 22, 2014 6:01:02 PM		Oracle Multitenant is being used.
Oracle Pluggable Databases	12.1.0.2.0		100	100	Tue Jul 22, 2014 9:39:02 PM	Sun Sep 18, 2016 1:41:14 PM		Oracle Pluggable Databases is being used.

In DBA_FEATURE_USAGE_STATISTIC the feature is called

- "Oracle Multitenant"
- 12.1.0.2 shows "Oracle Pluggable Databases" - Bug 20718081

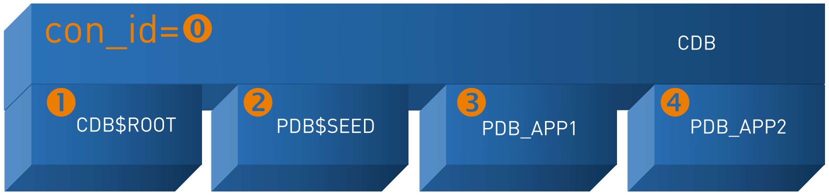
It detects multitenant architecture, not multitenant option usage

- Detail is in DBA_FEATURE_USAGE_STATISTICS.AUX_COUNT

```
SQL> select name, aux_count from dba_feature_usage_statistics;

NAME                    AUX_COUNT
--------------------   ----------
Oracle Multitenant           4096
```

Multitenant architecture
Feature usage detection

- CON_ID=0 is the CDB
- CON_ID=1 is CDB$ROOT
- CON_ID=2 is PDB$SEED
- CON_ID=3 is the first PDB



con_id=⓿

CDB

❶ CDB$ROOT

❷ PDB$SEED

❸ PDB_APP1

❹ PDB_APP2

In Standard Edition, ORA-65010 raised if creating con_id>=3

In Enterprise Edition with option, up to 252 PDBs (or even 4096 in the Oracle Cloud)

When not having the option, we can ensure that only one PDB is created:

```
SQL> alter system set max_pdbs=1 comment='single-tenant';
```

Multitenant architecture
Single-Tenant vs. non-CDB

−

So, from what you have seen, Single-Tenant is a database

- With more datafiles, slightly more memory
- And the risk to activate an expensive option in 12.1

+

But there's more reasons to use it:

- The non-CDB is deprecated
- It brings new features:
  - Data movement: pluggable databases
  - Fast upgrade: plug to new version
  - Security: lockdown profiles
  - Agility: flashback PDB

Multitenant architecture
Demo

```
 SQL> show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 PDB                            READ WRITE NO

SQL> exec dbms_feature_usage_internal.exec_db_usage_sampling(sysdate);
PL/SQL procedure successfully completed.

SQL> select * from dba_feature_usage_statistics
     where name like '%Pluggable%' or name like '%Multitenant%';

NAME                       VERSION            DETECTED_USAGES  AUX_COUNT
-------------------------- ------------------ --------------- ----------
Oracle Multitenant         12.2.0.1.0                       2          1

SQL> show parameter max_pdbs
NAME                        TYPE        VALUE
--------------------------- ----------- ------------------------------
max_pdbs                    integer     4098
```

12*c* Single-Tenant
Agenda

1. Multitenant architecture
2. Data movement: pluggable databases
3. Fast upgrade: plug to new version
4. Security: lockdown profiles
5. Agility: local undo, flashback PDB
6. Conclusion: Multitenant is not an option

Pluggable databases
Data Movement

exp/imp

- Logical transport (like select/insert) -> very slow

Data Pump export / import

- Includes direct-path but still logical -> flexible but slow

RMAN Duplicate

- Physical transport -> fast but same database version

Transportable Tablespaces

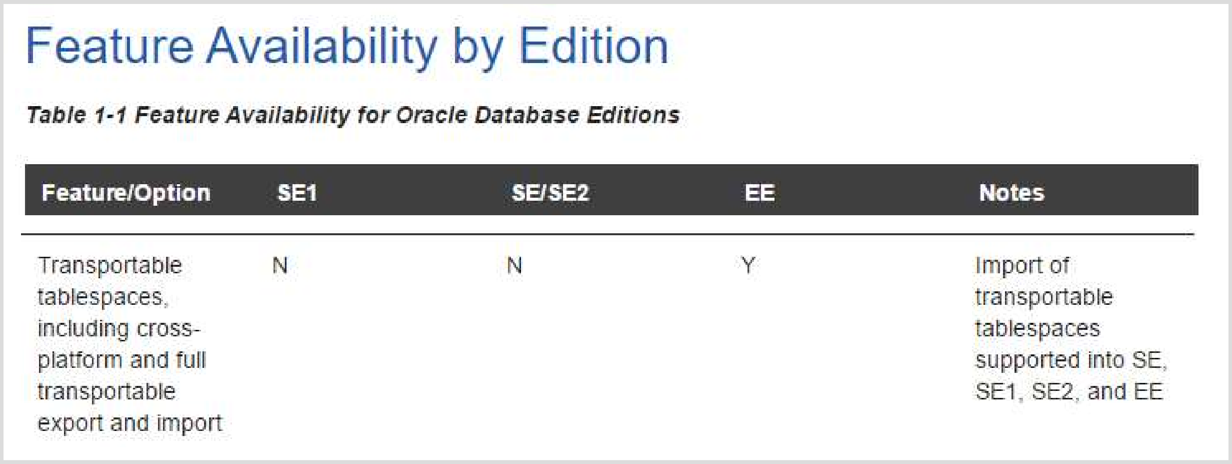
- Physical transport of user data, logical transport of metadata
  -> fast and cross version
  but source in read-only or RMAN

Unplug / Plug or PDB clone

- Physical transport of user data and metadata
  -> the fastest
  -> online in 12.2 in local undo

Pluggable databases
Transportable tablespaces

Transportable tablespace not available in Standard Edition



Feature Availability by Edition

Table 1-1 Feature Availability for Oracle Database Editions

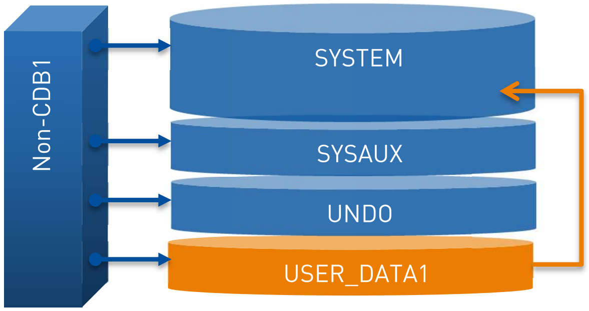
Feature/Option	SE1	SE/SE2	EE	Notes
Transportable tablespaces, including cross-platform and full transportable export and import	N	N	Y	Import of transportable tablespaces supported into SE, SE1, SE2, and EE

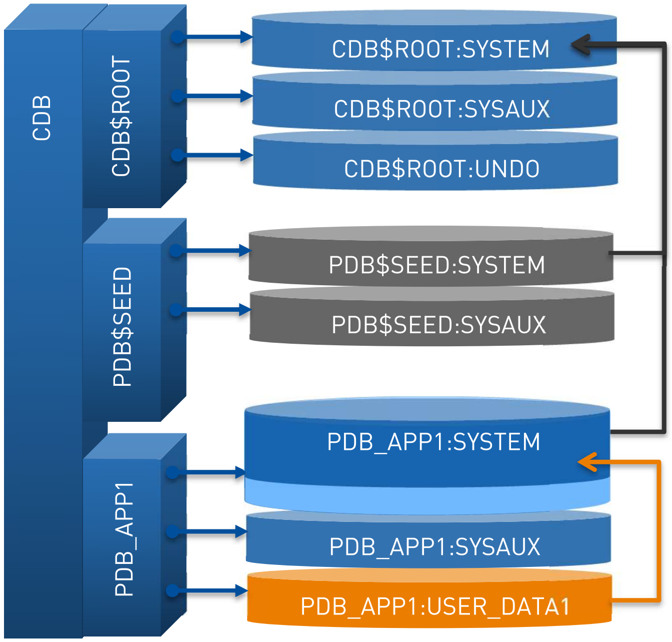
Unplug / Plug or Clone PDB is faster and is available in SE

- and even online (no need for read-only in local undo mode)
- relocate, proxy,…

Pluggable databases
From non-CDB to PDB

1. Source Non-CDB in read only and new CDB ready
2. Unplug from non-CDB and plug into CDB
3. noncdb_to_pdb.sql





Pluggable databases

Clones and refreshable clones

Cloning online

- do not need the source to be read-only
- and apply redo to the destination
- by default not refreshable and can be open read-write

```
SQL> create pluggable database … from …@…
     refresh mode none; -- default
```

You can refresh it later

- only when it is closed, redo is applied
- refreshable clone cannot be open read-write
- but can be opened read-only

```
SQL> create pluggable database … from …@…
     refresh mode manual;
```

Can also be automatically refreshed

- every n minutes, only if PDB is closed
  - if source PDB is not available, an alternative location to find archived logs can be set with REMOTE_RECOVERY_FILE_DEST instance parameter

```
SQL> create pluggable database … from …@…
     refresh mode every 5 minutes;
```

Pluggable databases
Near zero-downtime relocate PDB

Because you can clone online

- you can redirect connections to the target PDB
- and drop the source PDB
  - here is online PDB relocate to another CDB

Relocate availability normal

- source sessions are killed
- sessions must connect to new listener if different

```
SQL> create pluggable database … from …  relocate availability normal; -- default
```

Relocate availability max

- a 'tombstone' PDB is left to automatically forward connections to new listener
  - until it is dropped

```
SQL> create pluggable database … from … relocate availability max;
```

  - This is 'near zero downtime' because sessions are killed but can reconnect immediately

Pluggable databases
Data Movement summary

−

With non-CDB

- You can transport data only with TTS
- Metadata has to be re-created, indexes have to be rebuilt
- And this transport is not available in Standard Edition

+

In Single-Tenant

- You can transport physically data + metadata
- In all editions
- Online in 12*c*R2
- With a simple command

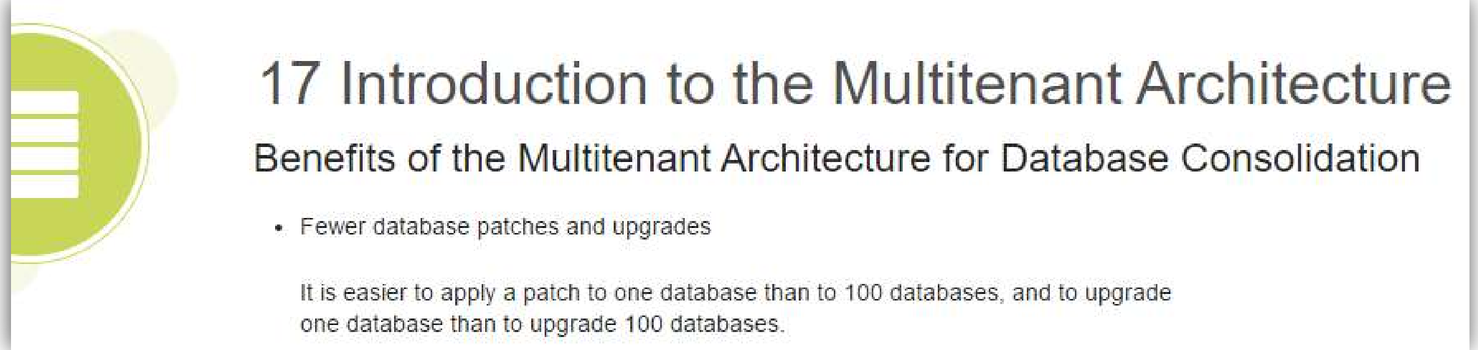
12*c* Single-Tenant
Agenda

1. Multitenant architecture
2. Data movement: pluggable databases
3. Fast upgrade: plug to new version
4. Security: lockdown profiles
5. Agility: local undo, flashback PDB
6. Conclusion: Multitenant is not an option

Upgrade by plug-in
Fast?

Documentation says it's as easy and faster:

- As it is only links you need only to upgrade CDB$ROOT



17 Introduction to the Multitenant Architecture

Benefits of the Multitenant Architecture for Database Consolidation

- Fewer database patches and upgrades

It is easier to apply a patch to one database than to 100 databases, and to upgrade one database than to upgrade 100 databases.

Because all system objects are in CDB$ROOT

- PDB has only links.
- No need to re-create, re-compile all objects
- But…

Upgrade by plug-in
Fast?

After plug-in or clone from previous release

- we need to run catupgrd:

```
$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catctl.pl
-d $ORACLE_HOME/rdbms/admin -l /tmp -c PDBMIG $ORACLE_HOME/rdbms/admin/catupgrd.sql

Argument list for [/u01/app/oracle/product/12202EE/rdbms/admin/catctl.pl]
Run in                     c = PDBMIG
Do not run in              C = 0
Input Directory            d = /u01/app/oracle/product/12202EE/rdbms/admin
Echo OFF                   e = 1
Log Dir                    l = /tmp
```

So... how fast is it?

Upgrade by plug-in
Fast?

```
Component                               Current         Version  Elapsed Time
Name                                    Status          Number   HH:MM:SS

Oracle Server                          UPGRADED      12.2.0.1.0  00:40:20
JServer JAVA Virtual Machine           UPGRADED      12.2.0.1.0  00:05:29
Oracle Real Application Clusters       UPGRADED      12.2.0.1.0  00:00:01
Oracle Workspace Manager               UPGRADED      12.2.0.1.0  00:00:56
OLAP Analytic Workspace                UPGRADED      12.2.0.1.0  00:01:07
Oracle OLAP API                        UPGRADED      12.2.0.1.0  00:00:23
Oracle Label Security                  UPGRADED      12.2.0.1.0  00:00:19
Oracle XDK                             UPGRADED      12.2.0.1.0  00:01:02
Oracle Text                            UPGRADED      12.2.0.1.0  00:00:37
Oracle XML Database                    UPGRADED      12.2.0.1.0  00:02:09
Oracle Database Java Packages          UPGRADED      12.2.0.1.0  00:00:13
Oracle Multimedia                      UPGRADED      12.2.0.1.0  00:01:52
Spatial                                UPGRADED      12.2.0.1.0  00:16:54
Oracle Application Express                VALID     5.0.0.00.31  00:00:08
Oracle Database Vault                  UPGRADED      12.2.0.1.0  00:04:16
Final Actions                                                    00:03:10
Post Upgrade                                                     00:02:11
```

Faster than a CDB (with 3 containers) but not faster than non-CDB

Upgrade by plug-in
Remote clone from previous version

PDB clone can be across versions

- Datafiles are compatible
- System objects are in CDB

Remote clone from 12*c*R1 to 12*c*R2*:*

```
ORA-17628: Oracle error 17630 returned by remote Oracle server
ORA-17630: Mismatch in the remote file protocol version client server
```

- Remote (source) alert.log:

```
ORA-17630: Mismatch in the remote file protocol version client 3 server 2
```

Need to apply patch on source and target:

  - patch18633374: COPYING ACROSS REMOTE SERVERS
  - Or use unplug/plug instead

Then apply catupgrd.sql on the PDB (catcon.pl -c)

Upgrade by plug-in
When?

Upgrade the whole CDB	vs.	Upgrade by PDB plug-in to new CDB	
> Everything is still there > AWR > Backups > Similar to what we know		> Faster than whole CDB > Faster than non-CDB > … in future versions maybe	+
> Takes longer > There are 3 containers		> AWR stays in old CDB > Needs two instances	−

12*c* Single-Tenant
Agenda

1. Multitenant architecture
2. Data movement: pluggable databases
3. Fast upgrade: plug to new version
4. Security: lockdown profiles
5. Agility: local undo, flashback PDB
6. Conclusion: Multitenant is not an option

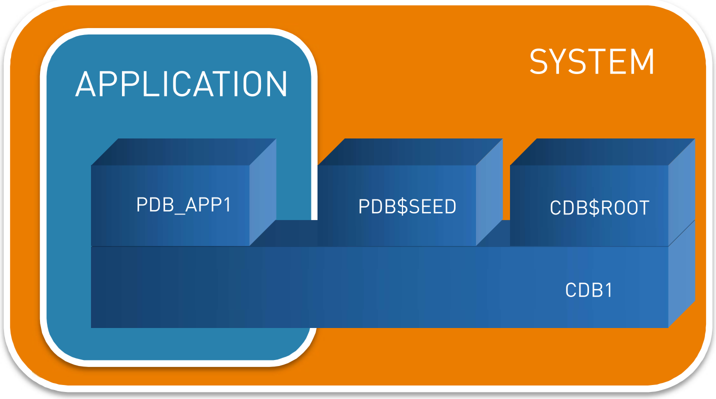
Security
Common and local users

Local users are for

- Application schemas
- Application users
- Application admins

Common users are for

- system administrator
- Oracle maintained
- Monitoring,...



- It look like inheritance but actually it's replicated
  - Altered in CDB$ROOT, changes replicated in PDB
    - Immediately if opened, deferred if closed
    - At creation (from seed, clone, plug-in...)

Single-Tenant vs. non-CDB: separation of roles

Security
Lockdown profiles

Did you ever

```
SQL> grant ALTER SYSTEM to MY_DEVELOPER;
```

- so that they can kill their sessions?
- That's a quite powerful privilege

Alternative:

```
SQL> create procedure kill_session (…) as begin
     if … /* check if session USERNAME */
        execute immediate 'alter system kill session …'
     …
SQL> grant EXECUTE on kill_session to MY_DEVELOPER;
```

- Great encapsulation, but how does it work with TOAD 'kill' button?

In 12*c* (next) Multitenant, there is a solution: lockdown profiles

- available in Single-Tenant but not in non-CDB
- to control PDB local users beyond privileges

Security
Lockdown profiles

You want to delegate some DBA roles to a PDB owner

- You need a finer level than system privileges

```
SQL> create lockdown profile APPDBA_PROF;
Lockdown Profile created.
SQL> select * from DBA_LOCKDOWN_PROFILES;
PROFILE_NAME  RULE_TYPE RULE         CLAUSE CLAUSE_OPT OPTION_VAL STATUS
------------- --------- ------------ ------ ---------- ---------- ------
APPDBA_PROF                                                       DISABLE

SQL> alter session set container=PDB1;
Session altered.
SQL> alter system set pdb_lockdown=APPDBA_PROF;
System altered.
```

With lockdown profiles you can lockdown local users

- disable database options, features, access to system files, network
- control what can be done by some powerful commands:
  - ALTER SYSTEM, ALTER SESSION, ALTER [PLUGGABLE] DATABASE

Security
Lockdown profiles – disable option

You want to disable options (cf. V$OPTION):

- Database queuing (also known as Advanced Queuing)
- Oracle Data Guard
- Partitioning
- Real Application Clusters

for a PDB

```
SQL> alter lockdown profile APPDBA_PROF disable option = ('Partitioning');
Lockdown Profile altered.

SQL> select * from DBA_LOCKDOWN_PROFILES;
PROFILE_NAME  RULE_TYPE RULE         CLAUSE CLAUSE_OPT OPTION_VAL STATUS
------------- --------- ------------ ------ ---------- ---------- ------
APPDBA_PROF   OPTION    PARTITIONING                              DISABLE
```

- ... disable option all;
- ... disable option all except=(...);

Security
Demo

```
…

SQL> alter lockdown profile DEMO_LOCKDOWN
  2   enable option = ('Partitioning');

Lockdown Profile altered.

SQL> select * from dba_lockdown_profiles;

PROFILE_NAME  RULE_TYPE RULE                                 CLAUSE       CLAUSE_OPTION  STATUS
------------- --------- ------------------------------------ ------------ -------------- -------
DEMO_LOCKDOWN FEATURE   AWR_ACCESS                                                       DISABLE
DEMO_LOCKDOWN FEATURE   DROP_TABLESPACE_KEEP_DATAFILES                                   DISABLE
DEMO_LOCKDOWN OPTION    PARTITIONING                                                     ENABLE
DEMO_LOCKDOWN STATEMENT ALTER SYSTEM                                                     DISABLE
DEMO_LOCKDOWN STATEMENT ALTER SYSTEM                         KILL SESSION                ENABLE


DEMO_LOCKDOWN STATEMENT ALTER SYSTEM                         SET          OPTIMIZER_MODE ENABLE


PRIVATE_DBAAS                                                                            EMPTY
PUBLIC_DBAAS                                                                             EMPTY
SAAS                                                                                     EMPTY
```

Security
Lockdown profiles – disable alter system

You want to disable:

- Some ALTER SYSTEM commands
- But not revoke whole ALTER SYSTEM privilege

for a PDB

```
SQL> alter lockdown profile APPDBA_PROF disable statement = ('ALTER SYSTEM')
                                        clause all except = ('KILL SESSION');

SQL> select * from DBA_LOCKDOWN_PROFILES;

PROFILE_NAME  RULE_TYPE RULE         CLAUSE      CLAUSE_OPT OPTION_VAL STATUS
------------- --------- ------------ ----------- ---------- ---------- -------
APPDBA_PROF   OPTION    PARTITIONING                                   DISABLE
APPDBA_PROF   STATEMENT ALTER SYSTEM KILL SESSION                       ENABLE
```

- Any user will get an 'ORA-01031: insufficient privileges' for any ALTER SYSTEM command, except an ALTER SYSTEM KILL SESSION

Only ALTER SYSTEM commands are currently documented

Security
Lockdown profiles – disable alter system

You want to disable:

- Some ALTER SYSTEM commands
- Allow ALTER SYSTEM SET, but for some parameters only

```
SQL> alter lockdown profile APPDBA_PROF disable statement = ('ALTER SYSTEM')
                                                   clause = ('SET');
Lockdown Profile altered.

SQL> alter lockdown profile APPDBA_PROF  enable statement = ('ALTER SYSTEM')
clause = ('SET') option = ('undo_retention', 'temp_undo_enabled',
'resumable_timeout', 'cursor_sharing', 'session_cached_cursors', 'heat_map',
'resource_manager_plan', 'optimizer_dynamic_sampling');
Lockdown Profile altered.

SQL> select * from DBA_LOCKDOWN_PROFILES where profile_name='APPDBA_PROF';

PROFILE_NAME  RULE_TYPE RULE         CLAUSE  CLAUSE_OPTION              STATUS
------------- --------- ------------ ------- ------------------------- -------
APPDBA_PROF   STATEMENT ALTER SYSTEM SET                               DISABLE
APPDBA_PROF   STATEMENT ALTER SYSTEM SET     CURSOR_SHARING             ENABLE
…
```

Security
Lockdown profiles – disable alter system

Example:

```
SQL> alter system set optimizer_dynamic_sampling=4;
System altered.

SQL> alter system set optimizer_index_cost_adj=1;
alter system set optimizer_index_cost_adj=1
*
ERROR at line 1:
ORA-01031: insufficient privileges
```

When disabling, we can set the value to set:

```
alter lockdown profile APPDBA_PROF disable statement=('ALTER SYSTEM')
clause=('SET') option=('cursor_sharing') value=('EXACT');
```

- Sets the value to in spfile
  - restart the PDB after setting the lockdown profile

Security
Lockdown profiles – disable alter system

Don't forget to:

```
SQL> alter lockdown profile APPDBA_PROF
                                        disable statement = ('ALTER SYSTEM')
                                        clause = ('SET')
                                        option=('pdb_lockdown');
```

- So that the DBA cannot reset pdb_lockdown

ALTER SYSTEM RESET

- is also disabled by clause=('SET')

Undocumented but possible:

- ALTER SESSION (same clause/option than ALTER SYSTEM)
- ALTER DATABASE
- ALTER PLUGGABLE DATABASE

Security
Lockdown profiles – disable features

You want to disable:

- Network access
  - DBMS_DEBUG_JDWP, UTL_HTTP, UTL_INADDR, UTL_SMTP, UTL_TCP
- XDB
- COMMON_SCHEMA_ACCESS through proxy users

for a PDB

```
SQL> alter lockdown profile APPDBA_PROF disable feature =
('UTL_HTTP','UTL_SMTP','UTL_TCP');
```

- ‘NETWORK_ACCESS’ disables all networking packages
- ‘OS_ACCESS’ disables all file manipulation
- DROP_TABLESPACE_KEEP_DATAFILES needed to drop a tablespace without 'including datafiles' with non-OMF datafile
- 'AWR_ACCESS' needed to create snapshot...

Security
PATH_PREFIX and CREATE_FILE_DEST

We want to limit the access to filesystems by a PDB

In 12.1

- PATH_PREFIX for directories

In 12.2

- CREATE_FILE_DEST for directories

```
SQL>  create pluggable database PDB1
      admin user admin identified by password
      create_file_dest='/u02/app/oracle/oradata/CDB2/PDB1';
Pluggable database created.
…
SQL> create tablespace APPDATA datafile '/tmp/appdata.dbf' size 5M;
create tablespace APPDATA datafile '/tmp/appdata.dbf' size 5M
*
ERROR at line 1:
ORA-65250: invalid path specified for file - /tmp/appdata.dbf
```

Security
PDB_OS_CREDENTIALS

A user can run a program on the host through

- dbms_scheduler
- external procedure
- external table pre-processor

You don’t want it to run with the oracle user

You create the credential for another OS user

```
exec dbms_credential.create_credential(
 credential_name=>'PDB1_OS_USER', username=>'limitedUser', password=>'secret'
);
```

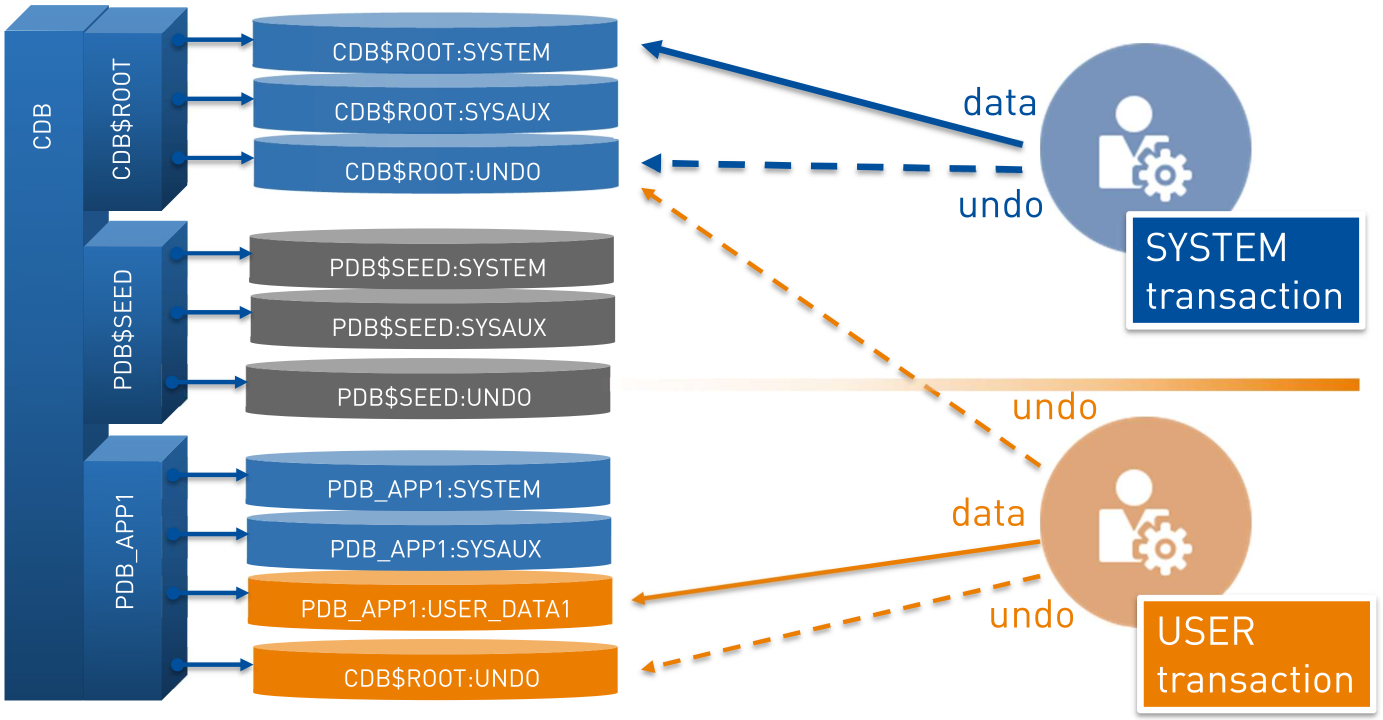
And limit a PDB to use this user:

```
alter session set container=PDB1;
alter system set pdb_os_credential=CDB_PDB_OS_USER scope=spfile;
```

12*c* Single-Tenant
Agenda

1. Multitenant architecture
2. Data movement: pluggable databases
3. Fast upgrade: plug to new version
4. Security: lockdown profiles
5. Agility: local undo, flashback PDB
6. Conclusion: Multitenant is not an option

Flashback PDB
Shared or local UNDO



- Point In Time Recovery needs to rollback transactions from that PIT
  - Restore UNDO at that PIT, and this cannot overwrite CDB UNDO

Flashback PDB
Shared or local UNDO

New: you can flashback a PDB

In SHARED UNDO mode

- Uses an auxiliary instance, as in PDBPITR to restore UNDO
- Exception is CLEAN RESTORE POINT
  - Restore point taken when PDB is closed do not need UNDO

```
SQL> create clean restore point REGTEST1;
create clean restore point REGTEST1
*
ERROR at line 1:
ORA-65025: Pluggable database  is not closed on all instances.
```

In LOCAL UNDO mode

- No need for auxiliary instance
- No need for clean restore points

Flashback PDB
In Single-Tenant

Flashback for Dev/Test databases in Single-Tenant

- At CDB or PDB level?
- Shared UNDO or Local UNDO

It is faster at PDB level

- No instance restart

It is easier at PDB level

- You don't loose AWR, etc

If you need to flashback to a PIT with active transactions

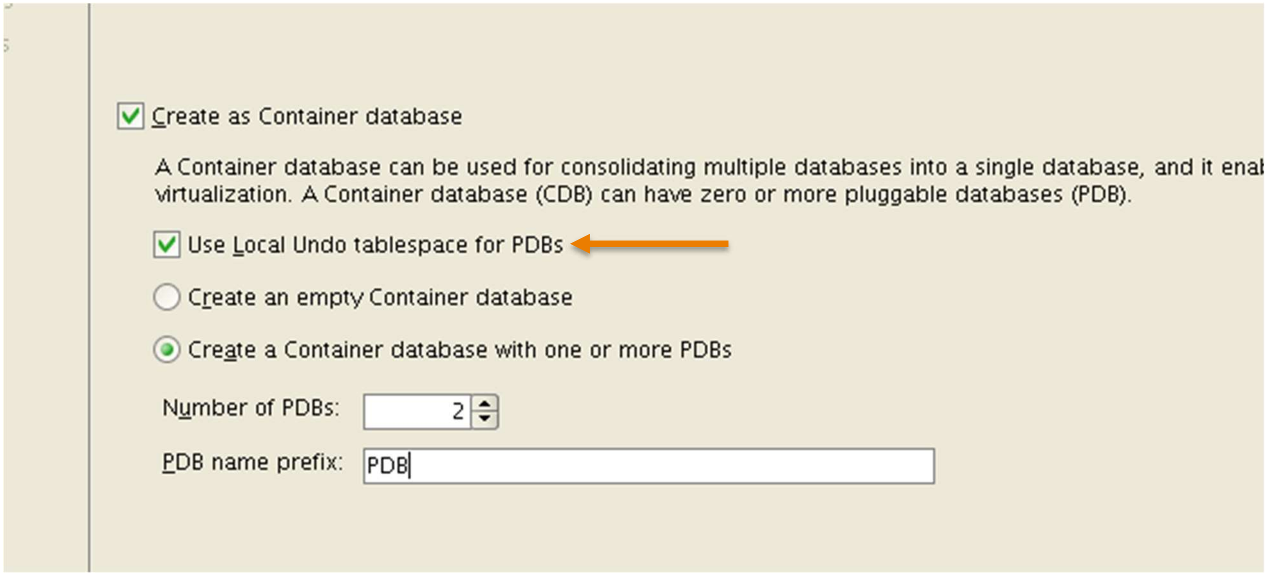
- Easier and faster with local undo

Flashback PDB
Demo

```
SQL> select * from database_properties where property_name like '%UNDO%';

PROPERTY_NAME        PROPERTY_VALUE        DESCRIPTION
-------------        --------------        -----------------------------
LOCAL_UNDO_ENABLED   TRUE                  true if local undo is enabled

SQL> alter pluggable database close;
Pluggable database altered.
SQL> flashback pluggable database to restore point INITIALSTATE;
Flashback complete.

SQL> alter pluggable database PDB open;
alter pluggable database PDB open
*
ERROR at line 1:
ORA-01113: file 11 needs media recovery
ORA-01110: data file 11: '/u02/oradata/CDB1/pdb/undotbs01.dbf'


SQL> alter pluggable database PDB open resetlogs;
Pluggable database altered.
```

Local Undo in 12.2
Local UNDO

Local UNDO is the default in DBCA



☑ Create as Container database

A Container database can be used for consolidating multiple databases into a single database, and it enal
virtualization. A Container database (CDB) can have zero or more pluggable databases (PDB).

☑ Use Local Undo tablespace for PDBs

○ Create an empty Container database

● Create a Container database with one or more PDBs

Number of PDBs: 2

PDB name prefix: PDB

- Recommended, even in single-tenant
- Mandatory for Flashback PDB, PDB clone online and PDB online relocation
- No need for auxiliary instance for PDBPITR (PDBS are self contained)

Local Undo in 12.2
Local UNDO

How to check if you are in LOCAL UNDO mode

```
SQL> select * from database_properties where property_name like '%UNDO%';
PROPERTY_NAME        PROPERTY_VALUE       DESCRIPTION
-------------------- -------------------- -----------------------------------
LOCAL_UNDO_ENABLED   TRUE                 true if local undo is enabled
```

- No row means FALSE…

You can enable LOCAL UNDO at database creation

- New PDBs will create their UNDO tablespace from PDB$SEED

You can change the defaults from PDB$SEED

```
SQL> create bigfile undo tablespace UNDOTBS2 datafile '/u02/oradata/CDB/pdb/undotbs02.dbf'
size 100M autoextend on next 100M maxsize 5G retention guarantee;
Tablespace created.
SQL> alter system set undo_tablespace='UNDOTBS2';
System altered.
SQL> drop tablespace UNDOTBS1 including contents and datafiles;
Tablespace dropped.
```

Local Undo in 12.2
Local UNDO

With LOCAL UNDO each container has an UNDO tablespace

```
RMAN> report schema;
…
File Size(MB) Tablespace           RB segs Datafile Name
---- -------- -------------------- ------- ------------------------
1    700      SYSTEM               YES     /u02/oradata/CDB/system01.dbf
2    210      PDB$SEED:SYSTEM      NO      /u02/oradata/CDB/pdbseed/system01.dbf
3    550      SYSAUX               NO      /u02/oradata/CDB/sysaux01.dbf
4    165      PDB$SEED:SYSAUX      NO      /u02/oradata/CDB/pdbseed/sysaux01.dbf
5    270      UNDOTBS1             YES     /u02/oradata/CDB/undotbs01.dbf
6    225      PDB$SEED:UNDOTBS1    NO      /u02/oradata/CDB/pdbseed/undotbs01.dbf
7    5        USERS                NO      /u02/oradata/CDB/users01.dbf
8    210      PDB:SYSTEM           NO      /u02/oradata/CDB/pdb/system01.dbf
9    185      PDB:SYSAUX           NO      /u02/oradata/CDB/pdb/sysaux01.dbf
10   225      PDB:UNDOTBS1         NO      /u02/oradata/CDB/pdb/undotbs01.dbf
11   5        PDB:USERS            NO      /u02/oradata/CDB/pdb/users01.dbf
```

Local Undo in 12.2
Local UNDO

Changing from SHARED to LOCAL UNDO

```
SQL> alter database local undo on;
alter database local undo on
*
ERROR at line 1:
ORA-65192: database must be in UPGRADE mode for this operation
```

- You must be in UPGRADE mode

```
SQL> shutdown immediate
SQL> startup upgrade
SQL> alter database local undo on;
SQL> shutdown immediate
SQL> startup
```

And create an UNDO tablespace in PDB$SEED (before open PDBs)

```
SQL> alter pluggable database PDB$SEED open read write force;
SQL> alter session set container=PDB$SEED;
SQL> create undo tablespace UNDO datafile size 100M autoextend on next 100M;
SQL> alter pluggable database PDB$SEED close;
SQL> alter pluggable database PDB$SEED open read only;
```

12*c* Single-Tenant
Agenda

1. Multitenant architecture
2. Data movement: pluggable databases
3. Agility: local undo, flashback PDB
4. Fast upgrade: plug to new version
5. Security: lockdown profiles
6. Conclusion: Multitenant is not an option

Conclusion
Move 12*c* non-CDB into Single-Tenant

1. Create an empty CDB (with only CDB$ROOT and PDB$SEED)
2. Open source non-CDB in READ ONLY

You can 'unplug' a non-CDB in 12*c* and plug into CDB

3. describe

```
exec dbms_pdb.describe('/oradata/PDBORCL.xml');
```

4. Shutdown source
5. Plug into CDB

```
create pluggable database PDBORCL using /oradata/PDBORCL.xml';
```

Alternative: clone through database link

3. Create database link to source non-CDB
4.

```
create pluggable database PDBORCL FROM NON$CDB@dblink;
```

5. Shutdown source

In all cases:

9. Run noncdb_to_pdb.sql in the PDB
10. Open the PDB + backup

Conclusion
Move 11*g* into Single-Tenant

Two steps upgrade

1. Upgrade to 12*c* non-CDB
2. Move to Single-Tenant

Or Full Transportable Tablespace

1. Create a 12*c* CDB with one PDB
2. Export from 11.2.0.3 or 11.2.0.4 as Full Transportable Tablespace
3. Import into the PDB

Or any other logical migration technique

- expdp/impdp
- Replication (Golden Gate, Dbvisit replicate)
- ...

Conclusion
Overhead? New skills?

Yes you have 3 containers

- More datafiles
- A PDB$SEED that you don’t need

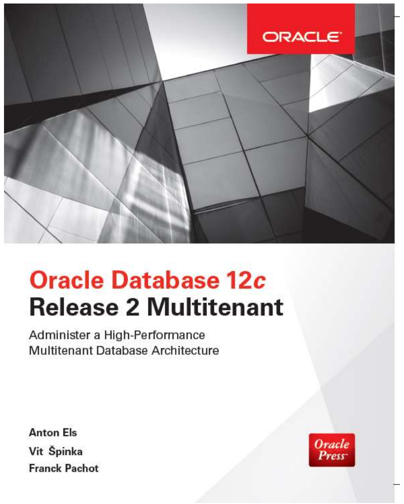
```
RMAN> CONFIGURE BACKUP OPTIMIZATION ON;
```

But… who cares?

- It's just one more (ORACLE_HOME, templates, …)
- Backup optimization on
  - if you don't want to fill backups with read-only tablespaces
  - don't set an external retention shorter than RMAN one

New skills for multitenant?

- you need to connect with services, this is a best practice for years
- you may need to adapt a few scripts but admin is at CDB level



12c Multitenant
Core Message

It's the future

- Non-CDB is deprecated, but still supported
- Yes there are more datafiles, but separating system data and user data is a good idea

It is not so complex

- when doing simple things

Multitenant features are not only for Multitenant Option

- SE and EE without option can benefit from
  - More agility in data movement, upgrades, migrations
  - PITR and flashback,
  - segregation of duty,
  - lockdown features, options and statement usage

Any questions? Please do ask.