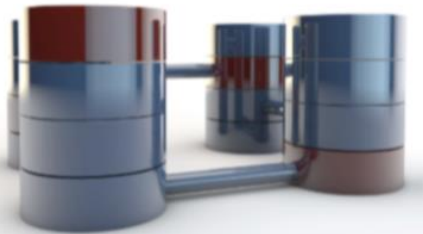


Oracle Multitenant 19c muss das sein?

Datenbank-Tage
28.-29.04.2021

Johannes Ahrends – CarajanDB GmbH



- Oracle Spezialist seit 1992
 - 1992: Presales bei Oracle in Düsseldorf
 - 1999: Projektleiter bei Herrmann & Lenz Services GmbH
 - 2005: Technischer Direktor ADM Presales bei Quest Software GmbH
 - 2011: Geschäftsführer CarajanDB GmbH
- 2011 → Ernennung zum Oracle ACE
- Autor der Bücher:
 - Oracle9i für den DBA, Oracle10g für den DBA, Oracle 11g Release 2 für den DBA
- DOAG Themenverantwortlicher Datenbankadministration, Standard Edition
- DOAG Botschafter 2019
- Hobbies:
 - Drachen steigen lassen (Kiting) draußen wie drinnen (Indoorkiting)
 - Motorradfahren (nur draußen)
 - Singen (überall)



Oracle Multitenant 19c
muss das sein?

NEIN!

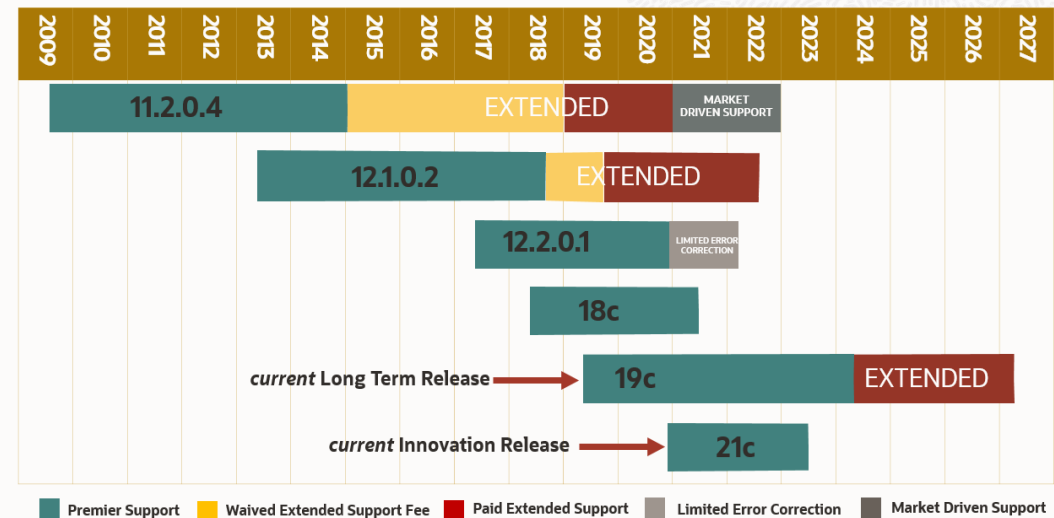
Aber ...

Warum 19c?

Long Term Support Release

- Premier Support (PS) ends April 30, 2024
- Extended Support (ES) fees will be required beginning May 01, 2024 through April 30, 2027
- Error Correction / Patching is available through April 30, 2027 with paid ES.
Without paid ES, patching is only available until April 30, 2024

Database Releases and Support Timelines



Derzeitige Releases

- 11.2.0.4 → Premier (Standard) Support bis 31. Januar 2015
Free Extended Support bis 31. Dezember 2018
Extended Support bis 31. Dezember 2020
- 12.1.0.1 → Premier Support bis 31. August 2016 (kein Extended Support)
- 12.1.0.2 → Premier Support bis 31. Juli 2018
Free Extended Support bis 31. Juli 2019
Extended Support bis 31. Juli 2022
- 12.2.0.1 → Premier Support bis 30. November 2020
Limited Error Correction Support bis 31. März 2022
- 18c → Premier Support bis 30. Juni 2021 (kein Extended Support)
- 19c → Premier Support bis 30. April 2024 ...
- 20c → gibt es nicht!
- 21c → Premier Support bis 30. Juni 2023 (kein Extended Support)

- Innovation Release:
 - Neue Features
 - Kein Extended Support
 - Zuerst benutzt bei 18c
 - 21c und wohl auch 22c
- Terminal Release
 - Letztes Release eines Zyklus
 - 11.2.0.4 und 12.1.0.2
- Long Term Support Release
 - Neue Namensgebung für Terminal Release
 - 19c und wahrscheinlich 23c

Long Term oder Innovation Release

- Long Term Release:
 - *Oracle Database Long Term Releases are ideal for use cases that benefit from less frequent upgrades to newer releases. Long Term Releases offer the highest level of stability and the longest length of error correction support. These releases have 5 years of Premier Support followed by 3 years of Extended Support. When combined with Extended Support, customers typically have almost 4 years to upgrade from one Long Term Release to the next Long Term Release.*
- Innovation Release:
 - *In between Oracle Database Long Term Releases, Oracle delivers Oracle Database Innovation Releases that include many enhancements and new capabilities which will also be included in the next Long Term Release. Innovation Releases are designed to enable customers to continuously use leading-edge technologies to rapidly develop or deploy new applications or augment existing applications. Support for Innovation Releases includes 2 years of Premier Support, but there is no Extended Support. Production workloads can be deployed on Innovation Releases if upgrading within 2 years to a newer release is factored into the deployment plan.*



Warum Multitenant?

- 12.1.0.1 Eine neue Datenbankarchitektur hält Einzug: Multitenant (Besser nicht einsetzen!)
- 12.1.0.2 Long Term Support Release
Multitenant brauchbar, erste Projekte
- 12.2.0.1 Flashback Pluggable Database möglich
viele neue Optionen für PDBs (Proxy PDB, Application Container, ...)
- 18c PDB Lockdown Profiles, Refreshable PDB Switchover
- 19c Drei Pluggable Databases in allen Editionen frei

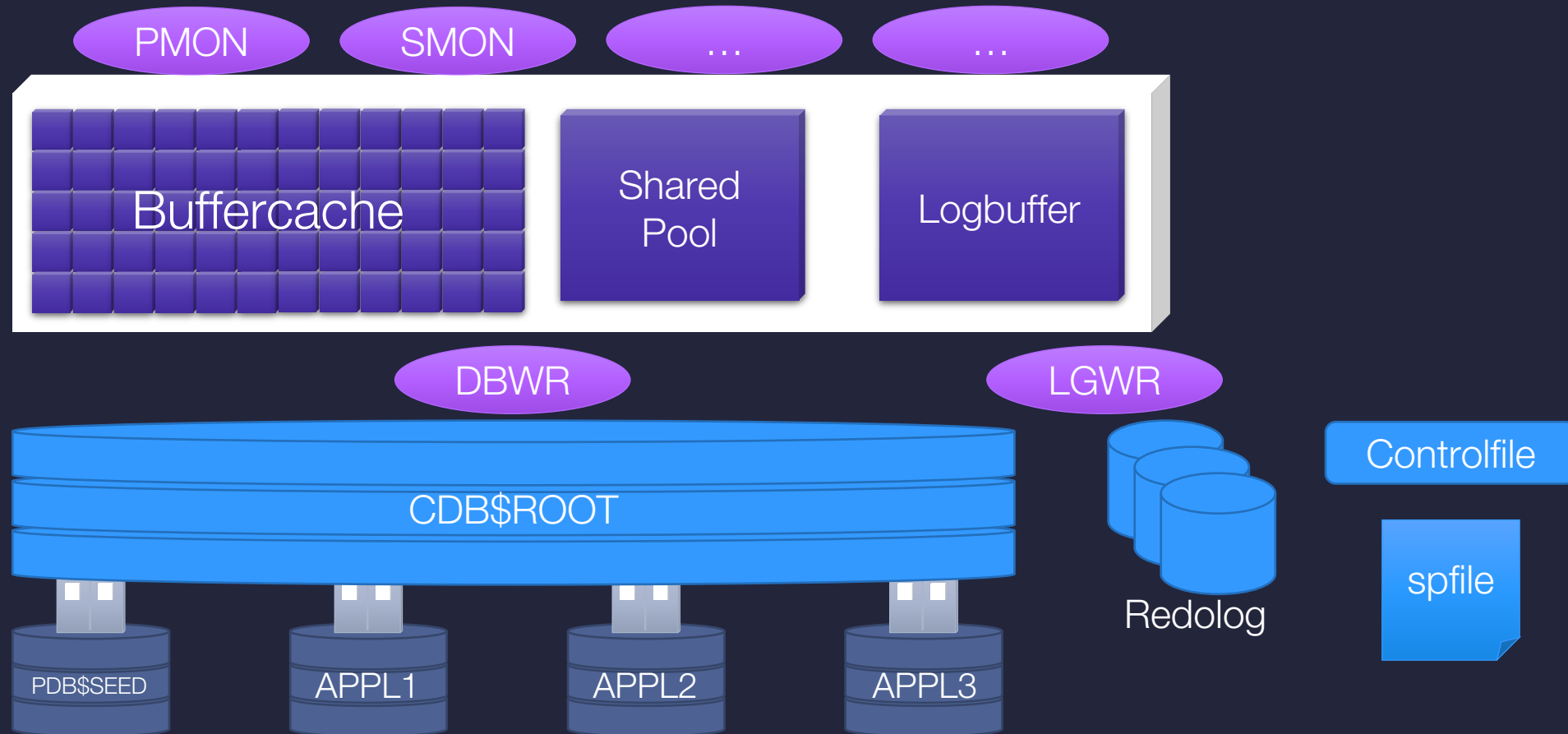
- non-CDB
 - Architektur bis Oracle 11.2
- Multitenant Architektur
 - Alternative Architektur ab Version 12.1.1 für alle Editionen
- Multitenant Option
 - Kostenpflichtig für die Enterprise Edition
 - Bis zu 253 PDBs pro CDB (Exadata ab 12.2 4096 PDBs)
- The non-CDB architecture was deprecated in Oracle Database 12c. It can be desupported and unavailable in a release after Oracle Database 12c Release Oracle recommends use of the CDB architecture. (Oracle Database 12c Release 1 Database Upgrade Guide)
- Starting in Oracle Database 21c, a multitenant container database is the only supported architecture. In previous releases, Oracle supported non-container databases (non-CDBs). (Oracle Database 21c Concepts Guide)

3 PDBs in jeder Edition

Feature / Option / Pack	SE2	EE	EE-ES	DBCS SE	DBCS EE	DBCS EE-HP	DBCS EE-EP	ExaCS	Notes
Oracle Multitenant - # of PDBs	3	252	4096	3	3	4096	4096	4096	The number value in each column indicates the maximum number of pluggable databases (PDBs) that can be created for each offering. - For all offerings, if you are not licensed for Oracle Multitenant, then you may have up to 3 PDBs in a given container database at any time. EE: Extra cost option; if you are licensed for Oracle Multitenant, then you can create up to 252 PDBs. EE-ES: Extra cost option; if you are licensed for Oracle Multitenant, then you can create up to 4096 PDBs. DBCS EE-HP, DBCS EE-EP, and ExaCS: Included option; you can create up to 4096 PDBs.
CDB Fleet Management	N	N	Y	N	Y	Y	Y	Y	
PDB Snapshot Carousel	N	N	Y	N	Y	Y	Y	Y	
Refreshable PDB switchover	N	N	Y	Y	Y	Y	Y	Y	



Quelle: Oracle Database Database Licensing Information User Manual 19c E94254-10 September 2019

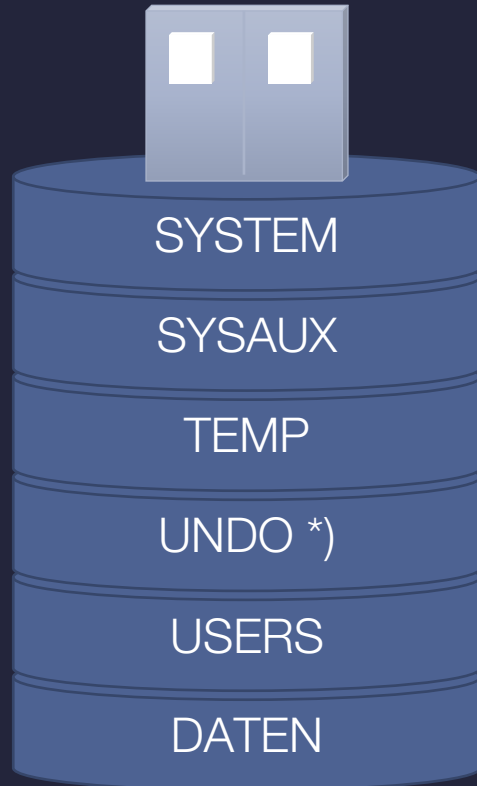


Root Container (CDB)



- Komponenten für den Aufbau der Instanz
 - Controlfiles
 - spfile
 - Connect an die Instanz nur über die CDB möglich
- Verwaltung der Redolog-Dateien
 - Vollständiges Recovery der Datenbank nur über die CDB möglich

Pluggable Database (PDB)



- In sich geschlossenes System
 - Data Dictionary
 - Tablespaces
 - Schemata
 - Objekte
 - Public Objekte
 - Workload Repository
 - Lokales Undo (nicht immer)
 - Parameter

- PDBs sind "virtuelle Datenbanken" innerhalb einer CDB
- Kein Unterschied für Applikationen und Anwender
- Konsolidierung
- "Database as a Service"

Migration – Alle Wege führen nach Rom

- AutoUpgrade
- Upgrade -> Plug-In
- Upgrade -> Remote-Cloning
- "Klassische Methoden"
 - Transportable Tablespaces / Full Transportable Database
 - DataPump
 - Golden Gate

Herausforderungen

- Beispiel:

```
SELECT tablespace_name,  
       trunc(sum(bytes)/1024/1024) Mbytes,  
       trunc(sum(decode(maxbytes,0,bytes,maxbytes)/1024/1024)) maxmbytes  
FROM dba_data_files  
GROUP BY tablespace_name;
```

TABLESPACE_NAME	MBYTES	MAXMBYTES
-----	-----	-----
SYSAUX	870	32767
UNDOTBS1	920	32767
USERS	5	32767
SYSTEM	800	32767

- Ist das richtig? – Es kommt darauf an!
- Es handelt sich hier wahrscheinlich um eine CDB, weil der UNDO-Tablespace angezeigt wird

- Ist das hier besser?

```
SELECT tablespace_name,  
       trunc(sum(bytes)/1024/1024) Mbytes,  
       trunc(sum(decode(maxbytes,0,bytes,maxbytes)/1024/1024)) maxmbytes  
FROM   cdb_data_files  
GROUP BY tablespace_name;
```

TABLESPACE_NAME	MBYTES	MAXMBYTES
USERS	405	38911
UNDOTBS1	920	32767
SYSTEM	1320	98303
SYSAUX	2070	98303

- Die MBYTES scheinen okay aber MAXMBYTES sehen merkwürdig aus!

Vielleicht passt das hier?

```
SELECT con_id, tablespace_name,  
       trunc(sum(bytes)/1024/1024) Mbytes,  
       trunc(sum(decode(maxbytes,0,bytes,maxbytes)/1024/1024)) maxmbytes  
FROM   cdb_data_files  
GROUP BY con_id, tablespace_name  
ORDER BY con_id;
```

CON_ID	TABLESPACE_NAME	MBYTES	MAXMBYTES
1	USERS	5	32767
1	UNDOTBS1	920	32767
1	SYSTEM	800	32767
1	SYSAUX	870	32767
3	USERS	100	5120
3	SYSAUX	600	32767
3	SYSTEM	260	32767
4	USERS	300	1024
4	SYSTEM	260	32767
4	SYSAUX	600	32767

Example 19-7 Showing the Data Files for Each PDB in a CDB

This example queries the `DBA_PDBS` and `CDB_DATA_FILES` views to show the name and location of each data file for all of the PDBs in a CDB, including the PDB seed.

```
COLUMN PDB_ID FORMAT 999
COLUMN PDB_NAME FORMAT A8
COLUMN FILE_ID FORMAT 9999
COLUMN TABLESPACE_NAME FORMAT A10
COLUMN FILE_NAME FORMAT A45
```

```
SELECT p.PDB_ID, p.PDB_NAME, d.FILE_ID, d.TABLESPACE_NAME, d.FILE_NAME
FROM DBA_PDBS p, CDB_DATA_FILES d
WHERE p.PDB_ID = d.CON_ID
ORDER BY p.PDB_ID;
```

Sample output:

PDB_ID	PDB_NAME	FILE_ID	TABLESPACE	FILE_NAME
2	PDB\$SEED	6	USERS	/disk1/oracle/dbs/pdbseed/cdbl_ax.f
2	PDB\$SEED	5	SYSTEM	/disk1/oracle/dbs/pdbseed/cdbl_db.f
3	HRPDB	9	SYSAUX	/disk1/oracle/dbs/hrpdb/hrpdb_ax.f
3	HRPDB	8	SYSTEM	/disk1/oracle/dbs/hrpdb/hrpdb_db.f
3	HRPDB	13	USER	/disk1/oracle/dbs/hrpdb/hrpdb_usr.dbf
4	SALESPDB	15	SYSTEM	/disk1/oracle/dbs/salespdb/salespdb_db.f
4	SALESPDB	16	SYSAUX	/disk1/oracle/dbs/salespdb/salespdb_ax.f
4	SALESPDB	18	USER	/disk1/oracle/dbs/salespdb/salespdb_usr.dbf

Datafile Layout NON-OMF

```
SQL> ALTER SYSTEM SET "_exclude_seed_cdb_view"=FALSE;
```

```
SQL> SELECT con_id, tablespace_name, file_name FROM cdb_data_files ORDER BY 1,2;
```

CON_ID	TABLESPACE_NAME	FILE_NAME
1	SYSAUX	/u02/oradata/FRANZ/sysaux01.dbf
1	SYSTEM	/u02/oradata/FRANZ/system01.dbf
1	UNDOTBS1	/u02/oradata/FRANZ/undotbs01.dbf
1	USERS	/u02/oradata/FRANZ/users01.dbf
2	SYSAUX	/u02/oradata/FRANZ/pdbseed/sysaux01.dbf
2	SYSTEM	/u02/oradata/FRANZ/pdbseed/system01.dbf
2	UNDOTBS1	/u02/oradata/FRANZ/pdbseed/undotbs01.dbf
2	USERS	/u02/oradata/FRANZ/pdbseed/users01.dbf
4	SYSAUX	/u02/oradata/FRANZ/horsti/sysaux01.dbf
4	SYSTEM	/u02/oradata/FRANZ/horsti/system01.dbf
4	UNDOTBS1	/u02/oradata/FRANZ/horsti/undotbs01.dbf
4	USERS	/u02/oradata/FRANZ/horsti/users01.dbf



Myten



New Paradigms for Rapid Patching and Upgrades

The investment of time and effort to patch one multitenant container database results in patching all of its many pluggable databases. To patch a single pluggable database, you simply unplug/plug to a multitenant container database at a different Oracle Database software version.

To upgrade all hosted pluggable databases in a container database, simply upgrade the container database and all hosted pluggable databases are upgraded 'in-place'. To upgrade a single PDB, you simply unplug/plug the pluggable database in to a container database at a higher version and upgrade the pluggable database as described in the [Database Upgrade Guide](#).

<http://www.oracle.com/technetwork/database/multitenant/overview/index.html>

- Die Investition von Zeit und Aufwand für das Patchen einer Multitenant-Containerdatenbank führt dazu, dass all die vielen zugehörigen plug-fähigen Datenbanken gepatcht werden. Um eine einzelne plug-fähige Datenbank zu patchen, trennen Sie einfach mit einer anderen Oracle Database-Softwareversion die Verbindung zu einer Multitenant-Containerdatenbank oder stellen sie her.

Quelle: <https://www.oracle.com/de/database/technologies/multitenant.html>

Autoupgrade

- Java-basierendes Tool für Upgrade
 - Aktuelle Version über MOS Note: 2485457.1 – AutoUpgrade Tool
 - Minimum Version 11.2.0.4
 - Zielversion 12.2, 18c, 19c, 21c
- Infos: <https://mikedietrichde.com/2019/04/29/the-new-autoupgrade-utility-in-oracle-19c/>
- Details: Oracle Dokumentation: Database Upgrade Guide

1. Sample Konfigurationsdatei

```
$ORACLE_HOME/jdk/bin/java -jar ./autoupgrade.jar -create_sample_file config config.cfg
```

- Erstellt die Datei `sample_config.cfg` mit mehreren Beispieleinträgen
- Einschränkungen sind möglich (z.B. full für NON-CDB Upgrades)

2. Konfigurationsdatei

- Dedizierte Version

```
cp sample_config.cfg configJOSEPH.cfg
```

```
cat configJOSEPH.cfg
```

```
upg1.log_dir=/home/oracle/autoupgrade
upg1.sid=JOSEPH
upg1.source_home=/u01/app/oracle/product/12.2.0/dbhome_1
upg1.target_home=/u01/app/oracle/product/19.0/dbhome_1
upg1.start_time=NOW
upg1.upgrade_node=haydn
upg1.run_utlrp=yes
#upg1.timezone_upg=[yes|no]
#upg1.target_version=[12.2|18|19|21]
```

3. Analyse

```
java -jar ./autoupgrade.jar -config configJOSEPH.cfg -mode analyze
```

```
No parameter 'global.autoupg_log_dir' found in config file, using
/u01/app/oracle/cfgtoollogs/autoupgrade
AutoUpgrade tool launched with default options
Processing config file ...
+-----+
| Starting AutoUpgrade execution |
+-----+
1 databases will be analyzed
Type 'help' to list console commands
upg> Job 100 completed
----- Final Summary -----
Number of databases          [ 1 ]

Jobs finished                [1]
Jobs failed                  [0]
Jobs pending                  [0]

Please check the summary report at:
/u01/app/oracle/cfgtoollogs/autoupgrade/cfgtoollogs/upgrade/auto/status/status.html
/u01/app/oracle/cfgtoollogs/autoupgrade/cfgtoollogs/upgrade/auto/status/status.log
```

- Parameter `global.autoupgrade_log_dir` auf das Log Verzeichnis setzen
 - Ansonsten zwei Verzeichnisse mit Log-Dateien

```
cat configJOSEPH.cfg
```

```
global.autoupgrade_log_dir=/home/oracle/autoupgrade  
upg1.log_dir=/home/oracle/autoupgrade  
upg1.sid=JOSEPH  
upg1.source_home=/u01/app/oracle/product/12.2.0/dbhome_1  
upg1.target_home=/u01/app/oracle/product/19.0/dbhome_1  
upg1.start_time=NOW  
upg1.upgrade_node=haydn  
upg1.run_utlrp=yes  
#upg1.timezone_upg=[yes|no]  
#upg1.target_version=[12.2|18|19|21]
```

4. Upgrade

```
java -jar ./autoupgrade.jar -config configJOSEPH.cfg -mode deploy
```

```
AutoUpgrade tool launched with default options
```

```
Processing config file ...
```

```
+-----+
```

```
| Starting AutoUpgrade execution |
```

```
+-----+
```

```
1 databases will be processed
```

```
Type 'help' to list console commands
```

```
upg>
```

- Jeder „run“ hat seine eigenen Log-Verzeichnisse:
- Im Beispiel:
 - 100 → Analyze Lauf
 - 101 → Deployment
- Zwei wichtige Logdateien
 - Übersicht:

```
/home/oracle/autoupgrade/JOSEPH/101/autoupgrade_20210426_user.log
```

- Details:

```
/home/oracle/autoupgrade/JOSEPH/101/dbupgrade/autoupgrade20210426111829joseph.log
```

- Bei Aufruf aus einem Scheduler (z.B. Control-M) Parameter `-noconsole` setzen

```
java -jar ./autoupgrade.jar -config configJOSEPH.cfg -mode deploy -noconsole
```

- Autoupgrade ist Restartable!

```
java -jar ./autoupgrade.jar -config configJOSEPH.cfg -mode deploy
```

```
Previous execution found loading latest data
Total jobs recovered: 1
+-----+
| Starting AutoUpgrade execution |
+-----+
Type 'help' to list console commands
upg>
```

- Außerdem gibt es einen Guaranteed Restore Point (nicht für SE2!)

```
SQL> select name from v$restore_point;
```

```
NAME
```

```
-----
AUTOUPGRADE_9212_JOSEPH122010
```

- Bei Fehlern
 - Backup einspielen (Standard Edition 2)
 - Flashback auf Guaranteed Restore Point

```
SQL> select name from v$restore_point;
```

```
NAME
```

```
-----  
AUTOUPGRADE_9212_JOSEPH122010
```

- Log-Dateien löschen!
 - Deployment merkt sich den Status der Datenbank

Multitenant Migration

Upgrade 12.1 → 19

- Quelle: 12.1.0.2 NON-CDB (DB_NAME → JOTESTK)
- Ziel: 19.10.0 PDB
- Voraussetzung: 19c CDB existiert schon (CDB-NAME → CDB01)
- Quelle: Mike Dietrich
 - <https://mikedietrichde.com/2019/07/25/database-migration-from-non-cdb-to-pdb-clone-via-noncdb-upgrade-convert>

- Erstellen eines Benutzers für die Migration (joupgrade)

```
sqlplus / as sysdba  
SQL> CREATE USER joupgrade IDENTIFIED BY manager;  
SQL> GRANT connect, resource, create pluggable database TO joupgrade;
```

- Erstellen eines PDB manifest

```
SQL> SHUTDOWN IMMEDIATE  
SQL> STARTUP OPEN READ ONLY  
  
SQL> EXEC dbms_pdb.describe('/home/oracle/upgrade/JOTESTK.xml');  
SQL> EXIT;
```

- Kopieren des Manifests auf das Zielsystem

```
scp /home/oracle/upgrade/JOTESTK.xml lxora19:/home/oracle/upgrade/JOTESTK.xml
```

- Erstellen eines TNS-Aliases für die Quelldatenbank (JOTESTK)

```
JOTESTK =  
  (DESCRIPTION =  
    (ADDRESS = (PROTOCOL = TCP) (HOST = lxora12) (PORT = 151))  
    (CONNECT_DATA =  
      (SERVER = DEDICATED)  
      (SERVICE_NAME = JOTESTK)  
    )  
  )
```

- Überprüfen der Kompatibilität

```
DECLARE
    compatible CONSTANT VARCHAR2(3) :=
        CASE DBMS_PDB.CHECK_PLUG_COMPATIBILITY(
            pdb_descr_file => '/home/oracle/upgrade/JOTESTK.xml',
            pdb_name        => 'JOTESTK')
            WHEN TRUE THEN 'YES' ELSE 'NO'
        END;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Is the future PDB compatible? ==> ' || compatible);
END;
/

==> Jede Menge Fehler und Warnings.
Wichtig: CTX muss in der CDB installiert sein
1 JOTESTK  ERROR      Database option CONTEXT mismatch: PDB installed version
12.1.0.2.0. CDB installed version NULL. PENDING
```

- Anlegen eines Datenbank-Links

```
SQL> CREATE DATABASE LINK jotestk  
      CONNECT TO joupgrade  
      IDENTIFIED BY manager USING 'JOTESTK';
```

- Local Undo Ausschalten (wegen 12.1 Kompatibilität)

```
SQL> SHUTDOWN IMMEDIATE  
SQL> STARTUP UPGRADE  
SQL> ALTER DATABASE LOCAL UNDO OFF;  
SQL> SHUTDOWN IMMEDIATE  
SQL> STARTUP
```

- Ansonsten beim Versuch, die PDB anzulegen:

```
CREATE PLUGGABLE DATABASE jotestk FROM non$cdb@jotestk  
*  
ERROR at line 1:  
ORA-65353: The undo tablespace is missing from the XML metadata file.
```

- Kopieren der Datenbank über Datenbank Link

```
SQL> CREATE PLUGGABLE DATABASE jotestk FROM non$cdb@jotestk;  
...
```

- Öffnen der Datenbank

```
SQL> ALTER PLUGGABLE DATABASE jtestk OPEN;
```

```
Warning: PDB altered with errors.
```

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	READ ONLY	NO
3	JTEST1	READ WRITE	NO
4	JTESTK	MIGRATE	YES

- Upgrade der Datenbank nach Version 19

```
mkdir -p /home/oracle/jo/logs
dbupgrade -l /home/oracle/jo/logs -c "JOTESTK"
```

```
...
```

```
Grand Total Upgrade Time:      [0d:0h:24m:56s]
```

```
sqlplus / as sysdba
```

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	READ ONLY	NO
3	JOTEST1	READ WRITE	NO
4	JOTESTK	MOUNTED	

- Konvertieren der Datenbank in eine PDB

```
SQL> ALTER SESSION SET container=jotestk;

SQL> SET timing ON
SQL> SET serveroutput ON
SQL> SET echo ON
SQL> SET termout ON
SQL> spool /home/oracle/jo/logs/jotestk_noncdb2pdb.log
SQL> @?/rdbms/admin/noncdb_to_pdb.sql
...
Dauer: 1 Stunde !!!
```

- Öffnen der neuen PDB

```
SQL> ALTER PLUGGABLE DATABASE OPEN;
```

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	READ ONLY	NO
3	JOTEST1	READ WRITE	NO
4	JOTESTK	READ WRITE	NO

```
SQL> ALTER PLUGGABLE DATABASE SAVE STATE;
```

- Local Undo einschalten

```
sqlplus / as sysdba
```

```
SQL> SHUTDOWN IMMEDIATE
```

```
SQL> STARTUP UPGRADE
```

```
SQL> ALTER DATABASE LOCAL UNDO ON;
```

```
SQL> SHUTDOWN IMMEDIATE
```

```
SQL> STARTUP
```

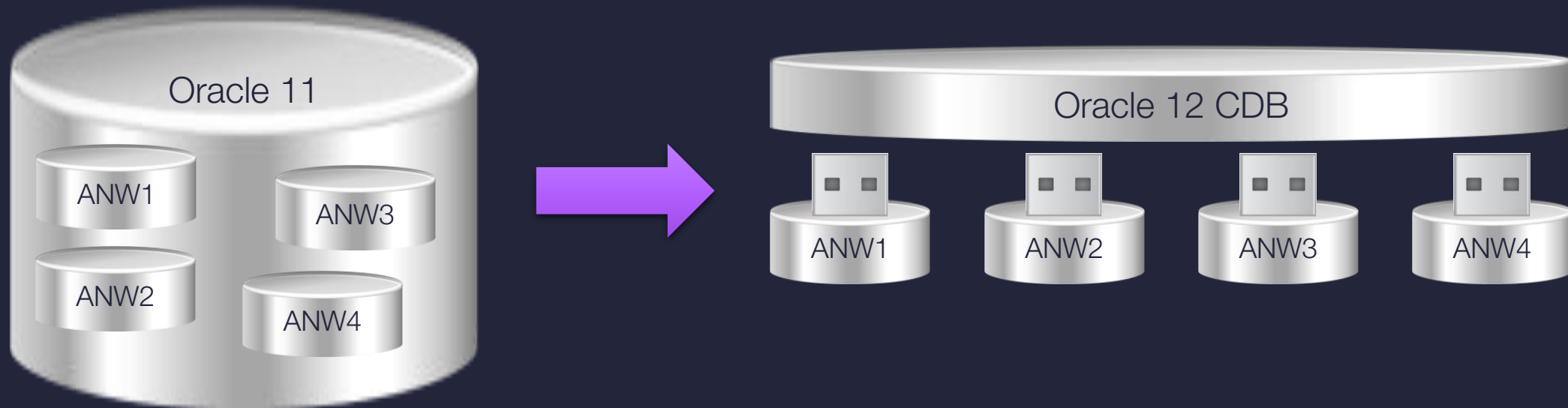
Autoupgrade

- NON-CDB to PDB Migration
- 19c CDB muss schon existieren

Migration in der Praxis

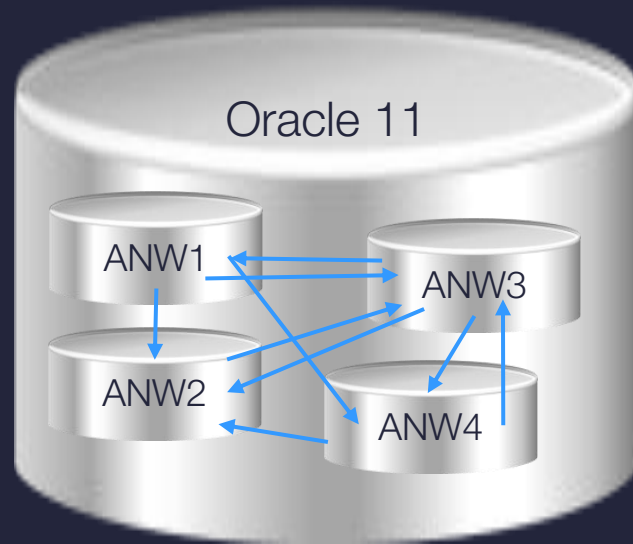
- Identifizieren des Projektes → Absprache mit Projekt-Verantwortlichen
 - Wie heißt die Anwendung?
 - Wie greift der Client darauf zu (Java, App-Server, Direkt, ...)?
 - Gibt es Restriktionen?
- Erstellen einer PDB für die Anwendung
- Data Pump Export des/der Schemata
- Data Pump Import des/der Schemata
- Logdaten der Migration müssen 10 Jahre aufbewahrt werden

- Quelle: 8 Datenbanken mit ca. 120 Anwendungen (Schemakonsolidierung)
- Ziel: 7 Datenbanken mit ca. 120 Pluggable Databases



Was Anwendungen wollen

- Anwendung 1: Wir brauchen Oracle 12.1
- Anwendung 2: Wir brauchen 12.2 aber nicht Oracle 19
- Anwendung 3: Wir brauchen unbedingt WE8ISO8859P15
- Anwendung 4: Wir brauchen unbedingt Unicode
- Anwendung 5: Wir benötigen einen Oracle 10 Client (nicht bewusst!)



- ANW1 greift auf Dateien von ANW3 zu
- ANW3 benötigt Daten von ANW2
- ...
- Ein User ändert Daten in ANW1 und ANW4
- Ein User liest Daten aus ANW2 und ANW4

- Schemanamen enthalten den Stage (z.B. ANW1_T, ANW1_V, ANW1_P)
 - Oracle kennt kein Rename Schema (offiziell)
- ➔ Kein Cloning der PDB über Stagegrenzen möglich

- Anwendung arbeitet mit SID statt Servicennamen
 - Ist bei PDBs nicht möglich
- Anwendung benutzt Serveridentifikation (OPS\$-Account)
 - OPS\$ muss in der CDB angelegt werden (C##OPS\$)
 - On Logon Trigger für Redirect auf die PDB
- Anwendung hat nur eine maximale Zeichenlänge für Connect (hier 30 Zeichen)
 - Anwendung kann nicht hochverfügbar sein (Alternative Adresse)

- Oracle 11g:
Zugriffe auf andere Schemata / Anwendungen durch GRANT der Privilegien
- Oracle 12c mit Multitenant:
Zugriffe auf andere PDBs nur über Database Link möglich
 - Database Link erlauben keine Synonyme
 - Erstellen von Views für Remote Datenbank Objekte
 - Synonyme auf Views
 - Vollqualifizierter Zugriff nicht mehr möglich (Dummyuser erforderlich)

- Multitenant ist die Zukunft
- Es gibt keinen Grund (außer Zeit) es nicht einzusetzen
- RAC funktioniert
- Data Guard funktioniert (bis auf PDB Cloning und TEMP Tablespaces)
- RMAN funktioniert (gerne auch PDB als Catalog)
- Tolle neue Möglichkeiten
 - Refreshable PDB
 - (Snapshot) Cloning

Oracle Multitenant 19c
muss das sein?

- Experten mit über 30 Jahren Datenbank Erfahrung
- Spezialisten für
 - Datenbank Administration (Oracle und PostgreSQL)
 - Hochverfügbarkeit (RAC, Data Guard, Replication, etc.)
 - Migrationen (Unicode, PostgreSQL)
 - Performance Optimierung
 - Monitoring (OEM, Foglight, CheckMK, PEM)
- Fernwartung
- Schulung und Workshops
 - PostgreSQL
 - Oracle Multitenant
 - Toad



- E-Mail: johannes.ahrends@carajandb.com
- Homepage: www.carajandb.com
- Adresse:
 - CarajanDB GmbH
Siemensstraße 25
50374 Erftstadt
- Telefon:
 - +49 (22 35) 1 70 91 84
 - +49 (1 70) 4 05 69 36
- Twitter: carajandb
- Facebook: johannes.ahrends
- Blogs:
 - blog.carajandb.com
 - www.toadworld.com