SOUG swiss oracle user group

Multitenant Database

Johannes Ahrends
CarajanDB GmbH

ORACLE ACE

CarajanDB

- Experten mit über 25 Jahren Oracle Erfahrung
- Firmensitz in Erftstadt bei Köln
- Spezialisten für
  - Oracle Datenbank Administration
  - Hochverfügbarkeit (RAC, Data Guard, Failsafe, etc.)
  - Einsatz der Oracle Standard Edition
  - Oracle Migrationen (HW, Unicode, Standard Edition)
  - Replikation
  - Performance Tuning
  - Datenbank Cloning (Delphix, Actifio, CloneDB)
- Fernwartung
- Schulung und Workshops (Oracle, Toad)

… über mich

- 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
- Hobbies:
  - Drachen steigen lassen (Kiting) draußen wie drinnen (Indoorkiting)
  - Motorradfahren (nur draußen)

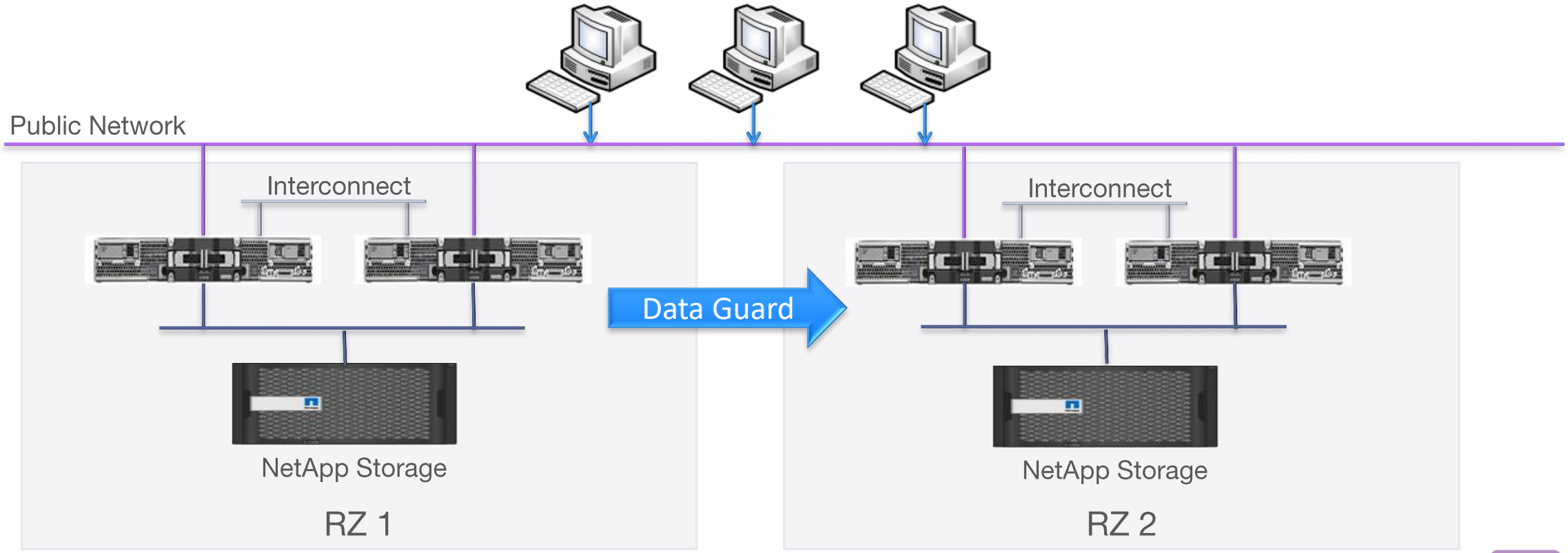
Kontakt

- 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
- Blog: blog.carajandb.com

SOUG swiss oracle user group

Projekt

Architektur



Architekturdetails (1)

- Zwei Rechenzentren identisch ausgestattet
- Drei Netzwerke (Public, Private / Interconnect, Storage)
- 4 Stages:
  - Maintenance → Nur für die DBAs
  - Test → Entwicklung, Test (weniger Hardwarekapazitäten)
  - Vorproduktion → QA (identisch mit Produktion)
  - Produktion
- NetApp Storage
  - Direct NFS
  - Snapshot Backup und Cloning

Oracle Details

- 5 Datenbanken
  - 3 Competence Center
  - 1 Single Tenant
  - 1 Unicode
- Rechenzentren identisch ausgestattet
  - Keine Unterscheidung zwischen Primary und Standby
  - Statt dessen db_unique_name ➔ _RZ1 bzw. _RZ2
- Oracle 12.1.0.2
  - Cloud First hat das Projekt drastisch verzögert!

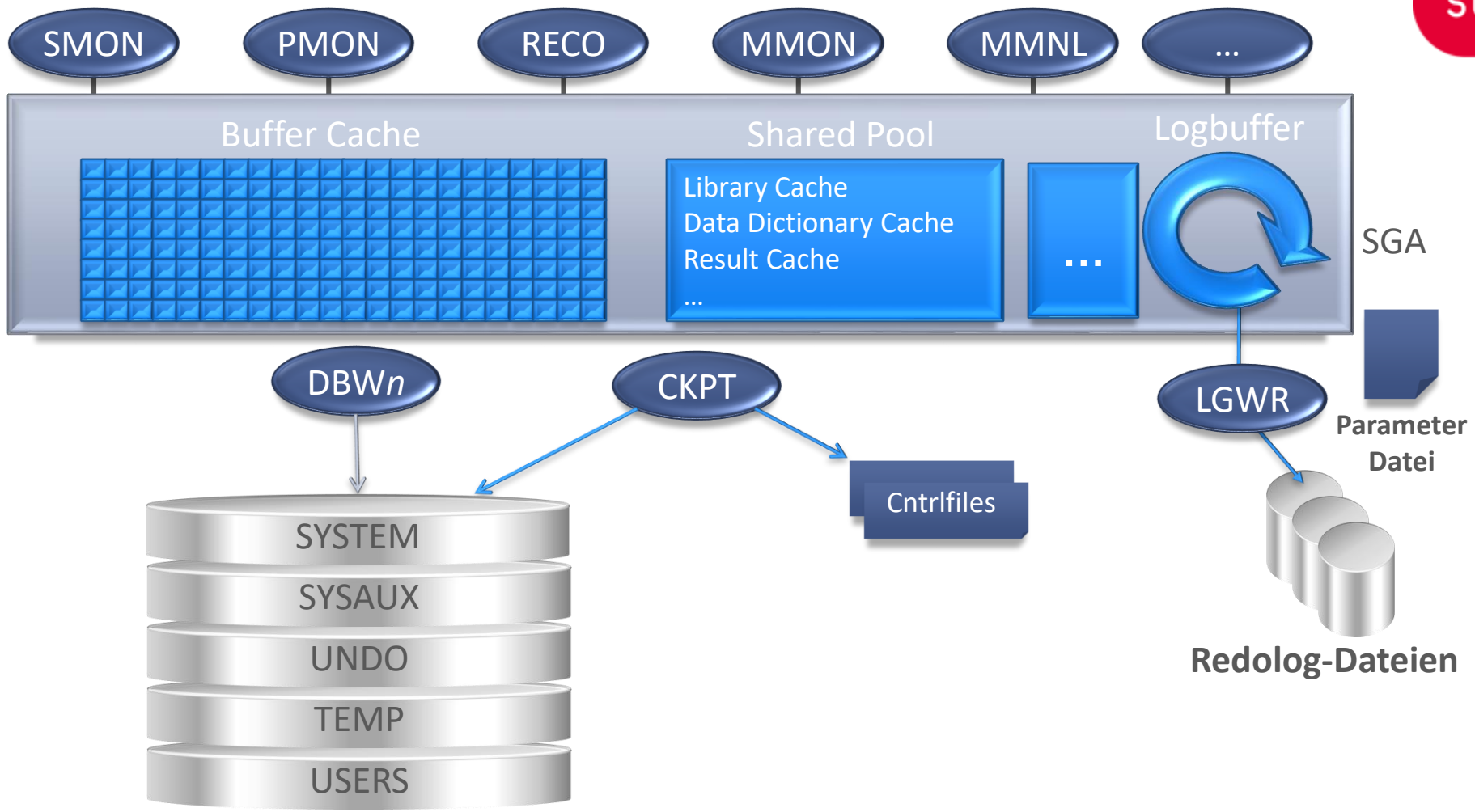
SOUG swiss oracle user group

Multitenant Database - Überblick

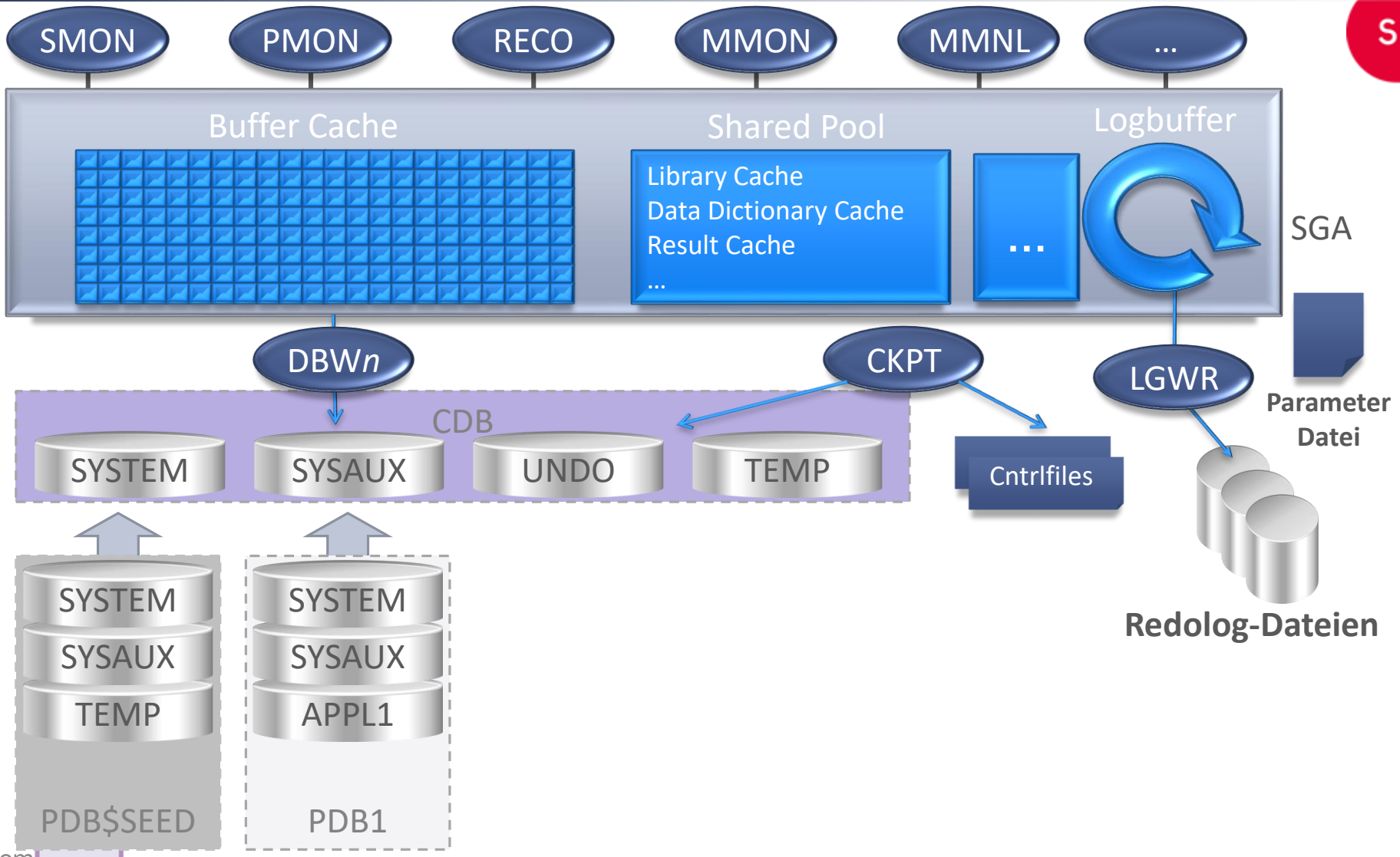
Ressentiments

- Geht nicht in unserer Umgebung!
- Da kann ja eine PDB alle Ressourcen verbrauchen
- Wir haben unterschiedliche Zeichensätze in unseren Datenbanken
- Wir müssen die Datenbank im Fehlerfall (Batch) zurücksetzen können

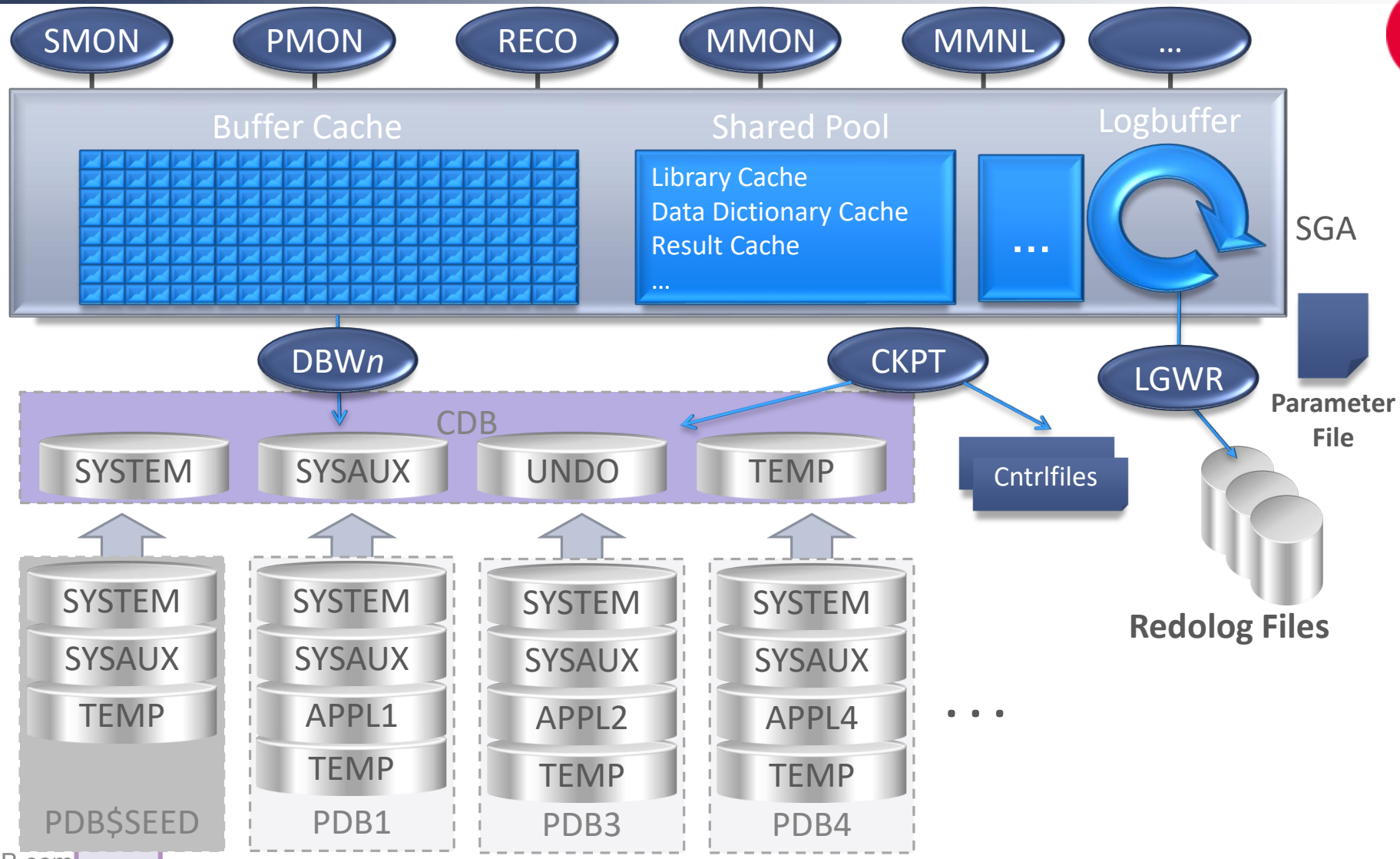
„Klassische“ Datenbank



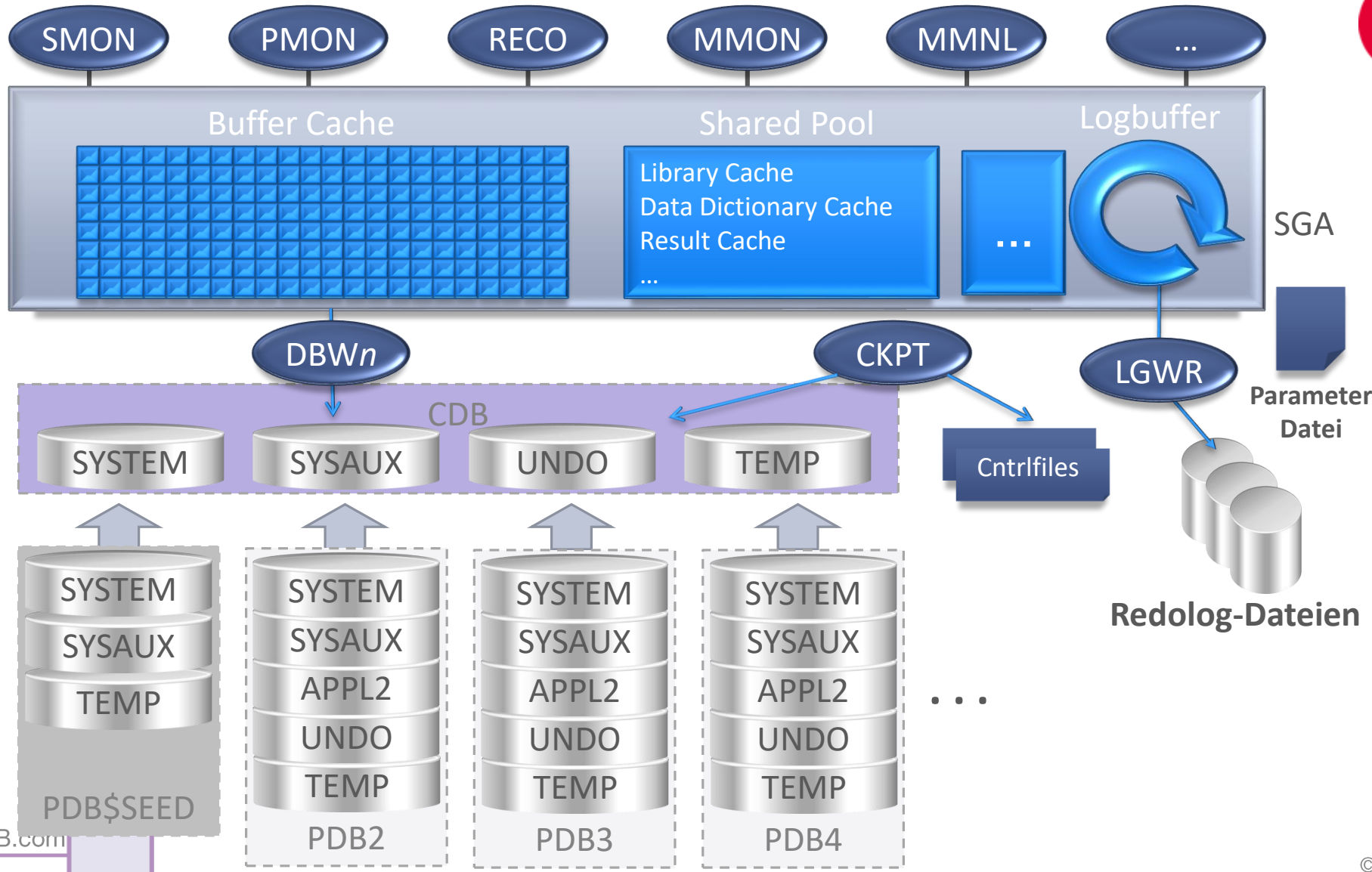
Singletenant Database



Multitenant Database Architecture



Multitenant Database 12.2



Multitenant Database

- Kostenpflichtige Option für die Enterprise Edition
- Single-Tenant Database auch für Standard Edition 2

"The non-CDB architecture is deprecated in Oracle Database 12c, and may be desupported and unavailable in a later Oracle Database release. Oracle recommends use of the CDB architecture." (Oracle 12c Database Upgrade Guide, Kapitel 8.1.1)

SOUG swiss oracle user group

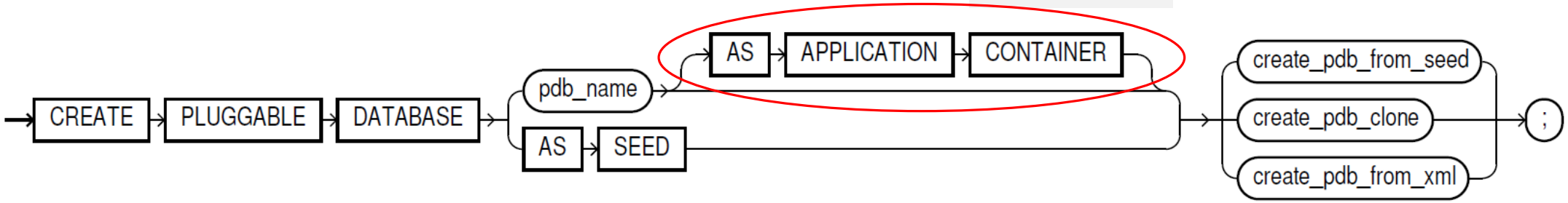
Create Pluggable Database

Erstellen einer Pluggable Database

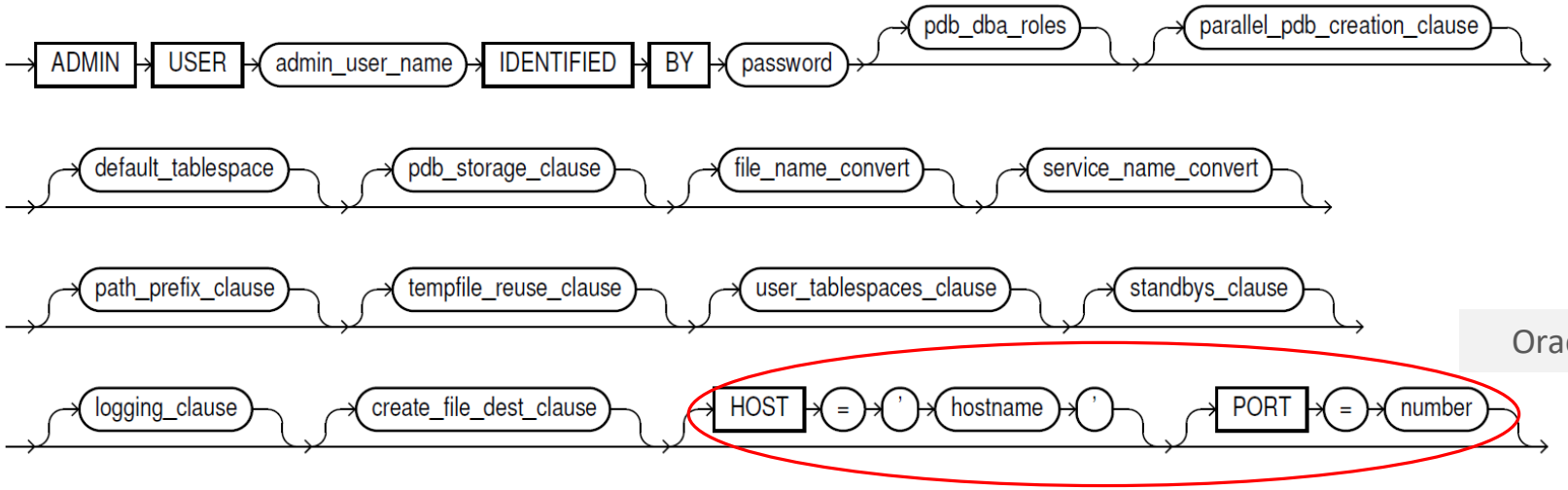
- CREATE PLUGGABLE DATABASE

create_pluggable_database::=

Oracle 12.2



create_pdb_from_seed::=



Oracle 12.2

Wovon erstellen?

- from SEED (Default)
  - Template PDB in jeder CDB
  - Ist Read-Only geöffnet und kann nicht (offiziell) geändert werden)
  - Bei OMF nichts weiter zu tun
  - Bei Non-OMF → FILE_NAME_CONVERT Parameter verwenden
- from Clone
  - Aus einer existierenden PDB
  - Identisch zu Seed, nur dass es „meine eigene“ PDB ist
  - Ab 12.2 auch bei geöffneter PDB möglich
  - Clone kann auch remote (CDB, Non-CDB) sein, dann DB-Link
- from XML (Manifest)
  - Datendateien werden kopiert oder angehängt
  - Konfiguration in der XML-Datei gespeichert
  - Z.B. bei NON-CDB nach PDB Migration

Erstellen einer PDB aus Seed

- Erstellen einer PDB aus der Seed-Datenbank:

```
SQL> CREATE PLUGGABLE DATABASE johann1 ADMIN USER joadmin
             IDENTIFIED BY manager
             FILE_NAME_CONVERT=('/u02/oradata/CJOHANN/pdbseed',
                                '/u02/oradata/CJOHANN/johann1');
```

- Einfacher

```
SQL> CREATE PLUGGABLE DATABASE johann1 ADMIN USER joadmin
             IDENTIFIED BY manager
             FILE_NAME_CONVERT=('pdbseed','johann1');
```

- Mit Privilegien und OMF

```
SQL> CREATE PLUGGABLE DATABASE johann1 ADMIN USER joadmin
             IDENTIFIED BY manager ROLES=(connect);
```

SOUG swiss oracle user group

Oracle Real Application Clusters

Erstellen von Pluggable Databases

CARAJANDB

SOUG swiss oracle user group

- Kein Unterschied zur Single Instance da
  - Kein eigener UNDO-Tablespace bei 12.1
  - N eigene UNDO-Tablespace bei 12.2 local Undo
  - Keine eigenen Redolog Files
  - Keine RAC spezifischen Oracle Parameter

Views für RAC

- Generell alle PDB spezifischen Views auf als Global Views vorhanden

```
SQL> SELECT * FROM gv$pdbs;

 INST_ID     CON_ID       DBID    CON_UID   GUID      NAME       OPEN_MODE  RES OPEN_TIME                 CREATE_SCN TOTAL_SIZE
---------- ---------- ---------- ---------- --------- ---------- ---------- --- -------------------------- ---------- ----------
         1          2 4074185444 4074185444 EB33…38DD PDB$SEED   READ ONLY  NO  21-JAN-14 04.45.44.869 PM    1720773  283115520
         1          3  513120842  513120842 F07E…19A0 PAUL1      MOUNTED        21-JAN-14 05.21.34.114 PM    3495540          0
         1          4  308564365  308564365 F07E…19A0 PAUL2      MOUNTED        21-JAN-14 05.28.03.393 PM    3496050          0
         2          2 4074185444 4074185444 EB33…38DD PDB$SEED   READ ONLY  NO  21-JAN-14 04.45.46.547 PM    1720773  283115520
         2          3  513120842  513120842 F07E…19A0 PAUL1      MOUNTED        22-JAN-14 08.28.11.922 AM    3495540          0
         2          4  308564365  308564365 F07E…19A0 PAUL2      READ WRITE NO  21-JAN-14 05.30.16.828 PM    3496050  283115520
```

Views für RAC

- Besser mit Instanzname

```
SQL> SELECT i.instance_name, p.name AS pdb_name, P.open_mode, P.open_time
       FROM gv$instance i, gv$pdbs P
      WHERE i.inst_id = p.inst_id;

INSTANCE_NAME    PDB_NAME   OPEN_MODE  OPEN_TIME
---------------- ---------- ---------- --------------------------
CPAUL_1          PDB$SEED   READ ONLY  21-JAN-14 04.45.44.869 PM
CPAUL_1          PAUL1      MOUNTED    21-JAN-14 05.21.34.114 PM
CPAUL_1          PAUL2      MOUNTED    21-JAN-14 05.28.03.393 PM
CPAUL_2          PDB$SEED   READ ONLY  21-JAN-14 04.45.46.547 PM
CPAUL_2          PAUL1      MOUNTED    22-JAN-14 08.28.11.922 AM
CPAUL_2          PAUL2      READ WRITE 21-JAN-14 05.30.16.828 PM
```

Öffnen von Pluggable Databases

- STARTUP Befehl funktioniert nicht bei RAC

```
SQL> STARTUP PLUGGABLE DATABASE <ALL> OPEN;
```

  - Öffnet die Pluggable Database nur für die aktuelle Instanz
- Bei RAC folgende Syntax:

```
SQL> ALTER PLUGGABLE DATABASE <pdbname> OPEN INSTANCES=ALL;
SQL> ALTER PLUGGABLE DATABASE <pdbname> OPEN INSTANCES=('INST1','INST2');
SQL> ALTER PLUGGABLE DATABASE <pdbname> OPEN EXCEPT INSTANCES='INST1','INST2';
```

- Tipp: Generell „ALTER PLUGGABLE DATABASE“ verwenden

Alertfile Einträge

- Primäre Instanz (This instance was first to open …)

```
Thu Jan 23 09:12:53 2014
This instance was first to open pluggable database PAUL1 (container=3)
Thu Jan 23 09:13:17 2014
Setting Resource Manager plan SCHEDULER[0x420C]:DEFAULT_MAINTENANCE_PLAN via scheduler
window
Thu Jan 23 09:13:32 2014
Errors in file /u01/app/oracle/diag/rdbms/cpaul/CPAUL_2/trace/CPAUL_2_m000_4697.trc:
ORA-01220: file based sort illegal before database is open
Thu Jan 23 09:13:34 2014
Opening pdb PAUL1 (3) with Resource Manager plan: DEFAULT_MAINTENANCE_PLAN
Thu Jan 23 09:13:37 2014
Auto-setting DEFAULT_CDB_PLAN resource plan
Starting background process VKRM
Thu Jan 23 09:13:37 2014
VKRM started with pid=98, OS id=4732
Thu Jan 23 09:13:41 2014
Pluggable database PAUL1 opened read write
```

Alertfile Einträge

- Zweite Instanz

```
ALTER PLUGGABLE DATABASE paul1 OPEN INSTANCES=ALL
Thu Jan 23 09:13:43 2014
Opening pdb PAUL1 (3) with Resource Manager plan: DEFAULT_MAINTENANCE_PLAN
Thu Jan 23 09:13:45 2014
Auto-setting DEFAULT_CDB_PLAN resource plan
Starting background process VKRM
Thu Jan 23 09:13:46 2014
VKRM started with pid=99, OS id=6845
Thu Jan 23 09:13:54 2014
Pluggable database PAUL1 opened read write
Completed: alter pluggable database paul1 open instances=all
```

Services

- Failover Service (Policy Managed Database)

```
% srvctl add service -db CPAUL -service APPPAUL1 -serverpool beatlespool
        -pdb PAUL1 -failovertype SELECT -failovermethod BASIC
% srvctl start service -db CPAUL -service APPPAUL1
```

```
INSTANCE_NAME    NAME        OPEN_MODE  OPEN_TIME
---------------- ----------- ---------- ----------------------------------------
CPAUL1           PAUL1       READ WRITE 23-JAN-14 10.29.16.432 AM
CPAUL1           PAUL2       MOUNTED
CPAUL1           PDB$SEED    READ ONLY  23-JAN-14 10.26.10.982 AM
CPAUL2           PAUL1       READ WRITE 23-JAN-14 10.25.16.366 AM
CPAUL2           PAUL2       MOUNTED
CPAUL2           PDB$SEED    READ ONLY  23-JAN-14 10.17.26.568 AM
```

Services

- Failover Service (Administrator Managed Database)

```
% srvctl add service -db CPAUL -service APPPAUL1 -preferred CPAUL1 -available CPAUL2
        -pdb PAUL1 -tafpolicy BASIC -failovertype SELECT -failovermethod BASIC
% srvctl start service -db CPAUL -service APPPAUL1
```

```
INSTANCE_NAME    NAME       OPEN_MODE  OPEN_TIME
---------------- ---------- ---------- ----------------------------------------
CPAUL1           PAUL1      READ WRITE 23-JAN-14 10.33.19.229 AM
CPAUL1           PAUL2      MOUNTED
CPAUL1           PDB$SEED   READ ONLY  23-JAN-14 10.32.14.237 AM
CPAUL2           PAUL1      MOUNTED
CPAUL2           PAUL2      MOUNTED
CPAUL2           PDB$SEED   READ ONLY  23-JAN-14 10.32.14.241 AM
```

Services

- tnsnames-Eintrag

```
APPPAUL1 =
  (DESCRIPTION =
       (TRANSPORT_CONNECT_TIMEOUT=3)
       (FAILOVER=ON)
     (ADDRESS = (PROTOCOL = TCP)(HOST = beatles)(PORT = 1521)
    )
    (CONNECT_DATA =
      (SERVICE_NAME = APPPAUL1.carajandb.intra))
  )
```

Starten von PDBs auf nur einer Instanz

- Vorteil: Die User können sich in jedem Fall nur auf einer Instanz anmelden
  - Keine „fehlerhaften“ Anmeldungen
- Nachteil: Failover funktioniert nicht!
  - PDB wird automatisch auf der „anderen“ Instanz geöffnet
  - Trotzdem Fehlermeldung

```
ERROR at line 1:
ORA-03113: end-of-file on communication channel
```

  - Bei Neuanmeldung

```
ERROR:
ORA-12514: TNS:listener does not currently know of service requested in connect
descriptor
```

  - Neuanmeldung beim Listener scheint zu lange zu dauern

Relocate Service

- Zurückschwenken der PDB über den Service

```
srvctl relocate service -db CPAUL -service APPPAUL1 -newinst CPAUL1 -oldinst CPAUL2
```

- PDB wird nicht automatisch geschlossen, wenn der Service gestoppt wird


```
srvctl relocate service -db CPAUL -service APPPAUL1 -newinst CPAUL1 -oldinst CPAUL2
```

PDB auf RAC – wenige Knoten

- Pluggable Database auf allen Instanzen starten
- Zusätzlichen Service für die Anwendung definieren
- Vorteil:
  - Kein Relocate notwendig
  - Gleiches Failover Verhalten wie bei NON-CDB
- Nachteil:
  - Fehlerhafter Connect der Anwendung möglich (das war aber immer schon so!)

PDB auf RAC – mehrere Knoten

- CDB auf allen Knoten gestartet
- PDB auf zwei oder wenigen Knoten gestartet
- Öffnen auf weiteren Knoten jederzeit möglich (auch automatisch)
- Relocate auf andere Knoten möglich

Besonderheit Lokales UNDO

- Oracle 12.1 → UNDO Tablespace nur für die CDB
- Oracle 12.2 → Optional lokales UNDO in der PDB

```
SQL> SELECT tablespace_name, file_name FROM cdb_data_files WHERE con_id=3;

TABLESPACE_NAME      FILE_NAME
-------------------- ------------------------------------------------------------
UNDOTBS1             /u03/oradata/TC01/pdbshared/JOHANNES/undotbs01.dbf
SYSAUX               /u03/oradata/TC01/pdbshared/JOHANNES/sysaux01.dbf
UNDO_2               /u03/oradata/TC01/pdbshared/JOHANNES/johannes_i2_undo.dbf
SYSTEM               /u03/oradata/TC01/pdbshared/JOHANNES/system01.dbf
USERS                /u03/oradata/TC01/pdbshared/JOHANNES/users01.dbf
```

→ UNDO Tablespace 1 = UNDOTBS1

→ UNDO Tablespace 2 = UNDO_2

→ UNDO Datafile 2 = <PDBNAME>_i2_undo.dbf (ohne 02, etc)

Projekt - Multitenant Database & RAC

- Optionen
  - Starten / Öffnen einer PDB auf einem Knoten
    - Beim Ausfall des Knotens wird die PDB auf dem anderen Knoten durch den zugehörigen Service gestartet
    - Nachteil: das Öffnen dauert eine bestimmte Zeit
    - Es erfolgt kein Schließen der PDB, wenn der Service zurückschwenkt
  - Starten / Öffnen der PDB auf allen Knoten
    - Anwendung kann gegen beide Instanzen betrieben werden
    - Einschränkung durch Anwendungsspezifische Services
- Lösung:
  - PDBs werden auf beiden Knoten geöffnet
  - Jede Anwendung erhält einen eigenen Service

➔ Kein Unterschied zu „klassischer / Non-CDB“ Datenbank

SOUG swiss oracle user group

Herausforderungen OMF

Oracle Managed Files

- Auf regulärer Disk und im ASM
  - Pluggable Database Files werden NICHT in dem Verzeichnis des Names der PDB eingetragen sondern im Verzeichnis mit der GUID
  - Ausnahme: PDB$SEED im Verzeichnis der CDB

```
    PDB_ID PDB_NAME          DBID GUID
---------- ---------- ---------- --------------------------------
         3 ANTON1     2660983207 EF89F348D7051BF3E043281E10ACCCD7
         2 PDB$SEED   4078985758 EF899360B02E192DE043281E10ACBA7D
         4 ANTON2     2965167653 EF8AC4B702D10788E043281E10ACE619

SQL> !ls -l /u02/oradata/CANTON
total 20
drwxr-x--- 2 oracle oinstall 4096 Jan  9 13:13 controlfile
drwxr-x--- 2 oracle oinstall 4096 Jan  9 13:16 datafile
drwxr-x--- 3 oracle oinstall 4096 Jan  9 13:43 EF89F348D7051BF3E043281E10ACCCD7
drwxr-x--- 3 oracle oinstall 4096 Jan  9 14:41 EF8AC4B702D10788E043281E10ACE619
drwxr-x--- 2 oracle oinstall 4096 Jan  9 13:13 onlinelog
```

Oracle Managed Files - SEED

- Standard: Im Verzeichnis der CDB:

```
oracle@wader[HANNES]% pwd
/u02/oradata/HANNES/datafile

-rw-r----- 1 oracle oinstall 723525632 Sep 27 17:50 o1_mf_sysaux_9x4qwrl1_.dbf
-rw-r----- 1 oracle oinstall 608182272 Jul 25 16:15 o1_mf_sysaux_9x4r6jxy_.dbf
-rw-r----- 1 oracle oinstall 838868992 Sep 27 17:49 o1_mf_system_9x4qxvot_.dbf
-rw-r----- 1 oracle oinstall 262152192 Jul 25 16:15 o1_mf_system_9x4r6jy1_.dbf
-rw-r----- 1 oracle oinstall 206577664 Sep 27 17:44 o1_mf_temp_9x4r5tvk_.tmp
-rw-r----- 1 oracle oinstall 461381632 Sep 27 17:50 o1_mf_undotbs1_9x4qzn1q_.dbf
-rw-r----- 1 oracle oinstall   5251072 Sep 27 16:57 o1_mf_users_9x4qzly8_.dbf
-rw-r----- 1 oracle oinstall 104865792 Jul 25 15:59 pdbseed_temp012014-07-25_03-52-10-PM.dbf
```

Oracle Managed Files - Seed

- Neu in Oracle 12.1.0.2
- SEED FILE_NAME_CONVERT
  - Kann nur beim Erstellen der CDB angepasst werden
- Beispiel Oracle Dokumentation (Administration Guide Seite 37 – 7):

> *This SEED FILE_NAME_CONVERT clause generates file names for the seed's files in the /oracle/pdbseed directory using file names in the /oracle/dbs directory.*
> *SEED*
> *FILE_NAME_CONVERT = ('/oracle/dbs/', '/oracle/pdbseed/')*

- (ich glaube, das ist kein wirklich gutes Beispiel!)
- Bei Erstellung über dbca nicht möglich

Verzeichnisstruktur für PDBs

- Verzeichnisstruktur vorgeben:
  - PATH_PREFIX → Funktioniert nicht bei OMF
  - FILE_NAME_CONVERT → Ändern des Standardpfades der Dateien
  - DB_CREATE_FILE_DEST → OMF Struktur für ALLE PDBs (spfile Parameter)
  - CREATE_FILE_DEST → OMF Struktur für PDB ändern
  - PDB_CREATE_FILE_DEST → Ändern von bestimmten Teilen der Verzeichnisstruktur

OMF – CREATE_FILE_DEST

- Verzeichnis muss existieren!

```
SQL> CREATE PLUGGABLE DATABASE theo ADMIN USER pdb_admin IDENTIFIED BY manager
     CREATE_FILE_DEST='/u02/oradata/HANNES/THEO';

SQL> SELECT d.file_name
     FROM   cdb_data_files d, v$pdbs p
     WHERE  p.con_id = d.con_id
     AND    p.name = 'THEO';

FILE_NAME
--------------------------------------------------------------------------------
/u02/oradata/HANNES/THEO/HANNES/040F34E2A2CF2025E0530B6E10ACD6DB/datafile/o1_mf_system_b2fsvkqd_.dbf
/u02/oradata/HANNES/THEO/HANNES/040F34E2A2CF2025E0530B6E10ACD6DB/datafile/o1_mf_sysaux_b2fsvkqg_.dbf
```

Pluggable Database mit OMF

- Tipp
  - Symbolischer Link vom PDB-Namen auf die GUID (Shell Script)
- Beim Löschen der PDB (DROP PLUGGABLE DATABASE) wird die Verzeichnisstruktur nicht gelöscht

OMF und Data Guard (1)

- Standby Dateien werden im „falschen“ Verzeichnis angelegt (12.1)

```
ALTER SYSTEM SET db_create_file_dest='/u03/oradata/TC01/pdbshared';
```

- Primäre Datenbank:

```
CREATE PLUGGABLE DATABASE OMFTEST ADMIN USER pdb_admin IDENTIFIED BY manager;
```

- Standby Datenbank:

```
SQL> select name from v$datafile where con_id=7;

NAME
--------------------------------------------------------------------------------
/u03/oradata/TC01/pdbshared/TC01_RZ/4F15FD89C8C141EDE053653C10AC83A4/datafile/o1_mf_system_dk375rxv_.dbf
/u03/oradata/TC01/pdbshared/TC01_RZ/4F15FD89C8C141EDE053653C10AC83A4/datafile/o1_mf_sysaux_dk375s0h_.dbf
/u03/oradata/TC01/pdbshared/TC01_RZ/4F15FD89C8C141EDE053653C10AC83A4/datafile/o1_mf_undotbs1_dk375s0z_.dbf
/u03/oradata/TC01/pdbshared/TC01_RZ/4F15FD89C8C141EDE053653C10AC83A4/datafile/o1_mf_undo_2__00vf50ox_.dbf
```

OMF und Data Guard (2)

- Fehler mit „falschem“ Pfad in 12.2 behoben
- Trotzdem sehr lange Pfadnamen ohne Bezug zur PDB
- Explizite Angabe des Pfades wird nicht erkannt:

```
CREATE TABLESPACE users DATAFILE '/u03/oradata/TC01/pdbshared/users01.dbf' SIZE 100M;
→ Primary: /u03/oradata/TC01/pdbshared/users01.dbf
→ Standby:
  /u03/oradata/TC01/pdbshared/TC01_RZ/4F1638C9B40B4A38E0536F3C10ACE597/datafile/o1_mf_users__020s9frn_.dbf
```

OMF und Data Guard (3)

- Rename Datafile:

```
SQL> ALTER DATABASE MOVE DATAFILE '/u03/oradata/TC01/pdbshared/users01.dbf';

Database altered.

SQL> SELECT name FROM v$datafile WHERE con_id=8;

/u03/oradata/TC01/pdbshared/TC01_RR/datafile/o1_mf_users__0227z7xs_.dbf
```

- Standby: keine Änderung

Schlussfolgerung OMF

- 12.1 → besser nicht verwenden (falls möglich)
- 12.2 → Okay, aber in keinem Fall „vergessen“

Projekt

- Verzicht auf OMF (Direct NFS)
- Skripte für das Erstellen bzw. Verwalten von Pluggable Databases
  - Erstellen der Verzeichnisse auf der Standby Seite

SOUG swiss oracle user group

Herausforderung Data Guard

Einschränkungen

- Temp-Datafiles werden auf der Standby Datenbank nicht automatisch angelegt (Ausnahme Active Data Guard)!

MOS Doc ID 1514588.1
Data Guard Physical Standby - Managing temporary tablespace tempfiles
Oracle Database - Enterprise Edition - Version 10.2.0.1 to 12.1.0.2 [Release 10.2 to 12.1]

The addition of temporary files to TEMP tablespaces in the primary site is not handled automatically through the normal redo apply mechanisms in the same way as regular datafiles if the parameter standby_file_management is set to AUTO. The DBA must manually synchronised the primary and standby tempfile configuration if they require both the sites to be the same.

Auswirkung auf die PDB

- Nach dem Anlegen einer PDB muss die Standby Datenbank geöffnet werden, um dann die Temp-Dateien anzulegen bzw. zu kopieren

CREATE ... FROM SEED

- PDB$SEED existiert auf beiden Seiten
  - Pluggable Database wird auf der Standby Seite automatisch angelegt
  - Keine weiteren Aktionen (bis auf TEMP-TS) notwendig

CREATE ... FROM CLONE

CARAJANDB

SOUG swiss oracle user group

- Obwohl die Quelle auf beiden Seiten existiert wird nicht einmal das Verzeichnis angelegt
- Beispiel:

```
CREATE PLUGGABLE DATABASE paul FROM johannes FILE_NAME_CONVERT=('JOHANNES','PAUL');
ALTER PLUGGABLE DATABASE paul OPEN INSTANCES=ALL;
```

Lösung: CREATE … FROM CLONE

- Primary:

```
ALTER PLUGGABLE DATABASE paul CLOSE INSTANCES=ALL;
```

- Standby:

```
mkdir /u03/oradata/TC01/pdbshared/PAUL
scp trrlora101:/u03/oradata/TC01/pdbshared/PAUL/* /u03/oradata/TC01/pdbshared/PAUL
```

- Data Guard:

```
DGMGRL> edit database "TC01_RZ" set state=APPLY-ON;
DGMGRL> edit database "TC01_RZ" set state=APPLY-ON;
DGMGRL> edit database "TC01_RZ" set state=APPLY-ON;
DGMGRL> edit database "TC01_RZ" set state=APPLY-ON;
DGMGRL> edit database "TC01_RZ" set state=APPLY-ON;
DGMGRL> edit database "TC01_RZ" set state=APPLY-ON;*)
```

*) Für jedes Datafile ein Mal

Create … FROM Manifest

- Es existiert keine „Quelle“, d.h. die Standby PDB wird nicht angelegt.
- Vorgehen wie bei „Create … FROM Clone“

SOUG swiss oracle user group

Resource Management

Ressentiments

- Einzelne PDB kann kompletten Cache verwenden
  - Bekanntes Problem → gilt genau so für Schemakonsolidierung
  - Oracle 12.2 → Neue SGA Parameter

Ressourcenbegrenzung 12.1

CARAJANDB

SOUG swiss oracle user group

- Nur eine SGA → Keine Einschränkung hinsichtlich Buffer-Cache, Shared-Pool, etc.
- Nur ein Set an Redologs und Undo-Tablespace
- Keine Einschränkung bezüglich I/O
- Ressourceneinschränkungen nur über den Resource Manager (z.B: CPU, Sessions, execution Time)
- Einschränkung der Gesamtgröße einer PDB, z.B:
  - `ALTER PLUGGABLE DATABASE STORAGE(MAXSIZE xG);`
- Eigener TEMP-TABLESPACE

Ressourcenbegrenzung 12.2

- Jede PDB eigene SGA
  - SGA_TARGET
  - SGA_MAX_SIZE
  - SGA_MIN_SIZE
  - DB_CACHE_SIZE
  - SHARED_POOL_SIZE
  - PGA_AGGREGATE_TARGET
  - PGA_AGGREGATE_LIMIT

```
SQL> ALTER SESSION SET CONTAINER=suzanne;

SQL> show parameter sga

NAME                                 TYPE        VALUE
------------------------------------ ----------- --------
sga_max_size                         big integer 2512M
sga_min_size                         big integer 0
sga_target                           big integer 0
SQL> ALTER SYSTEM SET sga_target=500M;
SQL> show parameter sga

NAME                                 TYPE        VALUE
------------------------------------ ----------- --------
sga_max_size                         big integer 2512M
sga_min_size                         big integer 0
sga_target                           big integer 500M
SQL> ALTER SESSION SET CONTAINER=CDB$ROOT;
SQL> show parameter sga

NAME                                 TYPE        VALUE
------------------------------------ ----------- --------
sga_max_size                         big integer 2512M
sga_min_size                         big integer 0
sga_target                           big integer 2512M
```

swiss oracle user group

Instance Caging – Nur CDB

- Instance Caging
  - CPU Zusweisung über cpu_count
- Voraussetzung
  - Resource Manager eingeschaltet
- cpu_count ist Instanz-spezifisch, d.h. setzen pro PDB ist nicht möglich

Resource Manager

- CDB Resource Plans
  - Aufteilung von Ressourcen auf einzelne PDBs
  - Verteilung der gesamten Ressourcen über Shares
- PDB Resource Plans
  - Aufteilung von Ressourcen innerhalb einer PDB

CDB Resource Plan

- Beschränkung über Shares
  - Verhältnis der PDBs zueinander
- Derzeit nur
  - CPU
  - Parallelisierungsgrad
- Kontrolliert über „Directives“

Erstellung von Directives

- DBMS_RESOURCE_MANAGER.CREATE_CDB_PLAN_DIRECTIVE
- Vorgehensweise wie bei „normalen“ Resourceplänen
  1. Erstellung einer „Pending“ Area
  2. Erstellung des Resource Plans
  3. Erstellung von Directives
  4. Validierung der Pending Area
  5. Ausführen (Submit) der Pending Area

Vorarbeiten

- Erstellung einer Pending Area

```
BEGIN
    dbms_resource_manager.create_pending_area();
END;
```

- Erstellen eines Resource Plans

```
BEGIN
    dbms_resource_manager.create_cdb_plan(
        PLAN                    => 'johanns_plan',
        COMMENT                 => 'CDB Resource Plan fuer Datenbank CJOHANN');
END;
```

Erstellung von Plan Directives

- create_cdb_plan_directive
  - Plan für Pluggable Database
  - Drei Parameter:
    - shares → Verhältnis der PDBs zueinander (Gesamtheit der Shares = 100%)
    - utilization_limit → max. prozentualer Anteil der CPU
    - parallel_server_limit → maximal prozentualer Anteil der Parallel Server Prozesse

```
dbms_resource_manager.create_cdb_plan_directive(
        PLAN                        => <planname>,
        pluggable_database          => <PDB>,
        shares                      => <number>,
        utilization_limit           => <Number>,
        parallel_server_limit       => <Number>);
```

Erstellung von Plan Directives

```
BEGIN
    dbms_resource_manager.create_cdb_plan_directive(
        PLAN                    => 'johanns_plan',
        pluggable_database      => 'JOHANN1',
        shares                  => 100,
        utilization_limit       => 100,
        parallel_server_limit   => 100);
    dbms_resource_manager.create_cdb_plan_directive(
        PLAN                    => 'johanns_plan',
        pluggable_database      => 'JOHANN2',
        shares                  => 100,
        utilization_limit       => 100,
        parallel_server_limit   => 100);
    dbms_resource_manager.create_cdb_plan_directive(
        PLAN                    => 'johanns_plan',
        pluggable_database      => 'JOHANN1CLONE',
        shares                  => 10,
        utilization_limit       => 50,
        parallel_server_limit   => 0);
END;
```

Ausführen

- Validierung der Pending Area

```
BEGIN
    dbms_resource_manager.validate_pending_area();
END;
```

- Submit

```
BEGIN
    dbms_resource_manager.submit_pending_area();
END;
```

Nutzen des Resource Plans

- Enable Resource Plan
- Nur in der CDB möglich

```
ALTER SYSTEM SET RESOURCE_MANAGER_PLAN = 'johanns_plan';
```

Deaktivieren und Löschen des Plans

- Deaktivieren des Plans

```
ALTER SYSTEM SET RESOURCE_MANAGER_PLAN = '';
```

- Löschen des Plans

```
BEGIN
    dbms_resource_manager.create_pending_area();

    dbms_resource_manager.delete_cdb_plan(
        PLAN => 'johanns_plan');

    dbms_resource_manager.validate_pending_area();

    dbms_resource_manager.submit_pending_area();
END;
```

Instance Resource Management

- Zuteilung von maximalen CPU-Ressourcen
- Begrenzung des Parallelisierungsgrades für Datenbankoperationen
- Begrenzung der Anzahl gleichzeitig aktiver Datenbanksitzungen
- Vorgaben für die Nutzung von Speicherplatz in Rollback-Segmenten (undo pool)

Projekt

- Derzeit kein Ressource Management aktiv
- Jede Datenbank eine preferred und eine available Instanz
- Stand April 2017
  - Ca. 15 PDBs in Test (3 CDBs)
  - 5 PDBs in Vorproduktion (2 CDBs)
  - 2 PDBs Produktion (1 CDB)
- NetApp Snapshot Cloning statt Flashback Database

Vorteile Multitenant 12.2

- Zurücksetzen einer einzelnen PDB (Flashback PDB)
  - Voraussetzung: lokales UNDO-Management
- Cloning im Read-Write Modus
  - Aufbau von Vorproduktion, Test schneller und einfacher
- Snapshot Clones
  - Schnelle und effektive Erstellung von Clones für Testsysteme
- Multiple Character Sets in einer CDB
  - Voraussetzung: CDB AL32UTF8

Vorteile Multitenant Database

- Besser als Schema- / Anwendungskonsolidierung
- Flashback Pluggable Database (ab 12.2 bei lokalem Undo-Management)
- Clone Pluggable Database
  - 12.1 nur bei Read-Only PDB
  - 12.2 bei geöffneter (Read-Write) PDB

Vorteile Multitenant mit Data Guard

CARAJANDB

SOUG swiss oracle user group

- Online Cloning einer PDB auch bei 12.1
  - Snapshot Standby
  - Erstellen eines Database Links in die Ziel-Datenbank
  - Cloning der PDB
- Flashback einer Pluggable Database über die Standby (12.1)
  - Set Guaranteed Restore Point
  - Flashback Standby Database
  - Unplug Pluggable Database
  - Kopieren der PDB in die Source
  - Kann auch für eine einzelne Tabelle benutzt werden

Fazit Projekt (1)

CARAJANDB

SOUG swiss oracle user group

- RAC funktioniert hervorragend
  - Diverse Abnahmetests auf allen Stages
  - Rolling Upgrade von Patches ohne Probleme (derzeit kein OJVM)
- Data Guard funktioniert hervorragend
  - Switchover und Failover mehrfach getestet
  - Snapshot Standby für Flashback Database genutzt
  - Große Akzeptanz, da Standby Datenbank jederzeit geprüft werden kann

Fazit Projekt (2)

- … und Multitenant
  - Derzeit noch Verständnisprobleme
    - Stages werden über Export / Import statt Clone erstellt
    - Connect an die PDB über Services und nicht (/ as sysdba)
  - Skripting vereinfacht Erstellung einer PDB
    - Dauer ca. 5 Minuten (inkl. Standby und Services)
  - Probleme
    - In Produktion OWM (Oracle Workspace Manager) nicht installiert
      → Plug-In einer PDB funktioniert nicht
    - Unterschiede zwischen Common User Definition Test / Vorproduktion
      → Nach Plug-In haben die Common User in der PDB keine Rechte
- Keine „Klagen“ der Produktmanager / Anwendungsverantwortlichen!!!

Ausblick

- DOAG 2017 Datenbank — 30. bis 31. Mai 2017
  - „Data Guard – Ist das Einfach“ — 31 Mai
- DOAG Expertenseminar — 12. bis 13. September 2017
  - Hochverfügbarkeit mit Multitenant Database

Kontakt

- 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
- Blog: blog.carajandb.com

SOUG swiss oracle user group

Fragen?