



Für wen lohnt sich schon der Umstieg auf Oracle 12c?

Johannes Ahrends
CarajanDB GmbH



- CarajanDB
- Derzeitige Herausforderungen
- Lösung mit Oracle 12c
- ... und was macht die Standard Edition?

- Experten mit über 20 Jahren Oracle Erfahrung
- Sitz in Erftstadt nahe Köln
- Spezialisten für
 - Backup & Recovery
 - Hochverfügbarkeit
 - Healthchecks
 - Performance Optimierung
 - Einsatz von Oracle Standard Edition
 - Oracle in virtuellen Umgebungen und in der Cloud
 - Monitoring (Grid / Cloud Control, HLMM, Foglight, Spotlight)
- Schulung und Workshops (Oracle, Toad)



- **Eintägiger Workshop zu Oracle 12c**
 - Überblick über viele neue Funktionen
 - Multitenant Database
 - Information Lifecycle Management
 - High Availability
 - Datenbank Administration
 - Application Development
 - Database Express

- **Weitere Informationen: info@carajandb.com**



Derzeitige Herausforderungen

- **Konsolidierung von Datenbanken**
 - > 100 Datenbanken auf einem Rechner
- **Anzahl von Benutzern / Prozessen wächst**
 - > 2000 Prozesse in einer Standard Edition Datenbank
- **Unicode Migration**
 - VARCHAR2(4000) reicht oft nicht
- **Daten archivieren**
 - Markieren oder Löschen von Datensätzen
- **Datenbank Cloning**
 - Schnelles und Ressourcenschonendes Kopieren von Datenbanken



... und wie hilft Oracle 12c?

- **Multitenant Database**
- **Mehrere „Pluggable“ Datenbanken in einem Container**
- **Vorteile:**
 - Weniger Ressourcenverbrauch (nur eine SGA, Instanz)
 - Schnelleres Patchen
- **Nachteil:**
 - Kostenpflichtig
 - „gewöhnungsbedürftig“
- **Derzeit nur für wenige Kunden wichtig**

➤ **Windows:**

- Oracle Prozesse laufen als Threads in einem gemeinsamen Prozess

➤ **Unix / Linux:**

- Jeder Server und Background Task läuft als eigener Prozess
- Ohne Anwendung ca. 40 – 50 Prozesse
- Bei Datenbank-Konsolidierung mehrere 1000 Prozesse möglich

- **Tasks laufen als Threads innerhalb eines Prozesses**
- **Background Prozesse reduzieren:**
 - Parameter `threaded_execution=TRUE`
 - Nur noch 7 Prozesse (statt 33 bei einer SE)
- **Zunächst immer noch dedizierte Serverprozesse**
 - Listener Parameter `DEDICATED_THROUGH_BROKER_LISTENER=ON`
 - Danach keine Serverprozesse mehr

➤ Überprüfung mit v\$processes:

```
SQL> SELECT spid, stid, pname, execution_type, program
        FROM v$process
```

```
        ORDER BY execution_type, stid;
```

SPID	STID	PNAME	EXECUTION_	PROGRAM
-----	-----	-----	-----	-----
			NONE	PSEUDO
2111	2111	PMON	PROCESS	oracle@albinoni.carajandb.intra (PMON)
2113	2113	PSP0	PROCESS	oracle@albinoni.carajandb.intra (PSP0)
2115	2115	VKTM	PROCESS	oracle@albinoni.carajandb.intra (VKTM)
2131	2131	DBW0	PROCESS	oracle@albinoni.carajandb.intra (DBW0)
2119	2119	SCMN	THREAD	oracle@albinoni.carajandb.intra (SCMN)
2125	2125	SCMN	THREAD	oracle@albinoni.carajandb.intra (SCMN)
2119	2132	LGWR	THREAD	oracle@albinoni.carajandb.intra (LGWR)
2119	2133	CKPT	THREAD	oracle@albinoni.carajandb.intra (CKPT)
2119	2135	SMON	THREAD	oracle@albinoni.carajandb.intra (SMON)
2125	2137	RECO	THREAD	oracle@albinoni.carajandb.intra (RECO)
2125	2363		THREAD	oracle@albinoni.carajandb.intra

...

➤ Vorteil

- Wesentlich weniger Prozesse
- ev. Alternative zu Multitenant Database

➤ Nachteil

- Kein '*KILL PROCESS*' mehr möglich
- Kein Login ohne Password (~~CONNECT / AS SYSDBA~~)

➤ **PROCESSOR_GROUP_NAME**

- Binden einer Instanz an bestimmte Prozessoren
- Derzeit nur für Linux x86-64 ab 2.6.32 (cgroups) und Solaris 11 SRU 4 (resource pools)
- Recommended „USE_DEDICATED_BROKER“ (nicht in 11.2.0.4)
- Backport auf 11.2.0.4 (cpu_count ist falsch)
- Bei NUMA Architektur auch Zuweisung des Memories



Unicode Migration

The maximum size of the VARCHAR2, NVARCHAR2, and RAW data types has been increased from 4,000 to 32,767 bytes.”

Quelle: Oracle Database 12c Release 1 (12.1) New Features

- **Bisher maximal 4000 BYTE für VARCHAR2**
- **Ansonsten Umstellung auf CLOB**
- **Nachteil:**
 - CLOB Standardmäßig UCS-2 Zeichensatz bei AL32UTF8
 - Jedes Zeichen belegt zwei mindestens 2 Byte (auch ASCII!)

```
SQL> CREATE TABLE beipackzettel(  
2      zettelid      NUMBER GENERATED AS IDENTITY PRIMARY KEY,  
3      medikament     VARCHAR2(50 CHAR) ,  
4      gegenanzeigen VARCHAR2(5000 CHAR)  
5  );  
      gegenanzeigen  VARCHAR2(5000 CHAR)  
                        *
```

ERROR at line 4:

ORA-00910: specified length too long for its datatype

➤ **Oracle Parameter: MAX_STRING_SIZE=STANDARD (Default)**

- **Oracle 12c: VARCHAR2(32768 BYTE) Aber nicht automatisch!**
- **Einstellung über Oracle Parameter MAX_STRING_SIZE**
 - MAX_STRING_SIZE=STANDARD
 - VARCHAR2(4000 BYTE)
 - NVARCHAR2(4000 BYTE)
 - RAW (2000 BYTE)
 - MAX_STRING_SIZE=EXTENDED
 - VARCHAR2(32768 BYTE)
 - NVARCHAR2(32768 BYTE)
 - RAW (32768 BYTE)

➤ Upgrade der Datenbank für VARCHAR2 > 4000

```
SQL> SHUTDOWN IMMEDIATE  
SQL> STARTUP UPGRADE  
SQL> ALTER SYSTEM SET MAX_STRING_SIZE=EXTENDED;  
SQL> @?/rdbms/admin/utl32k.sql  
SQL> SHUTDOWN IMMEDIATE  
SQL> STARTUP
```

➤ **STANDARD → EXTENDED**

okay

➤ **EXTENDED → STANDARD**

No Way Back!

```
SQL> CREATE TABLE beipackzettel(  
2      zettelid  NUMBER GENERATED AS IDENTITY PRIMARY KEY,  
3      medikament  VARCHAR2(50 CHAR),  
4      gegenanzeigen  VARCHAR2(5000 CHAR)  
5  );
```

Table created.

- **Erfolgreich!**
- **Warum ist das nicht standardmäßig so?**

```
SQL> CREATE INDEX idx_gegenanzeigen  
      ON beipackzettel (gegenanzeigen);  
  
ERROR at line 1:  
ORA-01450: maximum key length (6398) exceeded
```

➤ Wieso dass denn nun wieder?

Logical Database Limits:

Total size of indexed columns →

75% of the database block size minus some overhead

Quelle: Oracle Database Reference 12c

➤ MAX_STRING_SIZE = STANDARD

TABLE_NAME	COLUMN_NAME	DATA_LENGTH
-----	-----	-----
BEIPACKZETTEL	ZETTELID	22
BEIPACKZETTEL	MEDIKAMENT	200
BEIPACKZETTEL	GEGENANZEIGEN	4000

➤ MAX_STRING_SIZE = EXTENDED

TABLE_NAME	COLUMN_NAME	DATA_LENGTH
-----	-----	-----
BEIPACKZETTEL	ZETTELID	22
BEIPACKZETTEL	MEDIKAMENT	200
BEIPACKZETTEL	GEGENANZEIGEN	16000

4. Versuch – Unicode Migration

➤ Alte Datenbank (WE8MSWIN1252):

TABLE_NAME	NUM_ROWS	AVG_ROW_LEN	BLOCKS
-----	-----	-----	-----
BEIPACKZETTEL	5000	3991	5032

➤ Neue Datenbank (AL32UTF8):

TABLE_NAME	NUM_ROWS	AVG_ROW_LEN	BLOCKS
-----	-----	-----	-----
BEIPACKZETTEL	5000	4079	50

Woher kommt der Unterschied?

```
SELECT segment_name, BLOCKS FROM dba_segments
WHERE owner = 'DEMO'
ORDER BY BLOCKS desc
FETCH FIRST 10 PERCENT ROWS WITH TIES;
```

➤ MAX_STRING_SIZE=STANDARD

SEGMENT_NAME	BLOCKS
-----	-----
BEIPACKZETTEL	5120
SYS_LOB0000092011C00006\$\$	16
PK_POSITIONEN	16

➤ MAX_STRING_SIZE=EXTENDED

SEGMENT_NAME	BLOCKS
-----	-----
SYS_LOB0000092056C00003\$\$	5784
BEIPACKZETTEL	56
PK_POSITIONEN	24

- **VARCHAR2 > 4000 → CLOB**
- **CLOB aus VARCHAR2 → „normaler“ Datenbankzeichensatz (z.B. AL32UTF8)**
- **Indizes ev. in separatem Tablespace mit größerer Blocksize anlegen**
- **Beachten:**
 - CLOB wird bei der Tabellengröße nicht mit einkalkuliert!



Daten archivieren

➤ Unsichtbare Spalte für „normale“ SELECT-Befehle

```
SQL> CREATE TABLE mitarbeiter(  
    mitid      NUMBER(5) GENERATED AS IDENTITY,  
    vorname    VARCHAR2(50 CHAR),  
    nachname   VARCHAR2(50 CHAR),  
    gehalt     NUMBER(10) INVISIBLE);
```

```
SQL> SELECT * FROM mitarbeiter;
```

MITID	VORNAME	NACHNAME
1	Peter	Meier
2	Paul	Meuller
3	Karl	May
4	Franz-Peter	Tebartz van Elst

➤ Keine Darstellung bei DESCRIBE

```
SQL> DESC mitarbeiter
Name          Null?    Type
-----
MITID         NOT NULL NUMBER(5)
VORNAME              VARCHAR2(50 CHAR)
NACHNAME              VARCHAR2(50 CHAR)
```

➤ Spalte kann explizit ausgewählt werden

```
SQL> SELECT vorname, nachname, gehalt
       2 FROM   mitarbeiter;
```

VORNAME	NACHNAME	GEHALT
Peter	Meier	5000
Paul	Meuller	3000
Karl	May	2000
Franz-Peter	Tebartz van Elst	100000

Was soll das dann?

➤ **Invisible Column wird verwendet für:**

- In Database Archiving
- Temporal Validity

- „Inaktive“ Zeilen in einer Tabelle

- Nicht sichtbar für Anwendungen

- **ROW ARCHIVAL** einschalten

```
ALTER TABLE auftraege ROW ARCHIVAL;
```

- **Zusätzliche unsichtbare Spalte ORA_ARCHIVE_STATE**

➤ Anzeige über `user_tab_columns` zeigt keinen Unterschied

COLUMN_NAME	DATA_TYPE	DATA_LENGTH
-----	-----	-----
AUFDATUM	DATE	7
AUFID	NUMBER	22
AUFSTATUS	CHAR	4
LIEFERDATUM	DATE	7
PERSID	NUMBER	22

➤ Anzeige aller Columns über `user_tab_cols`

COLUMN_NAME	DATA_TYPE	DATA_LENGTH	HIDDEN_COL
-----	-----	-----	-----
AUFDATUM	DATE	7	NO
AUFID	NUMBER	22	NO
AUFSTATUS	CHAR	4	NO
LIEFERDATUM	DATE	7	NO
ORA_ARCHIVE_STATE	VARCHAR2	4000	YES
PERSID	NUMBER	22	NO
SYS_NC00006\$	RAW	126	YES

➤ Datensätze archivieren

```
SQL> SELECT min(aufdatum), max(aufdatum), count(*)
        FROM auftraege
        WHERE aufstatus='G'
        AND aufdatum < to_date('01.10.2013','DD.MM.YYYY');
```

MIN(AUFDATUM)	MAX(AUFDATUM)	COUNT(*)
2009-11-13	2013-09-28	571

```
SQL> UPDATE auftraege
        SET ora_archive_state = 'Am 2013-11-20 archiviert'
        WHERE aufstatus='G'
        AND aufdatum < to_date('01.01.2013','DD.MM.YYYY');
```

```
SQL> COMMIT;
```

➤ Gültige Datensätze anzeigen (ohne Änderung der Abfrage!)

```
SQL> SELECT min(aufdatum), max(aufdatum), count(*)  
        FROM auftraege  
        WHERE aufstatus='G'  
        AND aufdatum < to_date('01.10.2013','DD.MM.YYYY');
```

MIN(AUFDATUM)	MAX(AUFDATUM)	COUNT(*)
2013-01-02	2013-09-28	103

➤ **Vorteil:**

- Keine kostspielige DELETE Operation
- Keine Änderung der Anwendungslogik
- Datensätze stehen bei Bedarf immer noch zur Verfügung

➤ **Nachteil:**

- Datenbank wächst
- Gefahr von Row Chaining (Migrated Rows) durch zusätzliche Spalte

➤ Sichtbare Timestamp Spalten

```
SQL> ALTER TABLE adressen  
      ADD gueltig_von TIMESTAMP;  
SQL> ALTER TABLE adressen  
      ADD gueltig_bis TIMESTAMP;  
SQL> ALTER TABLE adressen  
      ADD (PERIOD FOR gueltige_adresse (gueltig_von, gueltig_bis));
```

➤ Sicht auf einen bestimmten Zeitpunkt in der Abfrage

```
SELECT adrid, persid, strasse, plz, ort  
FROM adressen  
AS OF period FOR  
   gueltige_adresse to_timestamp('2014-01-01', 'YYYY-MM-DD');
```

➤ oder mit Flashback Query:

```
execute dbms_flashback_archive.enable_at_valid_time('CURRENT');
```

➤ Temporal Validity mit unsichtbaren Spalten

```
SQL> ALTER TABLE adressen  
      ADD (PERIOD FOR gueltige_adresse);
```

➤ Drei unsichtbare Spalten

COLUMN_NAME	DATA_TYPE	LENGTH	HIDDEN_C
-----	-----	-----	-----
ADRID	NUMBER	22	NO
ADRTYP	NUMBER	22	NO
BUNDESLAND	NUMBER	22	NO
GUELTIGE_ADRESSE	NUMBER	22	YES
GUELTIGE_ADRESSE_END	TIMESTAMP (6) WITH TIME ZONE	13	YES
GUELTIGE_ADRESSE_START	TIMESTAMP (6) WITH TIME ZONE	13	YES
ORT	VARCHAR2	200	NO
PERSID	NUMBER	22	NO
PLZ	VARCHAR2	32	NO
STRASSE	VARCHAR2	200	NO

Temporal Validity – Welche Version?

➤ **Version 1:**

- Falls schon Gültigkeitszeiträume existieren (als Timestamp!)
- Funktioniert mit der Standard Edition

➤ **Version 2:**

- Keine Änderung der Anwendung (Falls mit SELECT * gearbeitet)

- **Geht schon mit Version 11g**
 - CloneDB basiert auf einer RMAN Kopie
- **Pluggable Database Snapshot Clone**
 - Voraussetzung
 - Pluggable Database im Read-Only Modus als Quelle
 - Snapshotfähiges Storage (NFS, ZFS) oder ACFS
 - ACFS für Datafiles erlaubt
 - Direkter Verweis der Blöcke auf die Pluggable Database
 - Kein Backup mehr erforderlich
 - Source PDB kann anschließend wieder RW geöffnet werden
- **Gute Methode um mehrere Test oder Entwicklungsdatenbanken zu erstellen**

... und was macht die Standard Edition?

- **Flashback Data Archive kostenlos auch in der Standard Edition**
 - Backport auch für Oracle 11.2.0.4
- **Multitenant Database mit genau einer Pluggable Database**
 - Schnelles Patching
- **Extended VARCHAR2**
- **Invisible Columns mit In Database Archiving und Temporal Validity**

... weitere Interessante Features

➤ **Data Redaction**

- Automatisches „verstecken“ sensibler Daten (z.B. Kreditkarte)
- Teil der Advanced Security Option
- Verfügbar schon mit 11.2.0.4

➤ **Application Continuity**

- RAC und Data Guard Failover auch von offenen Transaktionen

➤ **Data Guard Erweiterungen**

- Resumable Switchover

➤ **und und und ...**

- **19.02.2014 *Real World Performance Tour* München**
 - Tom Kyte, Senior Technical Architect ORACLE Server Technology Division, ORACLE Vice President
 - Graham Wood, Architect Server Technologies
 - Andrew Holdsworth, Senior Director Real World Performance Server Technologies
- **04.06.2014 *DOAG Datenbank Konferenz* Düsseldorf**
 - Call for Paper 02.12.2013 – 31.01.2014
- **11. - 12.02.2014 *Expertenseminar Pluggable Database* Berlin**
 - Referent: Johannes Ahrends, CarajanDB GmbH

➤ Verschieben eines Datafiles Online möglich

```
ALTER DATABASE MOVE DATAFILE  
'<alte Datei>' TO '<neue Datei>' [KEEP | RESIZE];
```

- **Eintägiger Workshop zu Oracle 12c**
 - Überblick über viele neue Funktionen
 - Multitenant Database
 - Information Lifecycle Management
 - High Availability
 - Datenbank Administration
 - Application Development
 - Database Express

- **Weitere Informationen: info@carajandb.com**



Fragen?

www.carajandb.com

johannes.ahrends@carajandb.com