Migration einer terrabytegroßen Datenbank bei minimaler Downtime

DOAG-Konferenz 2011 17. November in Nürnberg

Warum Migrationen?

- Wachstum von Daten- und Nutzerlast
- Support Lifecycle bei Hard- und Software
- Konsolidierung
- Rechenzentrum- oder Betreiberwechsel
- Strategiewechsel bei der Plattform

Arten von Migrationen

- Wechsel der Datenbank-Maschine
- Austausch des Speicher-Subsystems
- Paritionierung von Daten auf verschiedene Datenbanken
- Wechsel der Betriebsystemplattform – Cross Plattform Migration oder auch 32/64 Bit Migration
- Wechsel der Datenbankedition
- Wechsel der Datenbanksoftware
- Migration von Zeichensätzen

Datenwachstum

- Begünstigt durch die Entwicklung der Hardware-Technologie und ein stark ansteigendes Datenwachstum.
- Der Speicherspezialist EMC Corporation prognostiziert eine Steigerung der jährlich entstehenden Informationen bis 2020 um den Faktor 44.
- Sichwort Web 2.0 und NoSQL-Datenbanken.

Was ist Skalierung

- Skalierung als eine Eigenschaft eines Systems hat keine allgemeingültige Definition
- Im Folgenden ist ein System skalierbar, wenn der Durchsatz der zu verarbeitenden Anfragen steigt, wenn Hardware-Ressourcen hinzugefügt werden.
- Mehrprozessorsysteme werden in zwei Skalierungsgruppen unterteilt.

Horizontale Skalierung

- Die horizontale Skalierung fügt einer Gruppe von kleineren über einen Interconnect verbundenen Maschinen weitere hinzu.
- Als Interconnect dient ein schnelles und dediziertes Netzwerk. Jede Maschine hat ein eigenes Betriebssystem und eine eigene Hauptspeicherverwaltung.
- Auch als Scale-out oder *Sharding* bezeichnet.
- Meißt für DDBMS relevant.

Vertikale Skalierung

- Die vertikale Skalierung fügt einem symmetrischen Multiprozessorsystem (SMP) weitere Hardware hinzu.
- Bei dieser Art der Hardwarearchitektur sind zwei oder mehr identische Hauptprozessoren mit einem gemeinsamen Hauptspeicher verbunden und werden von einer Betriebssystem-Instanz gesteuert.
- Auch als Scale-up bezeichnet.
- Kennt Grenzen, die eine Migration notwendig machen können.

Cross-Plattform-Migrationen

- Entscheident ist die Betriebsystem- und Oracle-Version auf dem Start- und Zielsystem.
- Export/Import wahlweise mit Datapump ab Version 10.1.0.2
- Transportable Tablespaces
- RMAN Convert Database Funktion ab Oracle-10g
- Oracle-Streams Replikation
- Create table as (CTAS)
- DBMS_Redefinition Package
- Oracle-Golden-Gate
- Ausgangspunkt für Recherchen ist Metalink-Note 733205.1

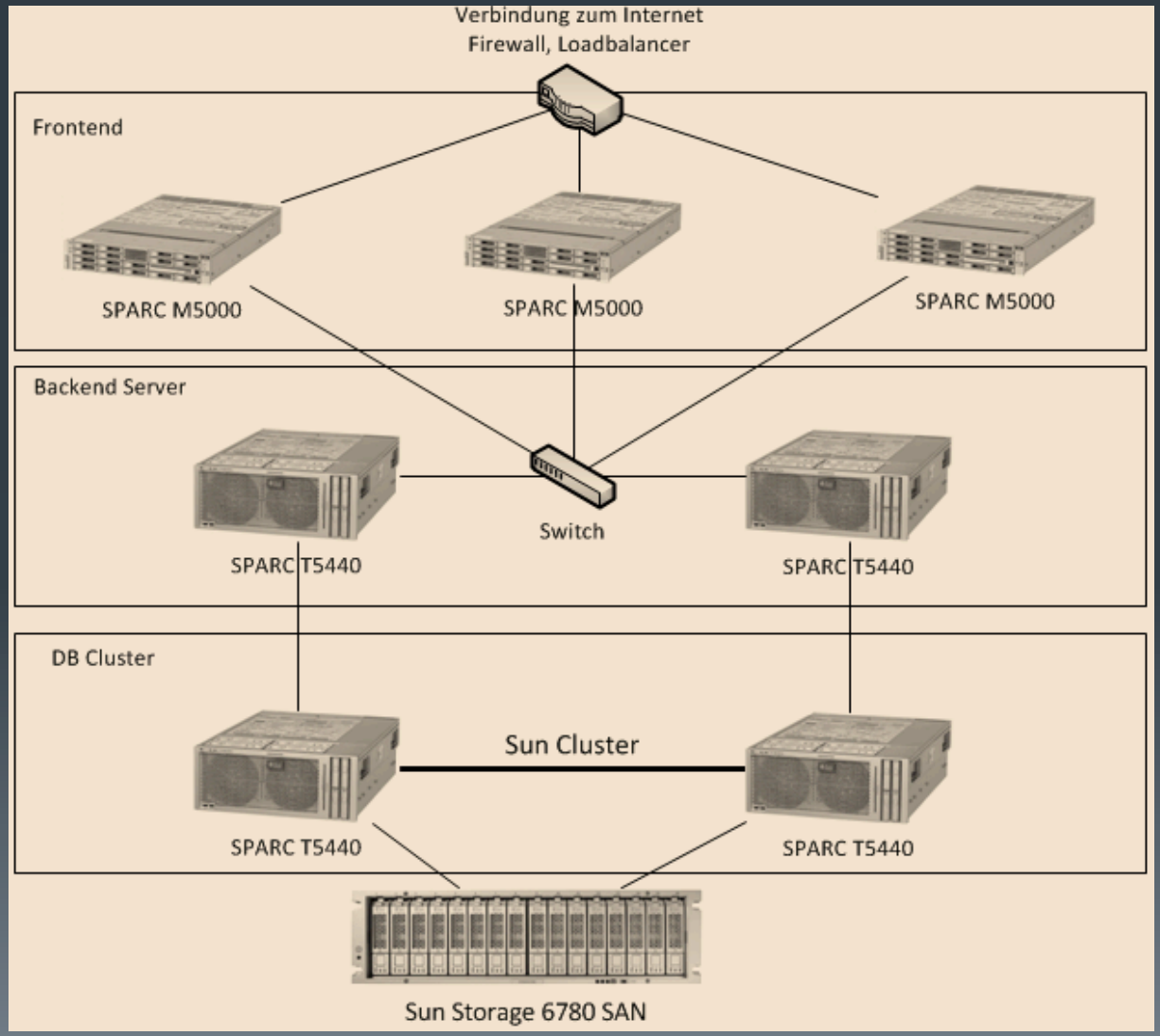
Andere Migrationen

- Gleiche Werkzeige wie Cross-Plattform-Migrationen zusätzlich:
- Für Zeichensatzmigrationen csscan/scalter
- „Database Migration Assistent for Unicode“ (DMU)
- Ordinäre RMAN-Wiederherstellung
- Austausch des Speichersubsystem mit anderen Lösungen (z. B. SANsymphonie, ASM-Rebalance, ZFS)

Das konkrete Projekt

- Oracle 10gR2 Datenbank unter Solaris 10 in einer aktiv/ passiv Cluster-Konfiguration.
- Backup mit RMAN auf ein RAID 5 Backup-Laufwerk.
- 24/7 Dienst
- Hohe Datenmengen aus vergleichbar langsamem Speicher-System => Volle Wiederherstellung ca. 24h Downtime.
- Austausch Datenbank-Maschine und Speicher-Subsystem
- Abwägung nur Migration, Update und Migration oder Upgrade und Migration.

Installationsbeispiel



Grunsätzliche Ansätze

- SAN-System an der neuen Maschine mounten und online migrieren.
  - Systeme sind räumlich gerennt
  - Problem DBMS_REDEFINITION nicht in der SE + sehr große einzelene Segmente wären ein Risiko
- Manuelle und programmatische Lösung. Partitionierung der Kundenbeiche und Kopieren in kleinen, jeweils nur für einen Bereich wirkenden Ausfallzeiten.
  - Testaufwand hoch
  - Gefahr der Inkonsistenz, da zwei Datenbanken Online sind, lange Migrationsdauer und Gefahr eines Systematischen Fehlers, der erst zu fortgeschrittener Zeit bemerkt wird => keine Fallback Möglichkeit

Grunsätzliche Ansätze II

- Ändern der Datenbank zu einer Enterprise-Edition mit Paritionierung nach Kundenbereichen und succesives Übertragen von Partitionen auf das neue System.
  - Lizenzkosten
  - Applikation muss wissen, welcher Kunden in welcher Datenbank zu finden ist.
  - Partionieren der Tabellen mit rdbms_redefinition hat beanspruchrt das E/A System der laufenden Produktion.
- RMAN-Wiederherstellung eines passiven Standby-Systems. Nachfahren von Archivelogs und Wechsel der Systeme zu einem bestimmten Zeitpunkt.
  - Atomare Lösung mit Fallback-Möglichkeit

Desaster recovery in a nutshell

run{

backup as backupset database plus archivelog format '/global/backupdatabase/AL_%d_%t_%s_%p' delete input;

}

RMAN> **connect target**

connected to target database (not started)

RMAN> **startup nomount**

startup failed: ORA-01078: failure in processing system parameters

LRM-00109: could not open parameter file '/opt/oracle/product/10.2/dbs/initMYDB.ora'

trying to start the Oracle instance without parameter files

Oracle instance started

...

Desaster recovery in a nutshell II

RMAN> restore spfile from '/opt/oracle/...‘;
RMAN> restore controlfile from '/opt/oracle...';
RMAN> startup force mount
RMAN> startup mount pfile='/local/oracle/.../init.ora‘
run{
restore database;
recover database;
}

- Neue Archivelogs hinzufügen:
  - catalog backuppiece '/backup/MYSID/01dmsbj4_1_1.bcp';
  - Dann „recover database“ zum Anwenden.

Desaster recovery in a nutshell III

- Datenbank kann mit „alter database open read only“ betrachtet werden.
- „alter database open resetlogs“
  - Neuer Redo-Stream
  - Kein weitere Recovery möglich.
  - Erstellt eine neue Inkarnation der Datenbank.
  - Es muss danach sofort eine neues Backup erstellt werden.

Planung – Input für PMs

- Vorbereitung bis zum Beginn der Downtime
  - Installationen, Aufbau Standby System, Abhängig vom System
  - Ziel: Möglichst viele Tätigkeiten in diese Phase verlegen
- Während der Downtime
  - Tatigkeiten Minutiös planen
  - Nutzung einer Checkliste
  - DBA-Tests, QS, Fallback-Bedingungen u.a.
- Nacharbeiten
  - Monitoring der Systeme
  - Vorhalten des alten Systems für evtl. Recherchen
  - Je länger das neue System Online ist, desto irrelavanter wird das ehemalige System.

Vorbereitung bis zum Beginn der Migration

- Erstellung einer Zeitabschätzung
- Beschaffungen / Installationen
  - Installation Betriebsystem
    - kann höheres Patch-Level haben
    - Zertifizierung beachten
    - Möglichst gleiche Parameter (/etc/system, Profile u. a.)
    - gleiche Laufwerkspfade
  - Installation Datenbank
    - Gleiche Version und Patchlevel wie das Produktionssystem.
    - Gleiche Pfade verwenden (für RMAN-Wiederherstellung oder Skripte)
  - Installation Solaris Clusterware
  - Einrichtung Netzwerk Multipathing u. a.

Vorbereitung bis zum Beginn der Migration II

- Testen des Shared-Memory-Layouts
  - Für alle Instanzen einer Maschine
  - $ORACLE_HOME/dbs/initoratest1.ora->
  - Starten der Instanz im *nomount* Modus

```
DB_NAME=oratest1
sga_max_size=4347483648
sga_target=4347483648
PROCESSES=500
sessions=555
compatible='10.2.0.1.0'
```

```
-bash-3.00$ export ORACLE_SID=oratest1
-bash-3.00$ sqlplus / as sysdba

SQL*Plus: Release 10.2.0.2.0 - Production on Mon Feb 11 21:21:54 2008

Copyright (c) 1982, 2005, Oracle. All Rights Reserved.

Connected to an idle instance.

SQL> startup nomount
ORA-27102: out of memory
SVR4 Error: 22: Invalid argument
```

„Shared memory and semaphores explained“ -> andschulte.de/shared_memory.html

Vorbereitung bis zum Beginn der Migration III

- Erstellung einer Testdatenbank
  - Testen der Shared-Memory-Konfiguration des Betriebssystems
  - Einrichtung der Cluster-Konfiguration
    - DB hat gleichen Namen wie die Produktions-DB
    - Listener Ressource
    - andere Ressourcen
  - Durchführung von Cluster-Schenktests
  - Einrichtung des Oracle-Networking
  - Einrichtung Backup-Mechanismus
  - Einrichtung sonstiger Cronjobs

Vorbereitung bis zum Beginn der Migration IV

- Konfiguration des Direct-I/O prüfen
  - Tool notwendig: http://www.solarisinternals.com/si/tools/directiostat/index.php
  - DB Parameter filesystemio_options = SETALL in der Testdb setzen
  - Künstliche Schreib-/Leselast erzeugen und wie folgt prüfen:

```
-bash-3.00$ ./directiostat 3
  lreads lwrites  preads pwrites     Krd     Kwr holdrds  nflush
      52      29      28      38       0       0       0       0
       0      18       0      18       6     160       0       0
       0       3       0       3       6      38       0       0
```

Vorbereitung bis zum Beginn der Migration V

- Messen der Leselast auf der aktuellen Produktion für einen Vergleich mit dem neuen System mit **dtrace**
  - Am besten zu einer Tageszeit hoher Last

```
-bash-3.00# dtrace -n 'syscall::read*:return {@a[execname ] = quantize(arg0);}'
dtrace: description 'syscall::read*:return ' matched 3 probes
^C
  oracle
           value  ------------- Distribution ------------- count
              -1 |                                         0
               0 |                                         2
               1 |                                         0
               2 |                                         0
               4 |                                         0
               8 |                                         3
              16 |                                         0
              32 |                                         5
              64 |                                         5
             128 |                                         6
             256 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  483
             512 |@                                        16
            1024 |                                         4
            2048 |                                         1
            4096 |                                         1
            8192 |                                         0
           16384 |                                         2
           32768 |                                         0
```

dtrace Rechte geben –
als root:

```
usermod -K defaultpriv=basic,
dtrace_kernel oracle
```

Vorbereitung bis zum Beginn der Migration VI

- Messen der Schreiblast auf der aktuellen Produktion für einen Vergleich mit dem neuen System mit **dtrace**
  - Beide dtrace-Tests nach der Migration wiederholen.

```
#pragma D option quiet
io:::start
/args[0]->b_flags&B_WRITE/
{
@[execname,args[2]->fi_dirname]=count();
}
END
{
printf("%20s%51s%5s\n","WHO","WHERE","COUNT");
printa("%20s%51s%5@d\n",@);
}

Aufruf/Ausgabe:
-bash-3.00# dtrace -s ./whowrites.d
^C
                 WHO                                                WHERECOUNT
              fsflush                    /global/ora_db/oradata/files     4
               oracle                    /global/ora_db/oradata/files  1143
                  ...
```

Vorbereitung bis zum Beginn der Migration VII

- Ermittlung Zahlen für die Zeitabschätzung
  - Wie lange dauert das Kopieren von Backupsets vom alten zum neuen System?
  - Wie viele Archivelogs fallen pro Tag an?
  - Wie ist die Nutzerlast kurz vor dem Switch?
  - Verbindliche Festlegung eines Zeitplans i. d. R. ASAP -> Sicherheitsaufschlag einplanen.

Aufbau des Standby-Systems

- Kopieren der Backupdateien
  - Ist das erste Backupset angekommen, kann die Wiederherstellung gestartet werden, während weitere Backupsets kopiert werden.
  - Das Produktionssystem darf ab jetzt keine Archivelogs mehr löschen.
  - Neu angelegt Datenfiles werden auch im Standbysystem neu angelegt -> Voraussetzung: Verzeichnis muss vorhanden sein.
- Anfallende Archivelogs
  - Kopieren
  - Katalogisieren
  - Anwenden

Todos während/kurz vor der Downtime

- Kurz vor der Downtime das Intervall für das Übertragen der Archivelogs verkürzen.
- Anzahl der dirty-Blocks im Block-Buffer verkleinern
  - Parameter FAST_START_MTTR_TARGET oder
  - Parameter LOG_CHECKPOINT_INTERVAL oder
  - Manuell „alter system checkpoint
- Herunterfahren des Listeners
- Entweder stoppen der Applikation oder Neustart der Instanz.
- Mit „select * from v$sesison“ prüfen, dass keine Nutzer mit der DB verbunden sind.

Todos während der Downtime II

- Übertragen der letzten Archivelogs zur neuen Produktion.
- Anwenden der ausstehenden Archivelogs.
- Instanz Nur-Lesend öffnen
- Mit einem vorher angefertigtem Skript:
  - Vergleich der Datenbestande alte/neue Produktion
  - Test auf Invalid Objects
  - Prüfen von Sequenz-Nummern
  - ...
- Wenn alles stimmt:
- Im *Mount*-Modus neu starten
- „alter database open resetlogs“
  - Kein weiteres Nachfahren von archivelogs möglich.
  - Neu erstellte Redo/Archivelog nicht mit bestehendem Backup nutzbar.

Todos während der Downtime III

- Optional einspielen von Patchsets, PSU bzw. CPUs
  - Vorteil Fallback System steht bereits, keine weiteres Backup notwendig.
  - Auch für Tests entsteht kein doppelter Auswand.
  - Zeitaufwand ca. bei 20-40 Minuten.
  - Nachteil je mehr Änderungen man hat, desto schwieriger ist die verursachenden Komponente ausfindig zu machen.
    - Neues Betriebsystem Patchlevel
    - Neuer Datenbank Patchlevel
    - Neue Hardware für die DB-Maschine
    - Neues Speicher-Subsystem
    - Evtl. Neue Netzwerk-Komponeten / Applikationsänderungen
- =>JE WENIGER ÄNDERUNGEN, DESTO BESSER.

Todos während der Downtime IV

- Starten des Listeners (als Cluster Ressource)
- Konfiguration der Applikation auf das neue Datenbank-System
- Tests durch die QS-Abteilung
- Finale Entscheidung für das „GO“
- Datenbank in die Cluster Konfiguration aufnehmen (erfordert einen Neustart)
- Vor oder nach dem QS-Test Cluster-Schwenktest mit der Produktionsdatenbank ausführen.
- Altes Datenbank-System nur lesend öffnen, damit nicht durch einen Konfigurationsfehler ein Teil der Anwendung auf das alte System zugreift.

Nach der Migration

- Wichtig: Sofort ein neues Backup starten.
- Cronjobs, sonstige regelmäßige Jobs einrichten.
- In der ersten Phase, prüfen ob der Archivelog-Mechanismus richtig läuft, sonst droht eine Ausfallzeit.
- Überprüfen von Betriebsystem-Ressourchen
  - CPU- / Netzlast, E/A-Last und Load-Average
  - Shared Memory / Semaphorenauslastung
  - Anzahl offener Dateideskriptoren zur Vermeidung: „too many open files error“
- Überprüfen von DB-Ressourchen und Vergleich mit altem System
  - Statspack
  - AWR

Nach der Migration: Erstes Monitoring

- Shared-Memory Segment:

```
-bash-3.00$ ipcs -pma
IPC status from  as of Mon Feb 11 21:23:01 CET 2008
T         ID      K-bash-3.00$ pfiles 20963|wc -l
      14
EY        MODE        OWNER    GROUP  CREATOR   CGROUP NATTCH      SEGSZ
Shared Memory:
m          1   0x3cb26b68 --rw-r-----   oracle oinstall   oracle oinstall     26  817897472
```

- Überwachen aller offenen Datei-Descriptoren:

```
let "TOTAL_OPENED=0"
for A_PID in `ls /proc`; do
FILES_OPENED=`/usr/proc/bin/pfiles $A_PID 2>/dev/null| grep "ino:" | wc -l`
let "TOTAL_OPENED = TOTAL_OPENED + $FILES_OPENED"
done
echo "Opened files = $TOTAL_OPENED"
```

- Wiederholung der dtrace Lese- und Schreibtests.

Nach der Migration: DB-Report

- Vergleich von Reports alter mit neuer Produktion
  - Entweder Statspack oder AWR
  - Automatic Workload Repository (AWR) ist Teil des „Database Diagnostic Pack“ – muss lizenziert werden.
  - Immer gleiche Zeiträume und gleiche Zeitdauer vergleichen i. d. R. 15 Minuten oder 1 Stunde.

Probleme: HW – contention I

- Oracle LOBs + ASSM + viele CPUs = enq: HW – contention
  - Geänderte Hardware hat mehr CPU-Cores und Threads
  - Problem war vorher bekannt und wurde deutlich verschärft

Top 5 Timed Events

Event	Waits	Time (s)	Avg wait (ms)	%Total Call Time	Wait Class
enq: HW - contention	**224,968**	**658,323**	**2926**	**93.1**	**Configurat**
db file sequential read	1,028,687	22,611	22	3.2	User I/O
CPU time		20,024		2.8	
read by other session	191,140	2,677	14	0.4	User I/O
direct path read	379,201	2,261	6	0.3	User I/O

^LWait Events
-> s - second
...

Event	Waits	%Time -outs	Total Wait Time (s)	Avg wait (ms)	Waits /txn
enq: HW - contention	224,968	99.9	658,323	2926	1.9
db file sequential read	1,028,687	.0	22,611	22	8.7

...

Probleme: HW – contention II

- Wichtig: Problem entsteht in diesem Fall beim Verschieben der High-Water-Mark
  - Durch Automatic Segement Space Management (ASSM)
  - Durch die geänderte CPU-Architektur mit mehr Cores und mehr Threads nahmen die Wait-Events zu.

```
^LEnqueue Activity
-> only enqueues with waits are shown
-> Enqueue stats gathered prior to 10g should not be compared with 10g data
-> ordered by Wait Time desc, Waits desc

Enqueue Type (Request Reason)
------------------------------------------------------------------------------
    Requests    Succ Gets Failed Gets       Waits  Wt Time (s) Av Wt Time(ms)
------------ ------------ ----------- ----------- ------------ --------------
HW-Segment High Water Mark
      12,154        8,437       3,398          23       16,624     722,796.87
RO-Multiple Object Reuse (fast object reuse)
         333          333           0          37            8         223.78
TX-Transaction (index contention)
          23           23           0          23            2          70.52
```

Probleme: HW – contention III

- Bug 6376915: ENQ: HW - CONTENTION WITH LOB SEGMENTS
- und weitere
- Je größer das Segment, je mehr konurierende CPU und je mehr gleichzeitige Lese/Schreibvorgäbge, dest schlimmer das Problem.
- Abhilfe:

ALTER TABLE <lob_table> MODIFY LOB (<column_name>) (allocate extent (size <extent size>));

Probleme: HW – contention IV

- Aber Vorsicht:

Bug 9711859 ORA-600 [ktsptrn_fix-extmap] during extent allocation caused by bug 8198906

ORA-600 [ktsptrn_fix-extmap] caused by a corruption in a LOB segment stored in an ASSM tablespace when there is a manual extent allocation through:

ALTER TABLE ALLOCATE EXTENT.

```
ERROR at line 1:
ORA-00607: Internal error occurred while making a change to a data block
ORA-00600: internal error code, arguments: [kddummy_blkchk], [22], [1858570], [18009],
[], [], [], []
```

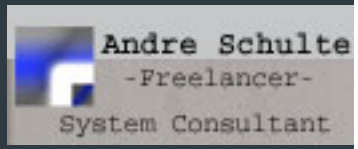
Probleme: Stuck Recovery

- Trat bei mehreren identischen Datenbanken bei nur einer einzigen auf:

```
RMAN-00571:===========================================================
RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============
RMAN-00571:===========================================================
RMAN-03002: failure of recover command at 12/04/2009 10:25:07
ORA-00283: recovery session canceled due to errors
RMAN-11003: failure during parse/execution of SQL statement: alter database recover logfile
'/global
/ora_de/flash_recovery_area/TMDE/archivelog/2009_12_04/o1_mf_1_83309_5kknk61p_.arc'
ORA-00283: recovery session canceled due to errors
ORA-00600: internal error code, arguments: [3020], [160], [932370], [1], [83309], [847366],
[412], []
```

Probleme: Stuck Recovery II

- Reproduzierbar bei jedem Restore/Recovery.
- Bug 5212539 LOB corruption for NOCACHE LOBs (ORA-1555/ORA-22924)
- Nach Anwendung des Patches Restore/Reocvery möglich
- Fehler trat nur auf, wenn ein Rollback für den Lob-Datensatz ausgeführt wurde.



Fragen und Antworten

Vielen Dank für die Aufmerksamkeit