Evaluierung von

Oracle logical Data Guard 10.1

Installation & Administration Guide

Inhaltsverzeichnis

1 Einführung

Die Data Guard Logical Standby Technologie bietet 3 Vorteile:

- **Lastverteilung**: Reporting-Abfragen müssen nicht auf der OLTP Instanz ausgeführt werden, sondern können auf der Logical Standby laufen.

- **Ausfallsicherheit**: Es ist möglich, bei Ausfall der Primary Database die Logical Standby zur Primary zu machen. Das funktioniert allerdings nur, wenn alle benötigten Schemas repliziert werden

- **Rolling Upgrade**s: Mittels Logical Standby ist es möglich, die Downtime bei Oracle Upgrades zu minieren.

Datenbank-Änderungen werden in den Redo-Logs protokolliert. Diese Redo-Logs werden dann zur Standby-Datenbank übermittelt. Dort werden die Redo-Informationen in SQL umgewandelt und auf der Standby Datenbank appliziert.

Die Replizierung bestimmter Tabellen oder Schemas kann bei Bedarf deaktiviert werden. Die Standby-Datenbank ist zu jeder Zeit offen und steht für Anfragen zur Verfügung.

Auf der Standby-Datenbank können zusätzliche Materialized Views oder Indizes auf propagierte Tabellen angelegt werden, ohne dass die Primary Instanz beeinflusst wird.

Ab der Version 10gR1 ist es möglich, die eintreffenden Redo-Informationen in real-time auf die Standby Datenbank anzuwenden. Früher musste bis zum nächsten Redo-Log Switch gewartet werden.

2 Vorbereitungen und Anforderungen

Um Data Guard Logical Standby einsetzen zu können, sind bestimmte Voraussetzungen notwendig:

- Die Betriebssystem-Architektur und Version der beiden Datenbanken muss identisch sein
- Die Oracle-Version der beiden Datenbanken muss identisch sein.
- Die beiden Datenbanken müssen im Archivelog Modus laufen.
- Es muss auf beiden Datenbanken „FORCE LOGGING“ aktiviert sein.
- Es gibt in Version 10gR1 folgende nicht unterstützte Datentypen: BFILE, ROWID, UROWID, user-defined types, IOTs mit Overflow oder LOB columns, object types REFs, varrays, nested tables, XMLType
- Bei Version 10gR2 sind folgende Datentype nicht unterstützt: BFILE, ROWID, UROWID, User-defined types, Collections (varrays und nested tables), XMLType, encrypted columns, Mulitmedia data Types (Spatial, Image and Context)
- Um zu prüfen, welche Objekte nicht für Logical Standby unterstützt werden, kann folgende View abgefragt werden: DBA_LOGSTDBY_UNSUPPORTED
- Oracle empfiehlt, entweder Primary Keys oder Unique indexes auf alle Tabellen zu legen. Dies kann geprüft werden über die View DBA_LOGSTDBY_NOT_UNIQUE
- Supplemental Logging muss aktiviert werden
- Sys-Passwort muss auf allen Systemen identisch sein.

3 Installation

3.1 Erstellen der Physical Standby Database als Grundlage der Logical Standby

3.1.1 Force Logging

```
SQL> alter database force logging;

Database altered.

SQL> select force_logging from v$database;

FORCE_LOGGIN
------------
YES
```

3.1.2 Password-File

```
SQL>  show parameter remote_login_p
NAME                                 TYPE
VALUE
------------------------------------ ---------------------------------------
---- --------------------------------
remote_login_passwordfile            string
EXCLUSIVE
```

3.1.3 Supplemental Logging

```
SQL> alter database add supplemental log data (primary key, unique index)
columns;

Database altered.

SQL> select supplemental_log_data_pk, supplemental_log_data_ui from
v$database;

SUPPLEMENTAL SUPPLEMENTAL
------------ ------------
YES          YES
```

3.1.4 Anlegen der neuen Physical Standby Umgebung

```
mkdir /data/MDDB1LS
ln -s /data/MDDB1LS /oracle/MDDB1LS
ln -s /oracle/MDDB1/10.1.0 /oracle/MDDB1LS/10.1.0
mkdir /oracle/MDDB1LS/oradata
mkdir /oracle/MDDB1LS/oratrace
mkdir  /oracle/MDDB1LS/oraarch
mkdir  /oracle/MDDB1LS/origlogA
mkdir  /oracle/MDDB1LS/origlogB
cd  /oracle/MDDB1LS/oratrace
mkdir adump bdump cdump udump
cd /oracle/MDDB1LS
```

3.1.4.1 listener.ora:

```
LISTENER_MDDB1LS =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = dbhost1.md.intra)(PORT = 1524))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = MDDB1LS))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC2))
      )
    )
  )

SID_LIST_LISTENER_MDDB1LS =
  (SID_LIST =
    (SID_DESC=
      (GLOBAL_DBNAME=MDDB1LS.WORLD)
      (SID_NAME=MDDB1LS)
      (ORACLE_HOME=/oracle/MDDB1LS/10.1.0)
    )
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/MDDB1LS/10.1.0)
      (PROGRAM = extproc)
    )
  )
```

3.1.4.2 tnsnames.ora

```
MDDB1LS.WORLD =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = dbhost1.md.intra)(PORT = 1524))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = MDDB1LS.WORLD)
    )
  )
```

3.1.4.3 Starten des Standby-Listeners

```
lsnrctl start LISTENER_MDDB1LS
```

3.1.4.4 Tests der Verbindung mit tnsping

```
tnsping MDDB1
tnsping MDDB1LS
```

3.1.4.5 Kopieren der Datafiles

```
sqlplus „/as sysdba“
alter database begin backup;
!cp /oracle/MDDB1/oradata/* /oracle/MDDB1LS/oradata/
alter database end backup;
alter database create standby controlfile as ‘/tmp/control01_standby.ctl’
!cp /tmp/control01_standby.ctl /oracle/MDDB1LS/oradata/control_01.ctl
!cp /tmp/control01_standby.ctl /oracle/MDDB1LS/oradata/control_02.ctl
```

3.1.4.6 Änderung der Init-Parameter von Primary

```
alter system set log_archive_config='DG_CONFIG=(MDDB1,MDDB1LS)';
alter system set log_archive_dest_1='LOCATION=/oracle/MDDB1/oraarch/
VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=MDDB1';
alter system set log_archive_dest_2='SERVICE=MDDB1LS LGWR ASYNC=61440
NET_TIMEOUT=30 VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
DB_UNIQUE_NAME=MDDB1LS';
alter system set log_archive_dest_state_1=ENABLE;
alter system set log_archive_dest_state_2=DEFER;
alter system set fal_server=MDDB1LS scope=spfile;
alter system set fal_client=MDDB1 scope=spfile;
alter system set DB_FILE_NAME_CONVERT='/oracle/MDDB1LS','/oracle/MDDB1'
scope=spfile;
alter system set LOG_FILE_NAME_CONVERT='/oracle/MDDB1LS','/oracle/MDDB1'
scope=spfile;
alter system set standby_file_management=auto scope=spfile;
```

3.1.4.7 Erstellen einer init.ora für Standby

Primary:

```
SQL> create pfile='/tmp/initMDDB1.ora' from spfile;
```

Anpassen der init.ora für Standby:

```
cp /tmp/initMDDB1.ora /oracle/MDDB1LS/10.1.0/dbs/initMDDB1LS.ora

*.background_dump_dest='/oracle/MDDB1LS/oratrace/bdump'
*.compatible='10.1.0.4.0'
*.control_files='/oracle/MDDB1LS/oradata/control01.ctl','/oracle/MDDB1LS/orad
ata/control02.ctl'
*.core_dump_dest='/oracle/MDDB1LS/oratrace/cdump'
*.db_block_size=8192
*.db_domain='WORLD'
*.db_file_multiblock_read_count=16
*.db_file_name_convert='/oracle/MDDB1','/oracle/MDDB1LS'
*.db_name='MDDB1'
*.db_unique_name='MDDB1LS'
*.db_recovery_file_dest='/oracle/MDDB1LS/flash_recovery_area'
*.db_recovery_file_dest_size=2147483648
*.dispatchers='(PROTOCOL=TCP) (SERVICE=MDDB1LSXDB)'
*.fal_client='MDDB1LS'
*.fal_server='MDDB1'
*.fast_start_mttr_target=300
*.instance_name='MDDB1LS'
*.job_queue_processes=4
*.log_archive_config='DG_CONFIG=(MDDB1,MDDB1LS)'
*.log_archive_dest_1='LOCATION=/oracle/MDDB1LS/oraarch/
VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=MDDB1LS'
*.log_archive_dest_2='SERVICE=MDDB1 LGWR ASYNC=61440 NET_TIMEOUT=30
VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=MDDB1'
*.log_archive_dest_state_1='ENABLE'
*.log_archive_dest_state_2='ENABLE'
*.log_archive_format='%t_%s_%r.dbf'
*.log_file_name_convert='/oracle/MDDB1','/oracle/MDDB1LS'
*.nls_length_semantics='CHAR'
*.open_cursors=300
*.pga_aggregate_target=149946368
*.processes=500
*.remote_login_passwordfile='EXCLUSIVE'
*.sessions=555
*.sga_target=218103808
*.standby_file_management='AUTO'
```

```
*.undo_management='AUTO'
*.undo_tablespace='UNDOTBS1'
*.user_dump_dest='/oracle/MDDB1LS/oratrace/udump'
```

3.1.4.8 Starten der Standby-Datenbank und Synchronisation

Standby:

```
cd
. MDDB1LS.env
sqlplus "/as sysdba"
SQL> startup nomount
SQL> alter database mount;
```

Primary:

```
SQL> alter system set log_archive_dest_state_2 = enable;
```

Standby:

```
SQL> alter database recover managed standby database disconnect from session;
```

Primary:

```
SQL> alter system switch logfile;
```

Standby:

Im Alert Log warten bis folgende Zeile auftritt:

```
Media Recovery Waiting for thread 1 sequence 653 (in transit)
```

3.2 Umwandeln der Physical Standby in eine Logical Standby

3.2.1 Ändern der init parameter der Primary Database

```
alter system set log_archive_dest_1='LOCATION=/oracle/MDDB1/oraarch/
VALID_FOR=(ONLINE_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=MDDB1';
alter system set log_archive_dest_2='SERVICE=MDDB1LS LGWR ASYNC=61440
NET_TIMEOUT=30 VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
DB_UNIQUE_NAME=MDDB1LS';
alter system set log_archive_dest_3='LOCATION=/oracle/MDDB1/oraarch2/
VALID_FOR=(STANDBY_LOGFILES,STANDBY_ROLE) DB_UNIQUE_NAME=MDDB1';
alter system set log_archive_dest_state_1=ENABLE;
alter system set log_archive_dest_state_3=ENABLE;
alter system set log_archive_dest_state_3=ENABLE;
alter system set undo_retention=3600;
```

3.2.2 Ändern der init Parameter der Logical Standby

```
SQL> alter system set log_archive_dest_1='LOCATION=/oracle/MDDB1LS/oraarch/
VALID_FOR=(ONLINE_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=MDDB1LS';
SQL> alter system set log_archive_dest_2='SERVICE=MDDB1 LGWR ASYNC=61440
NET_TIMEOUT=30 VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
DB_UNIQUE_NAME=MDDB1';
SQL> alter system set log_archive_dest_3='LOCATION=/oracle/MDDB1LS/oraarch2/
VALID_FOR=(STANDBY_LOGFILES,STANDBY_ROLE) DB_UNIQUE_NAME=MDDB1LS';
SQL> alter system set log_archive_dest_state_1=ENABLE;
SQL> alter system set log_archive_dest_state_3=ENABLE;
SQL> alter system set log_archive_dest_state_3=ENABLE;
SQL> alter system set undo_retention=3600;
```

3.2.3 Stoppen der physical Standby

```
SQL> shutdown immediate
```

3.2.4 Erzeugen eines logical standby controlfile auf primary

```
SQL> ALTER DATABASE CREATE LOGICAL STANDBY CONTROLFILE AS
'/tmp/logical_stby.ctl';
!cp /tmp/logical_stby.ctl /oracle/MDDB1LS/oradata/control01.ctl
!cp /tmp/logical_stby.ctl /oracle/MDDB1LS/oradata/control02.ctl
```

3.2.5 Starten der Logical Standby

```
SQL> startup mount;
SQL> alter database recover managed standby database;
SQL> alter database activate standby database;
```

3.2.6 Ändern des Database Name der logical Standby

```
SQL> shutdown immediate;
SQL> startup mount
exit;
nid TARGET=SYS@MDDB1LS DBNAME=MDDB1LS
cd $ORACLE_HOME/dbs
orapw file=orapwMDDB1LS password=xxx
sqlplus "/as sysdba"
SQL> create pfile from spfile;
exit;
Ändern von DB_NAME=MDDB1LS in initMDDBLS1.ora
sqlplus "/as sysdba
SQL> create SPFILE from pfile;
SQL> shutdown immediate;
SQL> startup mount;
SQL> alter database open resetlogs;
SQL> alter database rename global_name to MDDB1LS
```

3.2.7 Erzeugen des TEMP Tablespaces

```
SQL> alter tablespace TEMP add tempfile '/oracle/MDDB1LS/oradata/temp01.dbf'
size 30408704 reuse;
```

3.2.8 Start des SQL Apply

```
SQL> alter database start logical standby apply;
```

3.2.9 Standby Redo Logs

Primary:

```
SQL> alter database add standby logfile group 4
('/oracle/MDDB1/origlogA/stdby_04_01.log') size 20971520 reuse;
SQL> alter database add standby logfile group 5
('/oracle/MDDB1/origlogA/stdby_05_01.log') size 20971520 reuse;
SQL> alter database add standby logfile group 6
('/oracle/MDDB1/origlogA/stdby_06_01.log') size 20971520 reuse;
```

Standby:

```
SQL> alter database add standby logfile group 4
('/oracle/MDDB1LS/origlogA/stdby_04_01.log') size 20971520 reuse;
SQL> alter database add standby logfile group 5
('/oracle/MDDB1LS/origlogA/stdby_05_01.log') size 20971520 reuse;
SQL> alter database add standby logfile group 6
('/oracle/MDDB1LS/origlogA/stdby_06_01.log') size 20971520 reuse;
```

Achtung: Aufgrund von Bug 4038854 (gefixt in 10.1.0.5) dürfen die Standby Redo Log Groups nur ein Member haben!

Bei Produktiven Umgebungen muss geprüft werden, ob der Workload evtl. zusätzliche Standby-Redo Logs auf der Logical Standby benötigt.

3.2.10 Real-Time Apply

```
SQL> alter database stop logical standby apply;
SQL> alter database start logical standby apply immediate;
```

4 Administration / Wartung

4.1 Stop/Start von SQL Apply

4.1.1 Stoppen des SQL Apply

```
SQL> alter database stop logical standby apply;
```

4.1.2 Starten des SQL Realtime Apply

```
SQL> alter database start logical standby apply immediate;
```

4.2 Bereinigen von nicht mehr benötigten Logs

Nicht mehr für das SQL Apply benötigte Logs können hiermit gelöscht werden:

```
SQL> exec dbms_logstdby.purge_session;
SQL> select * from dba_logmnr_purged_log;
!rm <logs aus select>
```

5 Verwendung der Logical Standby

5.1 Zugriffskontrolle

Das Statement ALTER DATABASE GUARD steuert, welche Benutzer Zugriff auf die Logical Standby Tabellen haben.

- Kein Benutzer außer SYS darf Änderungen an Tabellen/Sequences vornehmen. (Read-Only)

```
SQL> ALTER DATABASE GUARD ALL;
```

- Kein Benutzer außer SYS darf DML / DDL Operationen auf Tabellen und Sequences anwenden, die mittels DataGuard repliziert werden. D.h., die von DataGuard replizierten Tabellen stehen als Read-Only, die restlichen Tabellen mit Read/Write zur Verfügung.

```
SQL> ALTER DATABASE GUARD STANDBY;
```

- Definiert herkömmlichen Schutz der Tabellen.

```
SQL> ALTER DATABASE GUARD NONE;
```

5.2 Transaction Consistency Modes

Der Transaction Consistency Modus beeinflusst die Performance und die Konsistenz des SQL Apply. Es gibt die folgenden 3 Modi:

5.2.1 Transaction Consistency FULL

Bei diesem Modus werden die Transaktionen in der exakten Reihenfolge appliziert, in welcher sie auf der Primary Database committed wurden. Es wird die geringste Apply-Performance dabei erreicht, jedoch ist die Datenbank konsistent für Reporting zu verwenden.

5.2.2 Transaction Consistency READ_ONLY

Der READ_ONLY Modus sollte nur verwendet werden, wenn „ALTER DATABASE GUARD ALL“ verwendet wird. In diesem Modus steht die Datenbank nur lesend zur Verfügung. Die Transaktionen werden in diesem Modus durcheinander committed, allerdings liefern SELECT Statements ein konsistentes Ergebnis basierend auf der letzten konsistenten SCN. Die Apply Performance ist höher als bei FULL.

5.2.3 Transaction Consistency NONE

Dieser Modus sollte nur verwendet werden, wenn temporär kein Reporting notwendig ist. In diesem Modus werden die Transaktionen in beliebiger Reihenfolge appliziert und SELECT Statements können inkonsistente Ergebnisse zurückliefern. Der Modus NONE sollte nur verwendet werden, wenn die Standby-Datenbank aufgrund von Netzwerk- oder Verfügbarkeitsproblemen weit hinter die Primary Datenbank zurückgefallen ist. Die Apply Performance ist hier am höchsten.

5.3 DDL Operationen auf replizierten Tabellen

Um z.B. zusätzliche Indizes auf replizierte Tabellen anzulegen, sind folgende Kommandos notwendig. Oracle empfiehlt, kein DML auf der Standby bei den replizierten Tabellen durchgeführt wird. Dies würde eine Inkonsistenz zwischen Primary und Standby bewirken.

```
SQL> ALTER DATABASE STOP LOGICAL STANDBY APPLY;
SQL> ALTER SESSION DISABLE GUARD;
SQL> ALTER TABLE .... ADD CONSTRAINT ....;
SQL> ALTER SESSION ENABLE GUARD;
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;
```

5.4 Ausschließen von Tabellen/Schemas von der Replikation

z.B. Ausschließen von den Tabellen MYTABLES% von Schema MYSCHEMA

```
SQL> EXEC DBMS_LOGSTDBY.SKIP('SCHEMA_DDL','MYSCHEMA','MYTABLES%');
SQL> EXEC DBMS_LOGSTDBY.SKIP('DML','MYSCHEMA','MYTABLES%');
```

z.B. Ausschließen eines ganzen Schemas

```
SQL> BEGIN
DBMS_LOGSTDBY.SKIP('SCHEMA_DDL', 'SMITH', '%', null);
DBMS_LOGSTDBY.SKIP('DML', 'SMITH', '%', null);
DBMS_LOGSTDBY.SKIP('TABLESPACE', null, null, 'SMITH.PROTECT_BONES');
END;
/
```

5.5 Manuelles replizieren einer Tabelle aufgrund von Manipulation auf der Standby Datenbank

Auf der Standby-Datenbank:

```
SQL> create database link MDDB1 connect to system identified by xxx using
'MDDB1';

Database link created.

SQL> select count(*) from all_objects@MDDB1;

  COUNT(*)
----------
     47219

SQL> alter database stop logical standby apply;

Database altered.

SQL>  execute dbms_logstdby.unskip('DML','QA','LOGICAL_STANDBY_TEST');

PL/SQL procedure successfully completed.

SQL> exec
dbms_logstdby.instantiate_table('QA','LOGICAL_STANDBY_TEST','MDDB1');

PL/SQL procedure successfully completed.

SQL> alter database start logical standby apply immediate;

Database altered.
```

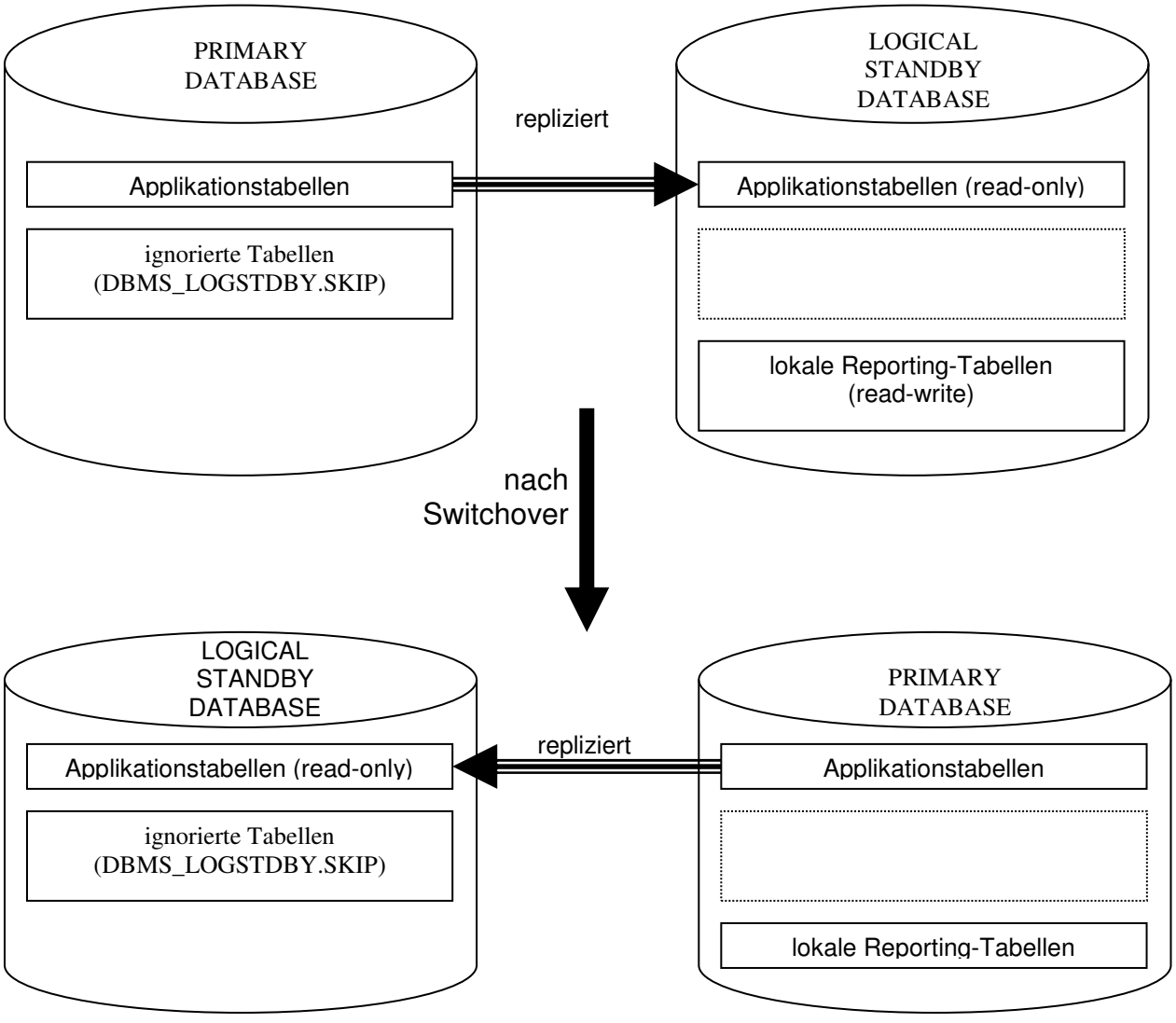
6 Role Management (Failover/Switchover)

6.1 Switchover

Beim Switchover wird die Logical Standby zur Primary Database und die bislang Primary Database wird zur Logical Standby. Es findet also ein Rollentausch statt. Dieser Vorgang ist beliebig oft durchführbar. Zusätzlich zu den hier aufgelisteten Aktionen muss natürlich die virtuelle IP-Adresse sowie der Applikations-Listener auf die jeweils Primary Database gewechselt werden.

Die folgende Darstellung zeigt die Inhalte der Datenbanken nach einem Switchover.

- Tabellen, die auf der ursprünglichen Primary nicht repliziert werden, stehen nach dem Switchover nur auf der Standby zur Verfügung.
- Lokale Reporting-Tabellen auf der ursprünglichen Standby stehen nach dem Switchover erst dann auf der neuen Standby zur Verfügung, wenn sie einmalig manuell über DBMS_LOGSTDBY.INSTANTIATE_TABLE repliziert und dann von der Replikation mittels DBMS_LOGSTDBY.SKIP exkludiert wurden.



6.1.1 Switchover-Commandos

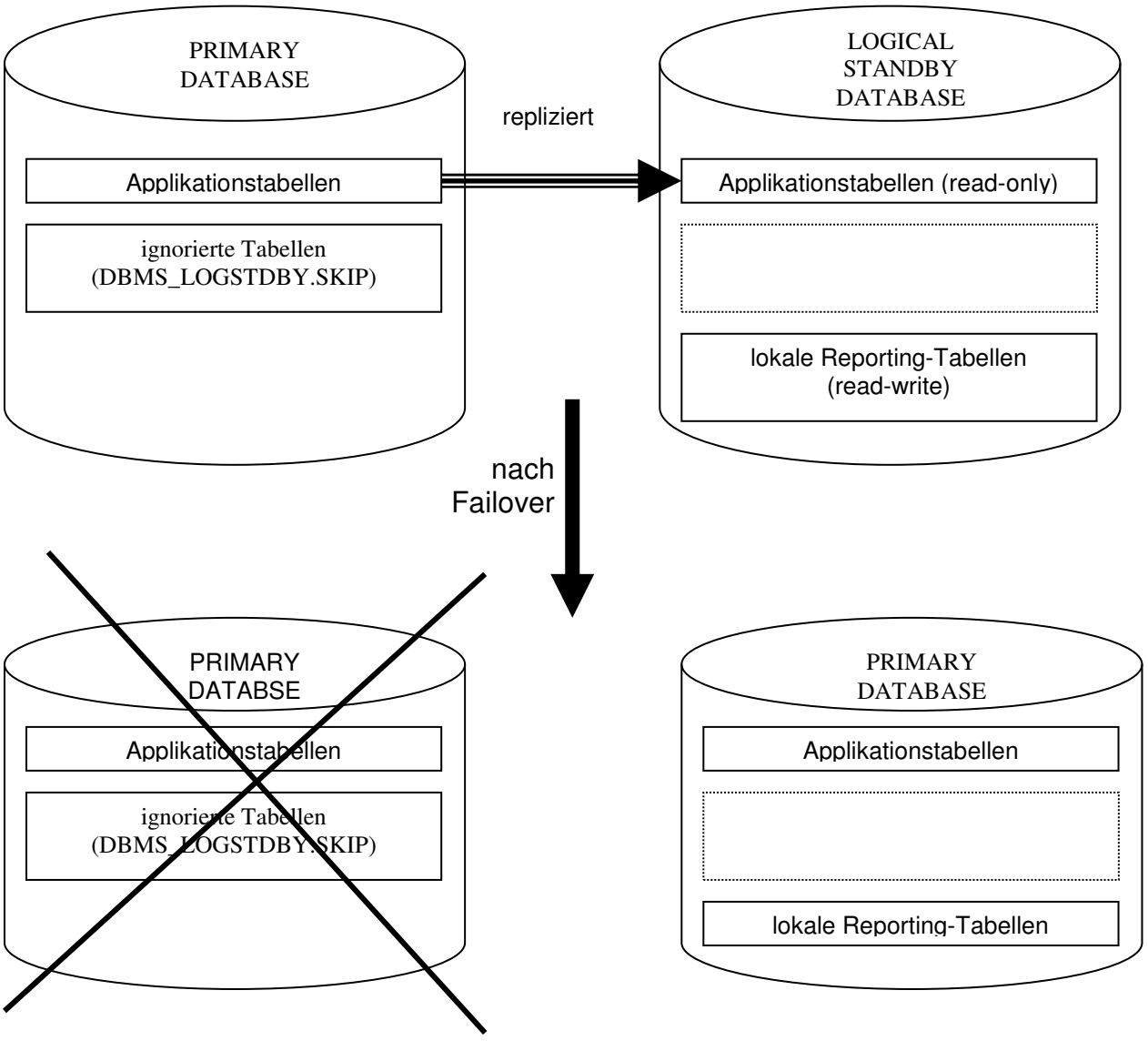
Physical Database (z.B. MDDB1)	Logical Standby (MDDB1LS)
SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE; TO_STANDBY, TO LOGICAL STANDBY, SESSIONS ACTIVE deuten darauf hin, dass ein Switchover möglich ist. SQL> ALTER DATABASE PREPARE TO SWITCHOVER TO LOGICAL STANDBY;	
	SQL> ALTER DATABASE PREPARE TO SWITCHOVER TO PRIMARY;
SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE; erfolgreich, wenn TO LOGICAL STANDBY SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO LOGICAL STANDBY;	
	SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE; SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY;	

6.2 Failover

Die Failover Operation ist durchzuführen, wenn es sich um einen permanenten Ausfall der Primary Datenbank handelt. In diesem Fall wird die Logical Standby zur Primary Database und die vorherige Primary Database muss, sobald sie wieder verfügbar ist, neu aufgebaut werden, um wieder in dem DataGuard Verbund zu gelangen.

Die folgende Darstellung zeigt der Inhalt der Datenbank nach dem Failover:

- Tabellen, die auf der ursprünglichen Primary von der Replikation exkludiert wurden, stehen nach dem Failover nicht zur Verfügung
- lokale Reporting-Tabellen auf der ursprünglichen Standby-Database stehen nach dem Failover auf der Primary zur Verfügung.
- Zu Beachten ist allerdings, was mit den lokalen Reporting-Tabellen auf der Standby-Seite passiert, wenn die logical Standby aufgrund eines Hardware-Problems nicht mehr verfügbar ist.



6.2.1 Failover-Durchführung

Die folgenden Schritte sind für eine Failover Operation durchzuführen:

- Es muss geprüft werden, welche Redo-Daten fehlen. Grundsätzlich dürfte beim LGWR Transport Modus keine Lücke entstehen, allerdings könnten Redo-Daten, die sich noch im LNS Buffer (LGWR ASYNC) befunden haben, verloren gehen.

  ```
  SQL> select applied_scn, newest_scn, sysdate, applied_time from
  dba_logstdby_progress;
  ```

  Wenn applied_scn und newest_scn identisch sind, wurden alle generierten Redo-Daten appliziert. Der zeitliche Unterschied kann über die Spalten sysdate und applied_time gemessen werden.

- Falls der SQL Apply Modus nicht aktiv war, kann er mit folgendem Befehl bis zum letzten vorhandenen Punkt applizieren:

  ```
  SQL> ALTER DATABASE START LOGICAL STANDBY APPLY NODELAY FINISH;
  ```

- Nun kann die Logical Standby Database zur Primary aktiviert werden:

  ```
  SQL> ALTER DATABASE STOP LOGICAL STANDBY APPLY;

  SQL> ALTER DATABASE ACTIVATE LOGICAL STANDBY DATABASE;
  ```

- Full-Backup der neuen Primary Datenbank

7 Monitoring - SQL (auf Logical Standby)

7.1 Funktionstest

Primary:

```
SQL> create table qa.logical_standby_test (a number, b varchar2(1000));
SQL> alter table qa.logical_standby_test  add primary key(a);
SQL> insert into qa.logical_standby_test select object_id, object_name from
dba_objects;
SQL> commit;
```

Standby:

```
SQL> select count(*) from qa.logical_standby_test;
```

Nach ein paar Sekunden (~ 5-30 Sekunden) sollten die Datensätze auf der Standby verfügbar sein.

7.2 Verifizieren, dass die Primary Archivelogs registriert wurden

```
SQL> alter session set nls_date_format = 'DD.MM.YY HH24:MI:SS';
SQL> select sequence#, first_time, next_time, dict_begin, dict_end from
dba_logstdby_log order by sequence#;
```

7.3 Verifizieren, dass Redo-Daten erfolgreich appliziert werden

```
SQL> select name, value from v$logstdby_stats where name = 'coordinator
state';
```

```
INITIALIZING: SQL Apply wird initialisiert

APPLYING:     SQL wird appliziert
```

7.4 Anzeigen von SQL Apply Aktivität

```
SQL> select type, high_scn, status from v$logstdby;

TYPE                         HIGH_SCN STATUS
---------------------------- ---------- ----------------------------------
COORDINATOR                   5489976 ORA-16116: no work available
READER                        5489978 ORA-16117: processing
BUILDER                       5489978 ORA-16116: no work available
PREPARER                      5489977 ORA-16116: no work available
ANALYZER                      5489978 ORA-16120: dependencies being
                                      computed for transaction at SCN 0x0000.0053c53a
APPLIER                       5489976 ORA-16116: no work available
APPLIER                       5489977 ORA-16117: processing
APPLIER                       5489857 ORA-16116: no work available
APPLIER                       5450943 ORA-16116: no work available
APPLIER                       5447146 ORA-16116: no work available

10 rows selected.
```

7.5 Anzeigen der Differenz zwischen Logical Standby und Primary Database

```
SQL> select applied_scn, newest_scn from dba_logstdby_progress;

APPLIED_SCN NEWEST_SCN
----------- ----------
    5490030    5490036
```

7.6 Status der Standby Redo Logs

```
SQL> select group#, sequence# as seq, bytes, used, archived as ARC, status,
first_change# as first_scn, first_time, last_change# as last_scn, last_time from
v$standby_log

GROUP# SEQ    BYTES     USED ARC  STATUS     FIRST_SCN FIRST_TIME LAST_SCN LAST_TIM
------ --- -------- -------- ---- ---------- --------- ---------- ---------- ------
     4 707 20971520  8625152 YES  ACTIVE     5487645   07-OCT-05  5490653 07-OCT-05
     5   0 20971520      512 NO   UNASSIGNED       0                     0
     6   0 20971520      512 YES  UNASSIGNED       0                     0
```

7.7 Logical Standby Events

```
SQL> select * from sys.dba_logstdby_events order by event_time desc,
commit_scn;
```

7.8 Logical Standby Statistics

```
SQL> select name, value from v$logstdby_stats where name like 'coordinator%' or
name like 'transactions%';
```

7.9 Aktueller Backlog prüfen

Mit folgendem SQL Statement kann angezeigt werden, an welchem Archivelog der Standby Apply Mechanismus gerade arbeitet.

```
SQL> select first_scn, first_time, next_time from dba_logmnr_log where
(select applied_scn from dba_logstdby_progress) between first_scn and next_scn;
```

8 Wait Events für Logical Standby

- LNS wait on ATTACH: Monitors Amount of Time spent by all network servers to spawn an RFS connection
- LNS wait on SENDREQ: Monitors Amount of Time spent by all network servers to write received redo data to disk as well as open and close the remote archived redo log files.
- LNS wait on DETACH: Monitors Amount of Time spent by all network servers to delete an RFS connection.
- LGWR wait on full LNS buffer: Monitors Amount of Time spent by the LGWR process waiting for the network server (LNS) to free up ASYNC buffer space. If buffer space has not been freed in a reasonable amount of time, availability of the primary database is not compromised by allowing the ARCn process to transmit the redo data.
- LGWR wait on LNS
- LNS wait on LGWR
- LGWR-LNS wait on channel

9 Troubleshooting

9.1 SQL Apply Abbruch nach create tablespace

Laut MetaLink NoteID 246937.1 wird db_file_name_convert bei Logical Standby nicht überstützt. Wenn also Primary und Logical Standby auf einem Server betrieben werden, dann sind neue Tablespaces manuell auf der Standby anzulegen.

Die folgende Fehlermeldung erscheint im Alert Log:

```
LOGSTDBY XID 0x0003.000.00003aab, Thread 1, RBA 0x0476.00001f0e.80
Wed Oct 19 16:48:59 2005
create tablespace soe datafile '/oracle/MDDB1/oradata/soe.dbf'
size 256M reuse
autoextend on next 2m maxsize unlimited
extent management local uniform size 200k
segment space management auto
Wed Oct 19 16:48:59 2005
ORA-1119 signalled during: create tablespace soe datafile
'/oracle/MDDB1/ora...
LOGSTDBY stmt: create tablespace soe datafile
'/oracle/MDDB1/oradata/soe.dbf'
size 256M reuse
autoextend on next 2m maxsize unlimited
extent management local uniform size 200k
segment space management auto
LOGSTDBY status: ORA-01119: error in creating database file ''
LOGSTDBY Apply process P004 pid=25 OS id=11336 stopped
Wed Oct 19 16:48:59 2005
Errors in file /oracle/MDDB1LS/oratrace/bdump/MDDB1ls_lsp0_11324.trc:
ORA-12801: error signaled in parallel query server P004
ORA-01119: error in creating database file
'/oracle/MDDB1/oradata/soe.dbf'
ORA-27086: unable to lock file - already in use
SVR4 Error: 11: Resource temporarily unavailable
Additional information: 8
Additional information: 521
LOGSTDBY Analyzer process P003 pid=24 OS id=11334 stopped
LOGSTDBY Apply process P006 pid=27 OS id=11340 stopped
LOGSTDBY Apply process P007 pid=28 OS id=11342 stopped
LOGSTDBY Apply process P005 pid=26 OS id=11338 stopped
LOGSTDBY Apply process P008 pid=29 OS id=11344 stopped
Wed Oct 19 16:49:00 2005
```

Die Lösung besteht darin, den Tablespace manuell anzulegen und die Transaktion zu überspringen.

```
sys@MDDB1LS>  EXECUTE DBMS_LOGSTDBY.GUARD_BYPASS_ON;

PL/SQL procedure successfully completed.

sys@MDDB1LS> create tablespace soe datafile
'/oracle/MDDB1LS/oradata/soe.dbf'  size 256M reuse autoextend on next 2m
maxsize unlimited extent management local uniform size 200k segment space
management auto;

Tablespace created.

sys@MDDB1LS> EXECUTE DBMS_LOGSTDBY.GUARD_BYPASS_OFF;

SQL> SELECT * FROM DBA_LOGSTDBY_EVENTS
     WHERE EVENT_TIME = (SELECT MAX(EVENT_TIME) FROM DBA_LOGSTDBY_EVENTS)
```

```
EVENT_TIME          CURRENT_SCN COMMIT_SCN     XIDUSN     XIDSLT
XIDSQN
------------------- ----------- ---------- ---------- ---------- ---------
-
EVENT
--------------------------------------------------------------------------------
-------
STATUS_CODE
-----------
STATUS
--------------------------------------------------------------------------------
-------
19-OCT-05              7203329    7203331          3          2
15026
create tablespace soe datafile '/oracle/MDDB1/oradata/soe.d
bf'
size 256M reuse
       1119

EVENT_TIME          CURRENT_SCN COMMIT_SCN     XIDUSN     XIDSLT
XIDSQN
------------------- ----------- ---------- ---------- ---------- ---------
-
EVENT
--------------------------------------------------------------------------------
-------
STATUS_CODE
-----------
STATUS
--------------------------------------------------------------------------------
-------
ORA-01119: error in creating database file ''


sys@MDDB1LS> EXECUTE DBMS_LOGSTDBY.SKIP_TRANSACTION(3, 2, 15026);

PL/SQL procedure successfully completed.

sys@MDDB1LS> alter database start logical standby apply immediate;

Database altered.
```

9.2 Probleme mit Recycle Bin

Aufgrund von zwei Bugs (2988193 und 4349720) muss das Feature Recycle-Bin sowohl auf der Primary als auch auf der Logical Standby ausgeschalten werden. Die beiden Bugs sind lt. MetaLink in Version 10.2 korrigiert.

```
SQL> alter system set “_recyclebin” = false;
```

9.3 ORA-1403: No Data Found

ORA-1403: Wenn Daten einer replizierten Tabelle auf der Standby-Seite manipuliert wurden, kann es zu "ORA-1043: No Data Found" kommen. In diesem Fall kann mittels Flashback geprüft werden, durch welches SQL Statement die Werte manipuliert wurden. Unter Umständen kann dann manuell die Manipulation rückgängig gemacht werden. Die zweite Möglichkeit ist, die Tabelle mit DBMS_LOGSTDBY.SKIP auszuschließen und anschließend neu zu replizieren (dbms_logstdby.instantiate_table).

9.4 LCR Cache zu klein

Falls der Standby-Datenbank zuwenig LCR Cache zur Verfügung steht, muss sie die Daten in den SYSAUX Tablespace rauspagen. Falls der Wert stetig steigt, sollte der Logical Standby Parameter MAX_SGA vergrößert werden.

```
SQL> select value from v$logstdby_stats  where name like '%paged%';
```

9.5 Long Running Transactions

```
Mon Feb 17 14:40:15 2003
WARNING: the following transaction makes no progress
WARNING: in the last 30 seconds for the given message!
WARNING: xid =
0x0016.007.000017b6 cscn = 1550349, message# = 28, slavid = 1
knacrb: no offending session found (not ITL pressure)
```

Diese Alert-Fehlermeldung deutet auf Full Table Scans hin. Es muss geprüft werden, ob ein Primary Key oder ein Unique Index auf der replizierten Tabelle vorhanden ist. Die Tabelle kann mit dieser Query ermittelt werden:

```
SQL> SELECT SAS.SERVER_ID
          , SS.OWNER
          , SS.OBJECT_NAME
          , SS.STATISTIC_NAME
          , SS.VALUE
     FROM V$SEGMENT_STATISTICS SS
          , V$LOCK L
          , V$STREAMS_APPLY_SERVER SAS
     WHERE SAS.SERVER_ID = &SLAVE_ID
        AND L.SID = SAS.SID
        AND L.TYPE = 'TM'
        AND SS.OBJ# = L.ID1;
```

9.6 Zu replizierende Daten erscheinen nicht in der Standby Datenbank

- Als erstes sind das Alert Log der Primary und die Parameter log_archive_dest_ log_archive_dest_state_ zu prüfen.

Primary:

```
SQL> select dest_id, status, destination, error from v$archive_dest;
```

- Dann sollte auf der Standby Datenbank geprüft werden, ob SQL Apply aktiv ist. Bei mehrmaligem Aufruf dieses Statements sollte die high_scn sich ändern.

```
SQL> select pid, type, status, high_scn from v$logstdby;
```

- Die Tabelle v$logstdby_stats zeigt die Werte der einzelnen Settings sowie Statistiken. Hier kann ermittelt werden, ob die Transaktionen sich aufstauen. (transactions applied vs. transactions read)
- Die Tabelle v$logstdby muss Zeilen enthalten. Ansonsten ist SQL Apply nicht aktiv.

- Das folgende Statement zeigt die SQL Statements an, die gerade bearbeitet werden. Über den SQL Hash Value kann in der Tabelle v$sql_plan der Ausführungsplan analysiert werden. Ebenfalls kann in der Tabelle v$session_wait geprüft werden, worauf die Session gerade wartet.

```
SQL> select ls.serial# "Apply Process"
         , sas.state "State"
         , sas.sid SID
         , s.sql_address "SQL Address"
         , s.sql_hash_value "SQL Hash Value"
         , sa.sql_text "SQL Text"
     from v$logstdby ls
       , v$streams_apply_server sas
       , v$session s
      , v$sqlarea sa
    where ls.type = 'APPLIER'
    and sas.state != 'IDLE'
    and sas.serial# = ls.serial#
    and s.sid = sas.sid
    and sa.address (+) = s.sql_address
    and sa.hash_value (+) = s.sql_hash_value;
```

9.7 Schlechte SQL Apply Performance aufgrund von Full Table Scans

Mit dem folgenden SQL Statement können SQL Apply Statements angezeigt werden, die Full-Table-Scans durchführen. Diese können verhindert werden, indem die replizierten Tabellen Primary Keys / Unique Indizes verwenden.

```
SQL> select sql_text, operation, options, object_name, cost
from v$sql_plan, v$session, v$logstdby, v$sql
where v$sql_plan.hash_value = v$session.sql_hash_value
and v$sql_plan.hash_value = v$sql.hash_value
and v$session.serial# = v$logstdby.serial#
and v$logstdby.status_code=16113
and v$sql_plan.options = 'FULL';
```

9.8 Nicht unterstützte DML und DDL Statements

Bei Auftreten von nicht unterstützten DML/DDL Statements, z.B. durch nicht unterstützte Datentypen, stoppt die SQL Apply Operation. IN der Tabelle DBA_LOGSTDBY_EVENTS kann der Fehler abgefragt werden.

```
SQL> select xidusn, xidslt, xidsqn, status, status_code from
dba_logstdby_events
     where event_time =
           (select max(event_time)
            from dba_logstdby_events);
```

Beim Problemen mit Datafiles, Tablespaces, etc. kann das Problem manuell behoben werden und SQL Apply neu gestartet werden. Bei Problemen mit Constraint Violations, etc. kann die Transaktion über DBMS_LOGSTDBY.SKIP_TRANSACTION übergangen werden. Beim Start von SQL Apply wird diese Transaktion dann einfach ignoriert.

9.9 Catch-Up Modus, wenn SQL Apply Modus im Verzug ist

Falls die Standby Database beim SQL Apply weit hinter der Primary Database zurück ist, kann man den Transactions-Consistency Modus von FULL (Default) auf NONE oder READ_ONLY ändern.

9.9.1 Beispiel

```
SQL> alter database stop logical standby apply;

SQL> exec dbms_logstdby.apply_set('TRANSACTION_CONSISTENCY','NONE');

PL/SQL procedure successfully completed.

SQL> alter database start logical standby apply immediate;
```

wenn die Standby-Datenbank den Rückstand aufgeholt hat:

```
SQL> alter database stop logical standby apply;

SQL> exec dbms_logstdby.apply_set('TRANSACTION_CONSISTENCY','FULL');

PL/SQL procedure successfully completed.

SQL> alter database start logical standby apply immediate;
```

9.10 ORA-04031: Shared Pool Bereich für SQL Apply zu klein

Die folgenden Fehlermeldungen deuten auf einen zu kleinen Cache Bereich für SQL Apply hin. Der Default beträgt 30M.

```
Wed Oct 12 23:33:46 2005
krvxerpt: Errors detected in process 24, role preparer.
Wed Oct 12 23:33:46 2005
krvxmrs: Leaving by exception: 4031
Wed Oct 12 23:33:46 2005
Errors in file /oracle/MDDB1LS/oratrace/bdump/MDDB1ls_p002_24544.trc:
ORA-04031: unable to allocate 4120 bytes of shared memory ("shared
pool","unknown object","Logminer LCR c","krvufa")
LOGSTDBY status: ORA-04031: unable to allocate 4120 bytes of shared memory
("shared pool","unknown object","Logminer LCR c","krvufa")
Wed Oct 12 23:33:46 2005
Errors in file /oracle/MDDB1LS/oratrace/bdump/MDDB1ls_p005_24550.trc:
ORA-00603: ORACLE server session terminated by fatal error
ORA-00604: error occurred at recursive SQL level 1
ORA-04031: unable to allocate 4096 bytes of shared memory ("shared
pool","select name,online$,contents...","Typecheck heap","kgghteInit")
ORA-00604: error occurred at recursive SQL level 1
ORA-04031: unable to allocate 4096 bytes of shared memory ("shared
pool","select name,online$,contents...","Typecheck heap","kgghteInit")
ORA-00604: error occurred at recursive SQL level 1
ORA-04031: unable to allocate 4096 bytes of shared memory ("shared
pool","select proc, nvl((select nam...","Typecheck heap","kgghteInit")
ORA-04031: unable to allocate 4096 bytes of shared memory ("shared
pool","select /*+ rule */ bucket_cn...","Typecheck heap","kgghteInit")
Wed Oct 12 23:33:46 2005
krvxerpt: Errors detected in process 24, role preparer.
Wed Oct 12 23:33:46 2005
krvsles: unhandled failure logging error 604
```

```
Wed Oct 12 23:33:46 2005
Errors in file /oracle/MDDB1LS/oratrace/bdump/MDDB1ls_p002_24544.trc:
ORA-00604: error occurred at recursive SQL level 2
ORA-04031: unable to allocate 4096 bytes of shared memory ("shared
pool","select obj#,type#,ctime,mtim...","Typecheck heap","kgghteInit")
ORA-04031: unable to allocate 4120 bytes of shared memory ("shared
pool","unknown object","Logminer LCR c","krvufa")
Wed Oct 12 23:33:46 2005
(krvxmrs) - error callback failed with status 604
Wed Oct 12 23:33:47 2005
Errors in file /oracle/MDDB1LS/oratrace/bdump/MDDB1ls_lsp0_24574.trc:
ORA-12801: error signaled in parallel query server P002
ORA-00604: error occurred at recursive SQL level 2
ORA-04031: unable to allocate 4096 bytes of shared memory ("shared
pool","select obj#,type#,ctime,mtim...","Typecheck heap","kgghteInit")
ORA-04031: unable to allocate 4120 bytes of shared memory ("shared
pool","unknown object","Logminer L
Wed Oct 12 23:33:47 2005
TLCR process death detected. Shutting down TLCR
LOGSTDBY Apply process P006 pid=28 OS id=24552 stopped
Wed Oct 12 23:33:47 2005
logminer process death detected, exiting logical standby
LOGSTDBY Analyzer process P003 pid=25 OS id=24546 stopped
LOGSTDBY Apply process P007 pid=29 OS id=24554 stopped
LOGSTDBY Apply process P008 pid=30 OS id=24556 stopped
LOGSTDBY Apply process P004 pid=26 OS id=24548 stopped
```

Er kann vergrößert werden mit diesen Kommandos:

```
SQL> alter database stop logical standby apply;
SQL> exec dbms_logstdby.apply_set('MAX_SGA',50);
SQL> alter database start logical standby apply immediate;
```

10 Logical Standby Kompatibilität

10.1 Inkompatible Tabellen

```
select owner, table_name, column_name, data_type, attributes from
dba_logstdby_unsupported

OWNER           TABLE_NAME                     COLUMN_NAME
DATA_TYPE                      ATTRIBUTES
--------------- ------------------------------ ------------------------------ -----
--------------------------- --------------------
MDSYS           SDO_GEOM_METADATA_TABLE        SDO_DIMINFO
SDO_DIM_ARRAY
MDSYS           SDO_INDEX_METADATA_TABLE       SDO_ROOT_MBR
SDO_GEOMETRY
MDSYS           SDO_TOPO_METADATA_TABLE        TOPO_GEOMETRY_LAYERS
SDO_TOPO_GEOMETRY_LAYER_ARRAY
MDSYS           CS_SRS                         CS_BOUNDS
SDO_GEOMETRY
MDSYS           USER_CS_SRS                    CS_BOUNDS
SDO_GEOMETRY
MDSYS           SDO_STYLES_TABLE               GEOMETRY
SDO_GEOMETRY
FLOWS_020000    WWV_FLOW_JOB_BIND_VALUES       JOB
NUMBER                         IOT with Overflow
FLOWS_020000    WWV_FLOW_JOB_BIND_VALUES       NAME
VARCHAR2                       IOT with Overflow
FLOWS_020000    WWV_FLOW_JOB_BIND_VALUES       VALUE
VARCHAR2                       IOT with Overflow
FLOWS_020000    WWV_FLOW_RT_DISTRIBUTIONS$     B_FILE                         BFILE

10 rows selected.
```

10.2 Tabellen ohne Primary Key / Unique Index

```
select owner, table_name, bad_column from dba_logstdby_not_unique where table_name
NOT IN (SELECT TABLE_NAME FROM DBA_LOGSTDBY_UNSUPPORTED) order by 1,3,2
  2  ;

OWNER           TABLE_NAME                     BAD_
--------------- ------------------------------ ----
FLOWS_020000    WWV_FLOW_ACTIVITY_LOG1$        N
FLOWS_020000    WWV_FLOW_ACTIVITY_LOG2$        N
FLOWS_020000    WWV_FLOW_ACTIVITY_LOG_NUMBER$  N
FLOWS_020000    WWV_FLOW_CLICKTHRU_LOG$        N
FLOWS_020000    WWV_FLOW_DATA_LOAD_BAD_LOG     N
FLOWS_020000    WWV_FLOW_DATA_LOAD_UNLOAD      N
FLOWS_020000    WWV_FLOW_DUAL100               N
FLOWS_020000    WWV_FLOW_FILE_OBJECTS$PART     N
FLOWS_020000    WWV_FLOW_MAIL_LOG              N
FLOWS_020000    WWV_FLOW_PURGED_SESSIONS$      N
FLOWS_020000    WWV_FLOW_SHORTCUT_USAGE_MAP    N
FLOWS_020000    WWV_FLOW_UPGRADE_PROGRESS      N
FLOWS_020000    WWV_FLOW_BANNER                Y
MDSYS           OGIS_GEOMETRY_COLUMNS          N
MDSYS           SDO_DATUMS                     N
MDSYS           SDO_ELLIPSOIDS                 N
MDSYS           SDO_PROJECTIONS                N
```

10.2.1 Information zur Spalte BAD_COLUMN:

- N ... Die Tabelle hat zwar keinen Primary Key /Unique Index, Oracle kann sie nur mittels Full Table Scans replizieren. Allerdings ist es aus Performance-Gründen sehr zu empfehlen, einen Primary Key / Unique Index anzulegen.

- Y ... Es gibt CLOB/BLOB Columns und es besteht keine Eindeutigkeit in den restlichen Spalten. Oracle ist es nicht möglich, diese Tabellen zu replizieren.

11 Referenzen

- MetaLink Note: 312434.1 Oracle10g Data Guard SQL Apply Troubleshooting
- Oracle10g Data Guard Concepts and Administration Guide
- Oracle10g Packages and Types Reference Guide (DBMS_LOGSTDBY)