UniCredit

Business Integrated Solutions

DataGuard in Practice

Matthias Mann

Nuremberg, 22. November 2012

AGENDA

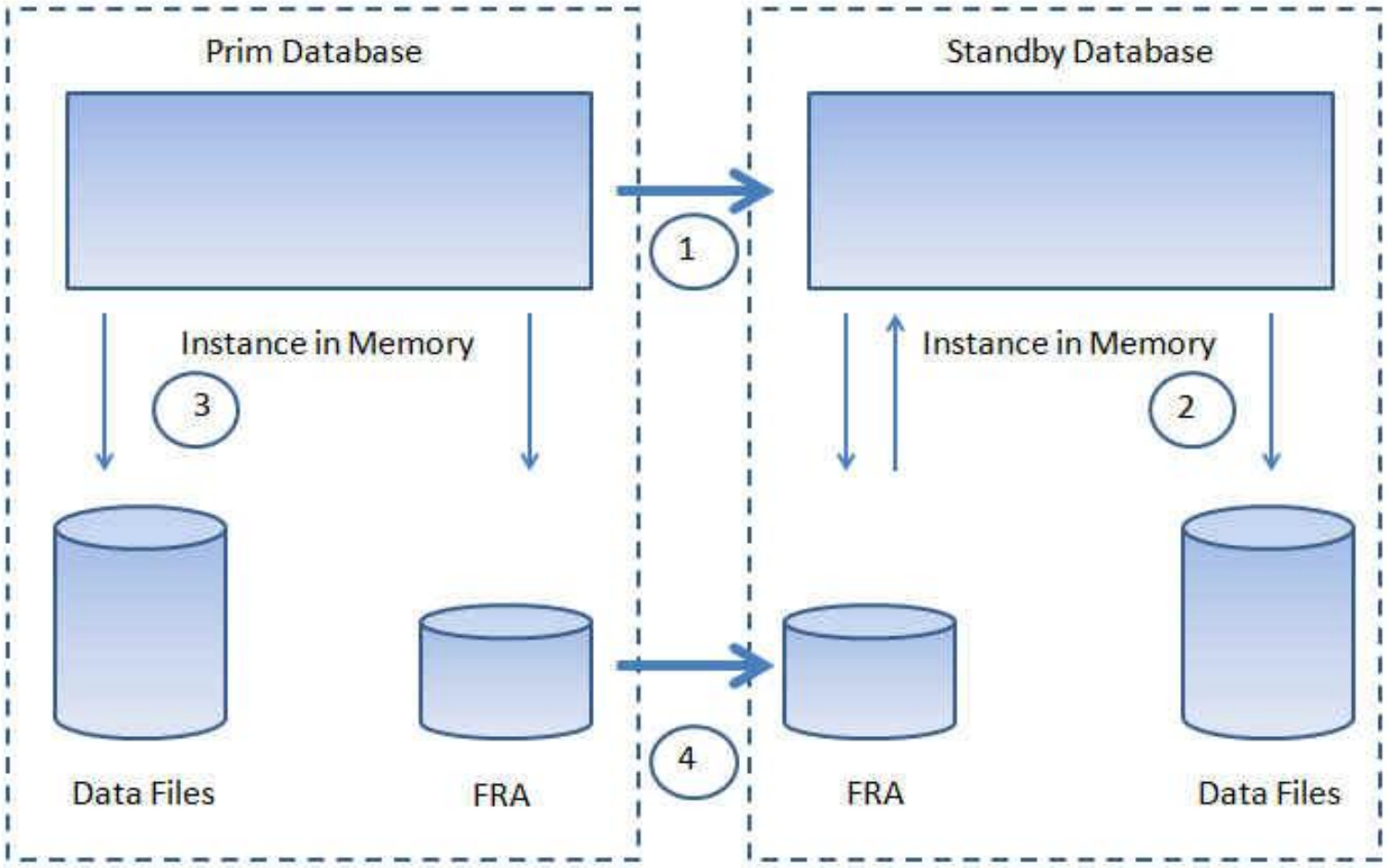
- DataGuard – Overview and Terms
- Clone from Active Database
- Active DataGuard Reader Farm
  - Use Case: Main and Satellite Application
  - Health Monitor
  - Standby Statspack
- Snapshot Standby Database
  - Use case : Day – 1 Data
  - Control of Standby Open Mode

What is DataGuard ? What is this talk about ?

- one primary and (multiple) standby databases
- database duplication by the Oracle kernel
- physical (block level) or logical (SQL apply)
- integrated with RMAN, Flashback, ...
- Oracle's suggested Disaster Recovery (DR) solution
- part of Maximum Availability Architecture (MAA)
- switchover: graceful role transition between primary and standby (purposely)
- failover: role transition due to unplanned event
- broker framework for easy administration

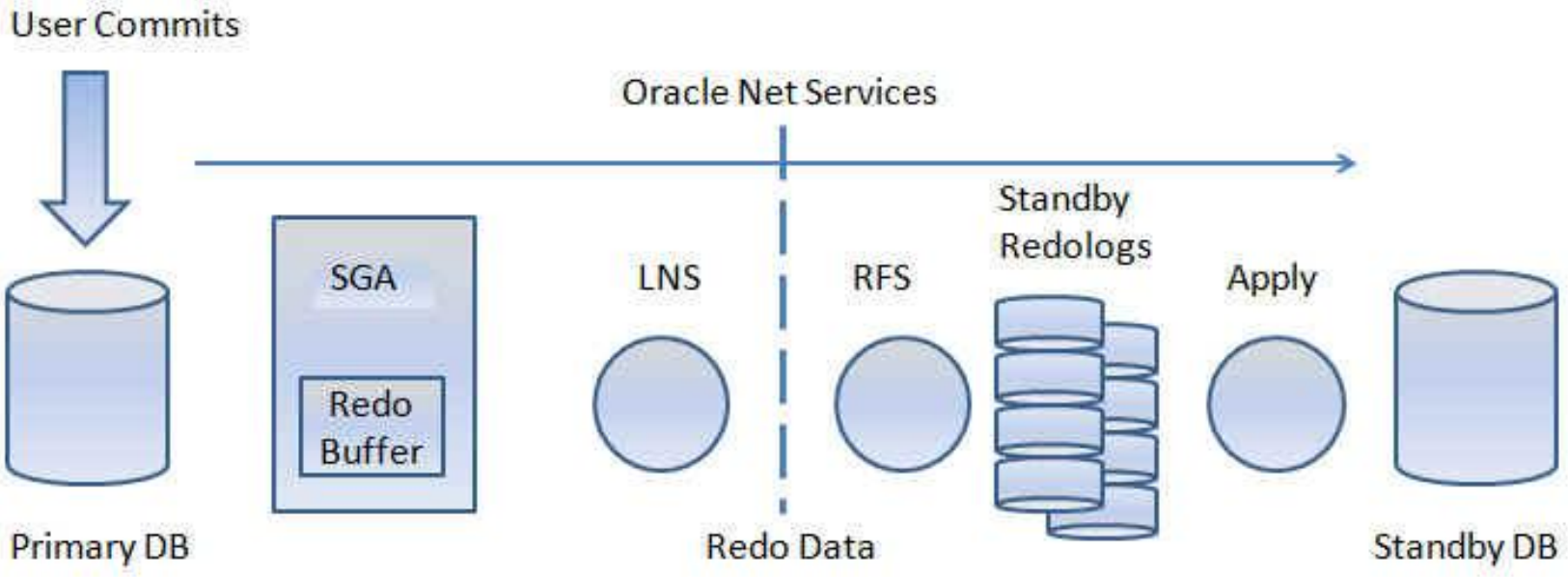
- we consider only physical standby
- we focus NOT on DR scenarios but on the use of DataGuard features in daily business

DataGuard - Architecture



1 -- redo transmission, 2 – redo validation and apply,
3 -- update of primary database files, 4 – automatic gap resolution

Architecture of Redo Transport



LNS – Log Network Server process
RFS – Remote Fileserver process

Standby Types

	Physical Standby	Active DataGuard	Snapshot Standby
Open Mode	mount	read only	read / write
License	EE	EE + ADG	EE
Redo Transport	yes	yes	yes
Redo Apply	yes	yes	no

Clone from Active Database

- easy to use method for creation of a standby database

1. Restore using the target database

```
connect target sys/***@source;
connect auxiliary sys/***@destination;
run {
  allocate channel prmy1 type disk;
  allocate auxiliary channel stdby1 type disk;
```

Clone from Active Database


```
duplicate target database for standby from active database
spfile
  parameter_value_convert 'pattern','pattern',...
  set db_unique_name='...'
  set instance_name='...'
  set log_archive_dest_1='location=USE_DB_RECOVERY_FILE_DEST'
  set db_file_name_convert='pattern','pattern'
  set log_file_name_convert='pattern','pattern'
  set control_files='.../control01.ctl','.../control02.ctl'
  set standby_file_management = 'AUTO'
  set db_create_file_dest='...'
  set db_create_online_log_dest_1='...'
  set db_recovery_file_dest='...'
  set db_recovery_file_dest_size='...'
  set dg_broker_config_file1='...'
  set dg_broker_config_file2='...'
  ...
  }
```

Clone from Active Database

2. Inclusion in the broker configuration and start of managed recovery

```
SQL>    alter database mount standby database;
DGMGRL> add database destination as connect identifier
        is 'destination' maintained as physical;
DGMGRL> enable database destination;
```

Use Case: Main and Satellite Application

Project

- implementation of the i-flex ® core banking system (main) together with a set of satellites (e.g. general ledger system)

Requirement

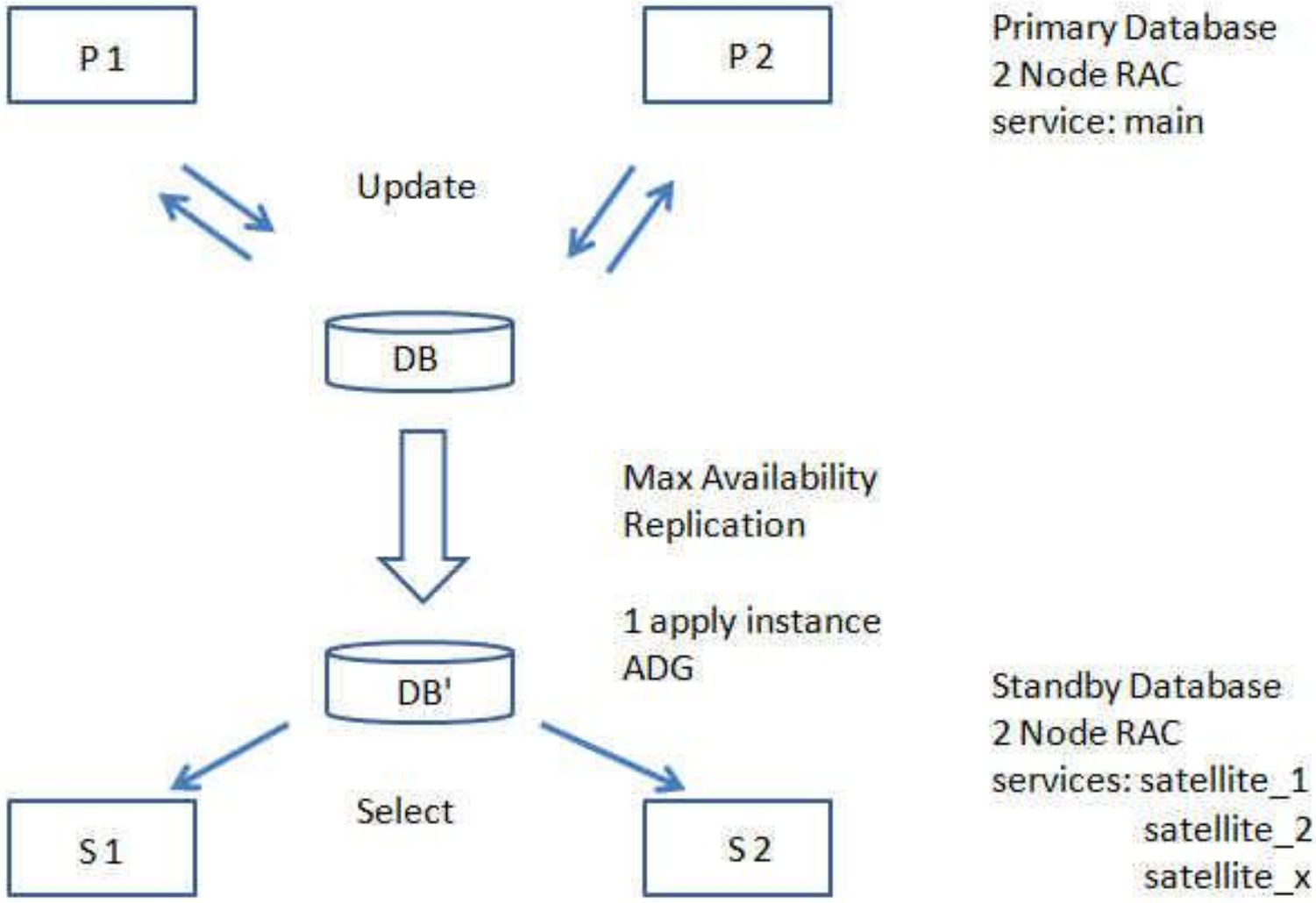
- technical isolation of main and satellite applications
- satellites MUST NOT use resources needed by the main application

Solution

- use standby database in ADG mode for queries performed by the satellite applications

Use Case: Main and Satellite Application

Target Architecture



Use Case: Main and Satellite Application

Target Architecture

TNS definitions

```
main = (description (load_balance=on)(failover=on)
          (address_list =
             (address = P1)
             (address = P2))
          (connect_data = (service_name=main))
        )

satellite_x = (description (load_balance=on)(failover=on)
                 (address_list =
                   (address = S1)
                   (address = S2))
                 (connect_data = (service_name=satellite_x))
               )
```

Use Case: Main and Satellite Application

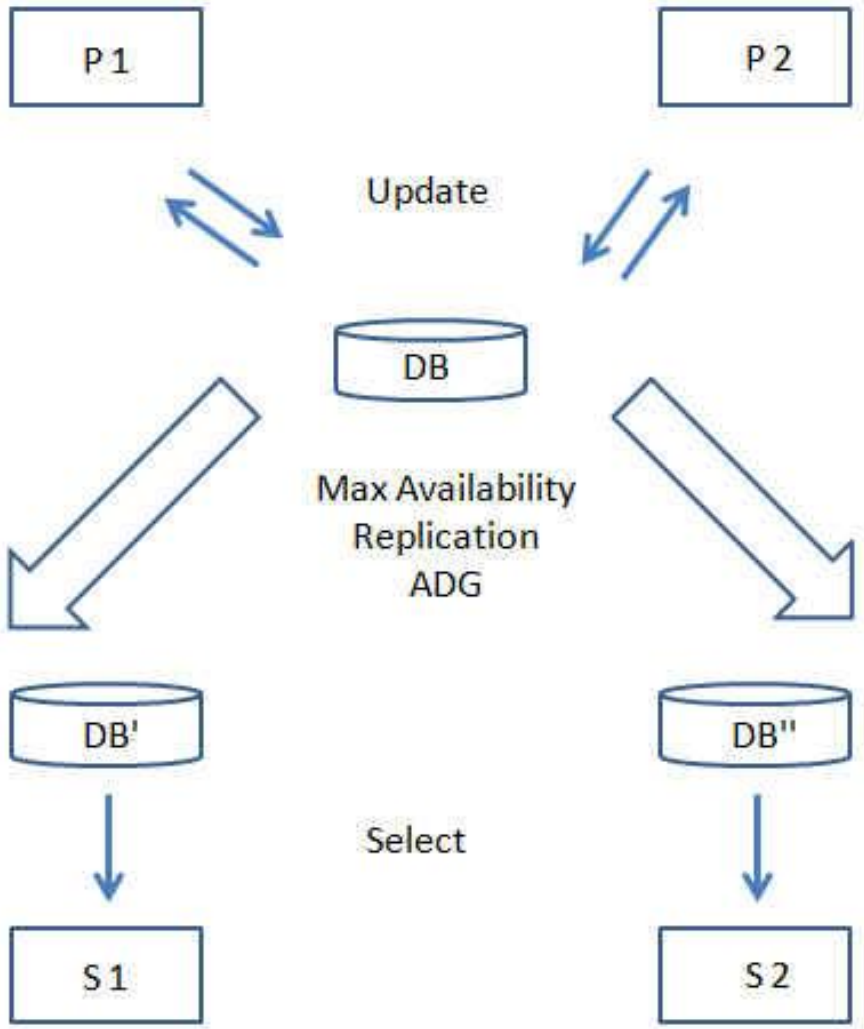
Proof of Concept

- only release 11.1.0.7 supported by i-flex® application
- time pressure in the project
- target architecture could not be implemented
- HA / DR tests did not succeed, but ended with inconsistent standby database

- alternative was looked up: Active DataGuard Reader Farm

Use Case: Main and Satellite Application

Implementation



Primary Database
2 Node RAC
service: main

Standby Databases
single instances
services: satellite_1
satellite_2
satellite_x

Use Case: Main and Satellite Application

Implementation

TNS definitions for satellites

```
satellite_x = (description (load_balance=off)(failover=on)
                (address_list =
                  (address = S1)
                  (address = S2))
                (connect_data = (service_name=satellite_x))
              )

satellite_y = (description (load_balance=off)(failover=on)
                (address_list =
                  (address = S2)
                  (address = S1))
                (connect_data = (service_name=satellite_y))
              )
```

Use Case: Main and Satellite Application

Data Consistency – High Availability

- depending on network throughput redo compression is advisable
- farm members are not guaranteed always to be in sync to each other
- satellite application connects in a given point in time only to ONE of the standby databases
- HA and load balancing is realized using the TNS layer

Protection Mode – Maximum Availability

```
log_archive_dest = LGWR SYNC AFFIRM
```

- commit waits until redo is written to SRL on standby
- at least one standby has to be available
- in case of network problems or loss of all standby's fallback to **Maximum Performance Mode**

```
log_archive_dest = LGWR ASYNC NOAFFIRM
```

Health Monitor

Heartbeat Table

- test application: inserts heartbeat records in primary database and probes for arrival at the standby side

```
CREATE TABLE dgm(
            SCN NUMBER NOT NULL,
            TIME_STAMP TIMESTAMP(3) NOT NULL,
            USER_COMMENT VARCHAR2(20 byte),
            CONSTRAINT DGM_PK PRIMARY KEY(SCN, TIME_STAMP))
            ORGANIZATION INDEX);

CREATE OR REPLACE TRIGGER tr_dgm_i
BEFORE INSERT ON dgm FOR EACH ROW
begin
 :new.scn:=dbms_flashback.get_system_change_number;
 :new.time_stamp:=systimestamp;
end;
```

Health Monitor


```
BEGIN

   -- insert marker record
   insert into dgm(user_comment)
      values (to_char(sysdate,'YYYYMMDDHH24MISS')||':MonTX')
      returning scn into v_scn;
   commit;

   -- wait a specified grace period
   dbms_lock.sleep(:timeout);
```

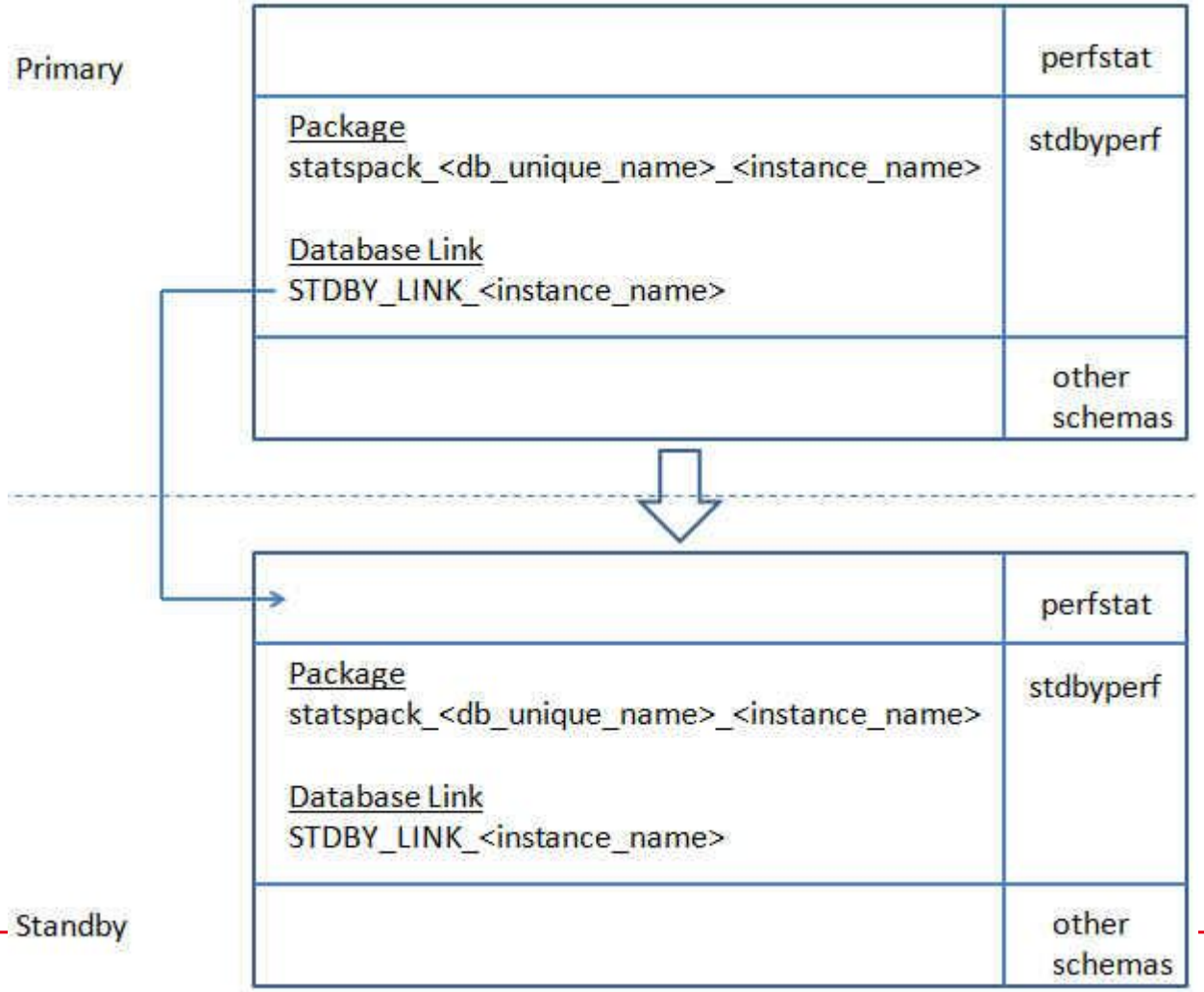
Health Monitor


```
-- find marker record on standby side
  select scn into v_scn from dgm@destination
     where scn=v_scn;
  EXCEPTION
  WHEN NO_DATA_FOUND THEN
  BEGIN
    v_ts_msg:=to_char(sysdate, v_date_format, v_nls_param);
    -- status recording in alert log
    sys.dbms_system.ksdwrt(2, v_ts_msg);
    sys.dbms_system.ksdwrt(2, v_fixed_msg || v_msg);
    raise_application_error(-20000, v_msg);
  END;
END;
/
```

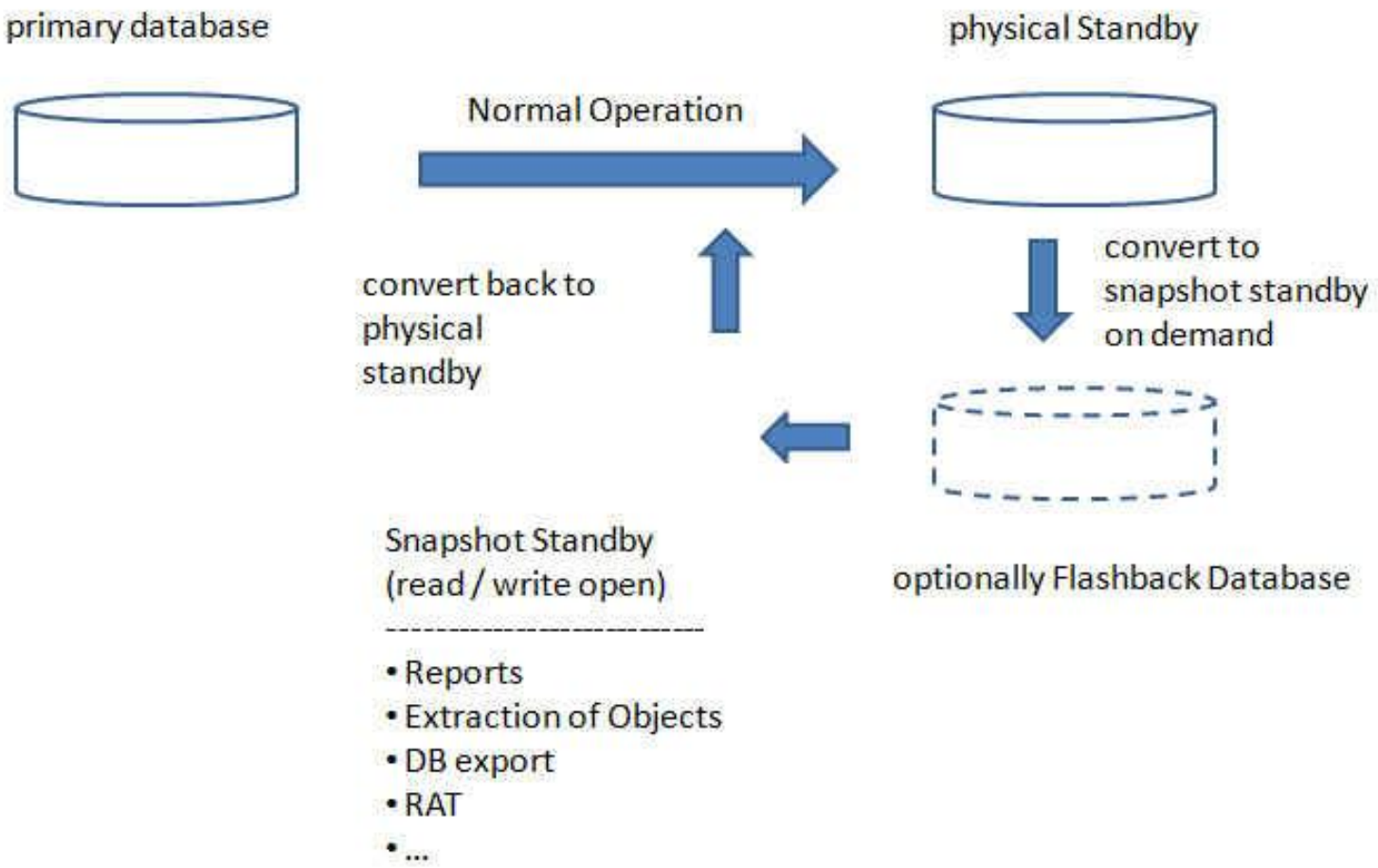
Standby Statspack

- MOS: 454848.1, AWR not possible because of R/O database

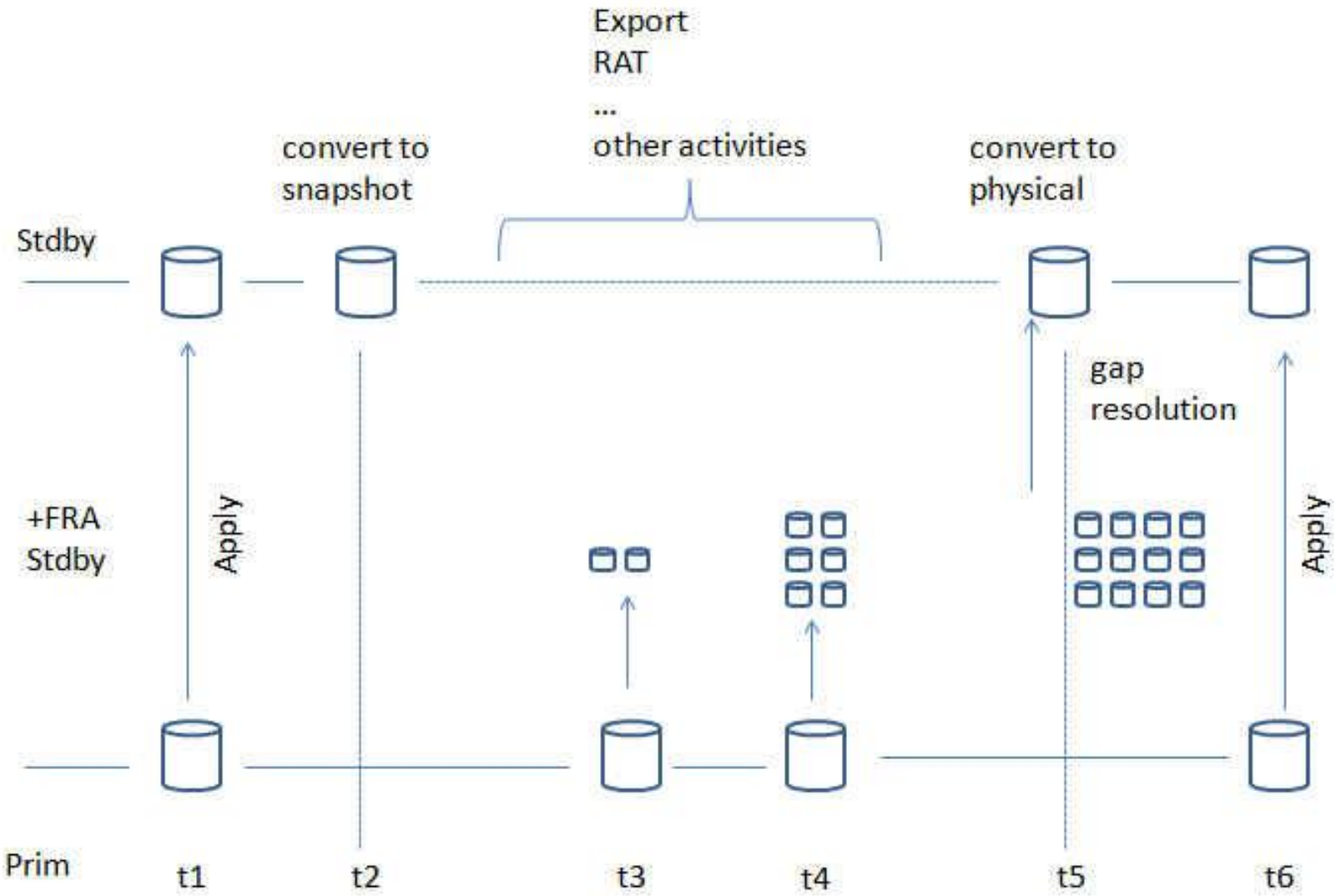
```
SQL>-- ?/rdbms/admin/sb*.sql
```



Snapshot Standby: Usage



Snapshot Standby: Usage



Conversion to Snapshot Standby

- management with sqlplus, not by broker
  lost connection because of restarts
  cluster integration

```
srvctl modify database -d <…> -r SNAPSHOT_STANDBY
sqlplus -s '/ as sysdba' <<EOF
alter database recover managed standby database cancel;
alter database convert to snapshot standby;
alter database open;
exit;
EOF
```

Conversion to Snapshot Standby


```
Mon Nov 05 03:00:11 2012
alter database recover managed standby database cancel
Mon Nov 05 03:00:11 2012
MRP0: Background Media Recovery cancelled with status 16037
Managed Standby Recovery not using Real Time Apply
Recovery interrupted!
Recovered data files to a consistent state at change
  7265279234056
Mon Nov 05 03:00:13 2012
MRP0: Background Media Recovery process shutdown
  (USTGKOR01)
Managed Standby Recovery Canceled (USTGKOR01)
Completed: alter database recover managed standby database
  cancel
```

Conversion to Snapshot Standby


```
alter database convert to snapshot standby
Starting background process RVWR
Mon Nov 05 03:00:14 2012
RVWR started with pid=30, OS id=2167
Allocated 31874880 bytes in shared pool for flashback
  generation buffer
Created guaranteed restore point
  SNAPSHOT_STANDBY_REQUIRED_11/05/2012 03:00:13

RESETLOGS after complete recovery through change
  7265279234056
Waiting for all non-current ORLs to be archived...
All non-current ORLs have been archived.
Resetting resetlogs activation ID 2987970536 (0xb218cfe8)
```

Conversion to Snapshot Standby


```
Standby became primary SCN: 7265279234054
Mon Nov 05 03:00:19 2012
Setting recovery target incarnation to 49
CONVERT TO SNAPSHOT STANDBY: Complete - Database mounted as
  snapshot standby
Completed: alter database convert to snapshot standby
alter database open
```

Conversion to Physical Standby


```
srvctl stop database -d <…>
sqlplus -s '/ as sysdba' <<EOF
startup mount
alter database convert to physical standby;
shutdown abort;
exit:
EOF

sqlplus -s '/ as sysdba'<<EOF
startup mount
alter database recover managed standby database throughall
switchover disconnect using current logfile;
shutdown immediate;
exit;
EOF
```

Conversion to Physical Standby


```
srvctl modify database -d <…> -r PHYSICAL_STANDBY
srvctl start database -d <…>
sqlplus -s '/ as sysdba' <<EOF
alter database recover managed standby database throughout
all switchover using current logfile;
exit;
EOF
```

Conversion to Physical Standby


```
Shutting down instance: further logons disabled
…
Instance shutdown complete
Mon Nov 05 23:00:36 2012
Starting ORACLE instance (normal)
…
Completed: ALTER DATABASE   MOUNT
Mon Nov 05 23:00:57 2012
alter database convert to physical standby
ALTER DATABASE CONVERT TO PHYSICAL STANDBY (USTGKOR01)
…
```

Conversion to Physical Standby


```
Flashback Restore Start
Mon Nov 05 23:01:40 2012
Flashback Restore Complete
Drop guaranteed restore point
Stopping background process RVWR
Deleted Oracle managed file
  +FRA/ustgkor0/flashback/log_1.7126.798519615
Deleted Oracle managed file
  +FRA/ustgkor0/flashback/log_2.2439.798519617
Deleted Oracle managed file
  +FRA/ustgkor0/flashback/log_3.1010.798519633
…
Completed: alter database convert to physical standby
Shutting down instance (abort)
License high water mark = 9
USER (ospid: 18116): terminating the instance
…
```

Conversion to Physical Standby

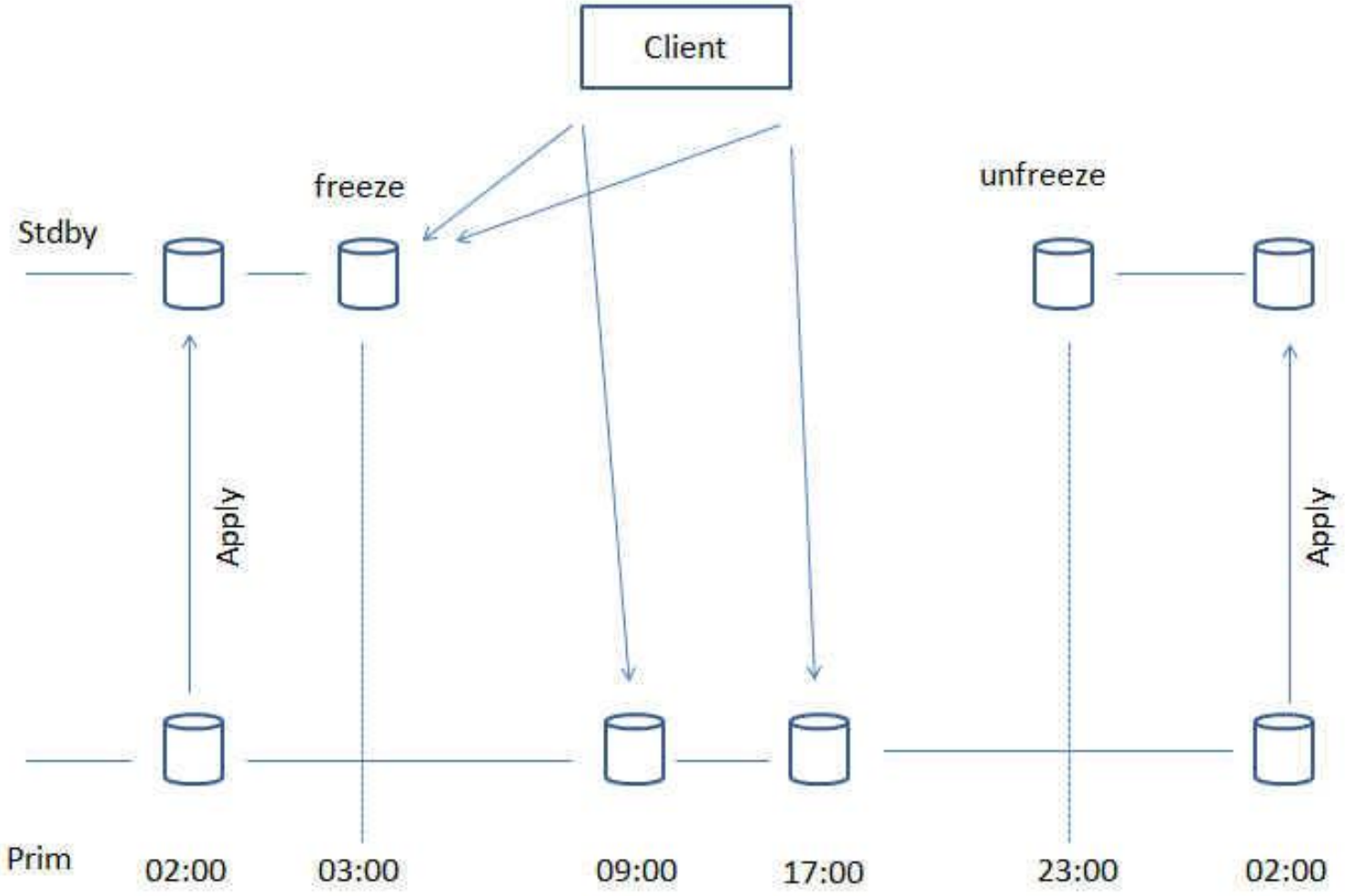

```
Completed: ALTER DATABASE   MOUNT
Mon Nov 05 23:02:05 2012
alter database recover managed standby database through all
  switchover DISCONNECT  USING CURRENT LOGFILE
Attempt to start background Managed Standby Recovery
  process (USTGKOR01)
…
```

Conversion to Physical Standby


```
Completed: alter database recover managed standby database
  through all switchover DISCONNECT  USING CURRENT LOGFILE
Media Recovery Log
  +FRA/ustgkor0/archivelog/2012_11_05/thread_4_seq_18465.389
  7.798519953
Media Recovery Log
  +FRA/ustgkor0/archivelog/2012_11_05/thread_1_seq_19802.847
  0.798519925
Media Recovery Log
  +FRA/ustgkor0/archivelog/2012_11_05/thread_2_seq_18307.862
  2.798519969
…
Media Recovery Waiting for thread 3 sequence 17152 (in
  transit)
Recovery of Online Redo Log: Thread 3 Group 11 Seq 17152
  Reading mem 0
   Mem# 0: +MISC/ustgkor0/onlinelog/group_11.350.794343587
   Mem# 1: +DATA1/ustgkor0/onlinelog/group_11.436.794343589
```

Use Case: Day – 1 Research

- Business requirement: compare Day – 1 data to live data



Control of Standby Open Mode

- measure apply lag and convert to snapshot mode if either the lag becomes zero or at the latest possible time

```
SQL> select sysdate, s.value svalue, d.database_role
     from v$dataguard_stats s, v$database d
     where s.name = 'apply lag';
```

Control of Standby Open Mode

- insert periodically records in a time table
- give applications possibility to select

```
 Name                  Null?    Type
 --------------------  -------- ----------------------------
 PRIM_SCN              NOT NULL NUMBER
 PRIM_TIMESTAMP        NOT NULL DATE
 STDBY_SCN             NOT NULL NUMBER
 STDBY_TIMESTAMP       NOT NULL DATE
```

SCN: `select current_scn from v$database;`

Timestamp:
```
select to_char(scn_to_timestamp(${scn}),
        'YYYY-MM-DD-HH24:MI:SS' ) from dual;
```

Control of Standby Open Mode

- provide a view selecting max values from primary and standby

```
Name                           Type
------------------------------ ------------------
PRIMARY_TIME                   DATE
STANDBY_TIME                   DATE

primary_time = select max(primary_timestamp)
standby_time = select max(primary_timestamp)@stdby
```