Standby in 5 Minuten

und was man sonst noch so mit Dataguard machen kann

About Me

- Benjamin Kurschies
- Jahrgang 1980
- seit 2007 Oracle DBA
- Seit 2015 Stellv. DOAG Regioleiter Hamburg

Technisch

- Spezialisiert auf HA und Performance Tuning

Inhalt

- Evolution der Standby-DB
- erstellen einer Standby-DB
- Standby != Dataguard!
- Funny things with Dataguard

Evolution der Standby-DB

Am ersten Tag schuf <VENDOR> einen Server. Der DBA hat ihn für gut befunden und Oracle installiert.

Evolution der Standby-DB

Am ersten Tag schuf <VENDOR> einen Server. Der DBA hat ihn für gut befunden und Oracle installiert.

Am zweiten Tag ist der Server ausgefallen und der Chef hat sich beschwert, dass es so lange gedauert hat, bis der Server repariert war. Also hat der DBA einen zweiten Server bestellt, auf dem er einen Restore machen kann, wenn es wieder einen Ausfall gibt.

Evolution der Standby-DB

Am ersten Tag schuf <VENDOR> einen Server. Der DBA hat ihn für gut befunden und Oracle installiert.

Am zweiten Tag ist der Server ausgefallen und der Chef hat sich beschwert, dass es so lange gedauert hat, bis der Server wieder lief. Also hat der DBA einen zweiten Server bestellt, auf dem er einen Restore machen kann, wenn es wieder einen Ausfall gibt.

Am dritten Tag ist der Server wieder ausgefallen. Da alle Mitarbeiter der Firma um den DBA herum standen und ständig gefragt haben wann der Restore fertig ist, hat diese versehentlich den falschen Knopf gedrückt und das Backup gelöscht. Ab Morgen hat die Firma einen neuen DBA.

Evolution der Standby-DB

Am ersten Tag schuf <VENDOR> einen Server. Der DBA hat ihn für gut befunden und Oracle installiert.

Am zweiten Tag ist der Server ausgefallen und der Chef hat sich beschwert, dass es so lange gedauert hat, bis der Server wieder lief. Also hat der DBA einen zweiten Server bestellt, auf dem er einen Restore machen kann, wenn es wieder einen Ausfall gibt.

Am dritten Tag ist der Server wieder ausgefallen. Da alle Mitarbeiter der Firma um den DBA herum standen und ständig gefragt haben wann der Restore fertig ist, hat diese versehentlich den falschen Knopf gedrückt und das Backup gelöscht. Ab Morgen hat die Firma einen neuen DBA.

Am vierten Tag ist der Server wieder ausgefallen. der neue DBA hatte aber vorher den Ersatz-Server so weit vorbereitet, dass nur noch ein Skript gestartet werden muss, dass die Datenbank wieder herstellt. Die Datenbank ist leider inzwischen 10 TB Groß, der Restore hat sehr lange gedauert.

Evolution der Standby-DB

Am ersten Tag schuf <VENDOR> einen Server. Der DBA hat ihn für gut befunden und Oracle installiert.

Am zweiten Tag ist der Server ausgefallen und der Chef hat sich beschwert, dass es so lange gedauert hat, bis der Server wieder lief. Also hat der DBA einen zweiten Server bestellt, auf dem er einen Restore machen kann, wenn es wieder einen Ausfall gibt.

Am dritten Tag ist der Server wieder ausgefallen. Da alle Mitarbeiter der Firma um den DBA herum standen und ständig gefragt haben wann der Restore fertig ist, hat diese versehentlich den falschen Knopf gedrückt und das Backup gelöscht. Ab Morgen hat die Firma einen neuen DBA.

Am vierten Tag ist der Server wieder ausgefallen. der neue DBA hatte aber vorher den Ersatz-Server so weit vorbereitet, dass nur noch ein Skript gestartet werden muss dass die Datenbank wieder herstellt. Die Datenbank ist leider inzwischen 10 TB Groß, der Restore hat sehr lange gedauert.

Am fünften Tag ist der Server wieder ausgefallen. Der DBA hatte vorher aber auf dem Ersatz-Server schon ein Restore gemacht und spielt per Skript jede Stunde die inzwischen angefallenen Archivelogs ein. Der DBA musste nur noch die Archivelogs seit dem letzten Start des Skriptes einspielen und war innerhalb von 10 Minuten wieder online. Der DBA nannte seine Erfindung eine „Standby Datenbank“

Evolution der Standby-DB

Am ersten Tag schuf <VENDOR> einen Server. Der DBA hat ihn für gut befunden und Oracle installiert.

Am zweiten Tag ist der Server ausgefallen und der Chef hat sich beschwert, dass es so lange gedauert hat, bis der Server wieder lief. Also hat der DBA einen zweiten Server bestellt, auf dem er einen Restore machen kann, wenn es wieder einen Ausfall gibt.

Am dritten Tag ist der Server wieder ausgefallen. Da alle Mitarbeiter der Firma um den DBA herum standen und ständig gefragt haben wann der Restore fertig ist, hat diese versehentlich den falschen Knopf gedrückt und das Backup gelöscht. Ab Morgen hat die Firma einen neuen DBA.

Am vierten Tag ist der Server wieder ausgefallen. der neue DBA hatte aber vorher den Ersatz-Server so weit vorbereitet, dass nur noch ein Skript gestartet werden muss dass die Datenbank wieder herstellt. Die Datenbank ist leider inzwischen 10 TB Groß, der Restore hat sehr lange gedauert.

Am fünften Tag ist der Server wieder ausgefallen. Der DBA hatte vorher aber auf dem Ersatz-Server schon ein Restore gemacht und spielt per Skript jede Stunde die inzwischen angefallenen Archivelogs ein. Der DBA musste nur noch die Archivelogs seit dem letzten Start des Skriptes einspielen und war innerhalb von 10 Minuten wieder online. Der DBA nannte seine Erfindung eine „Standby Datenbank“

Am sechsten Tag sah Oracle die Standby Lösung und befand sie für gut – lediglich dass sie daran nichts verdienen misfiel. Darauf hin erfand Oracle das Enterprise feature "Dataguard", mit der nicht mehr die Archivelogs auf der Standby restored werden sondern die Redo-Logs während der entstehung gestreamt werden. Jetzt muss der DBA keine Archivelogs mehr einspielen und im Fehlerfall sind (fast)* keine Daten weg.

Erstellung einer Standby-DB

1. Voraussetzungen

1.1 für eine Standby DB

- Datenbank muss im Archivelog mode sein
- Datenbank muss im “force logging” mode sein

1.2 für eine Dataguard DB

- Oracle EE
- Standby redo logs *
- empfehlung: flashback einschalten

Erstellung einer Standby-DB

2. Durch Kopieren

1.1 Offline

```
sqlplus> shutdown immediate;
scp /u01/oradata/orcl/* standby:u01/oradata/orcl/
```

1.2 Online

```
sqlplus> alter database begin backup;
scp /u01/oradata/orcl/* standby:u01/oradata/orcl/
sqlplus> alter database end backup;
```

Erstellung einer Standby-DB

3. rman duplicate

voraussetzungen

- Listener eintrag wurde statis hinzugefügt
- spfile(!) und password file wurden auf standby kopiert
- zukünftige Standby ist im NoMount

```
rman target sys@std01:1521/orcl auxiliary sys@std02:1521/orcl
rman> duplicate target database for standby from active database
```

Vorteil:

- funktioniert auch mit ASM
- kann einfach parallelisiert werden (rman config)

Standby != Dataguard

- Oracle Dataguard erweitert die möglichkeiten einer “normalen” Standby-DB

- Vorteile:
  - (near) zero data loss: es wird direkt aus dem log buffer gestreamt, im fehlerfall sind (fast) keine Daten weg *
  - archivelog preserve: Wenn die Standby noch archivelogs zum “aufholen” benötigt, werden diese von der Primary nicht (automatisch) gelöscht.
  - sync after network outage: Sobald sich die Datenbanken wieder “sehen”, werden sofort wieder daten auszutauschen; innerhalb kürzester Zeit ist die Datenbank wieder syncron.
  - comfortables CLI um die Standby einzurichten

Standby != Dataguard

- Oracle Dataguard erweitert die möglichkeiten einer “normalen” Standby-DB

- die neuen Möglichkeiten haben (natürlich) auch neue abhängigkeiten:
  - unterschiedliche DB_UNIQUE_NAME

    wir müssen die Datenbanken ja irgendwie unterscheiden können…
  - standby-redo-logs

    Die Primary kann nicht direkt in die Standby-DB rein “streamen”, die empfangenen Daten müssen in Standby Redo Logs zwischengespeichert werden.

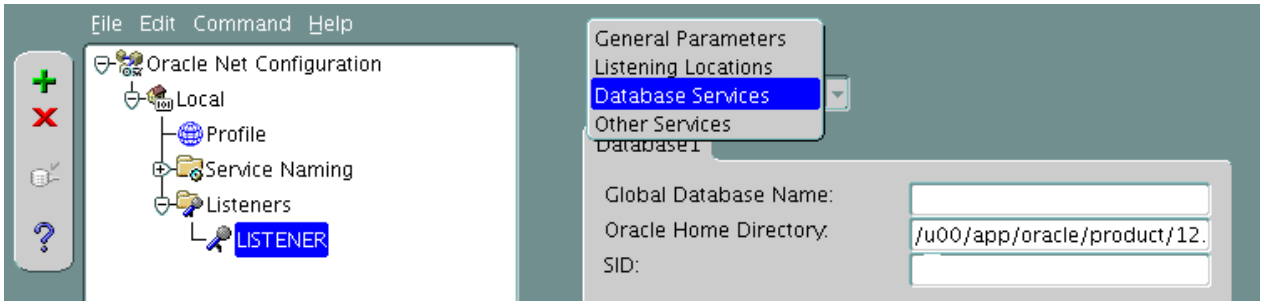
    Es werden n+1 standby logs benötigt, wobei n die anzahl der redo logs ist
  - Es muss ein spfile genutzt werden
  - recommendation: OFA und “db_create_file_dest” / “log_create_file_dest” nutzen
  - recommendation: standby_file_management=AUTO setzen

Standby != Dataguard

Upgrade to Dataguard

1. Add static listener Entry

```
netmgr -> add "DB_UNIQUE_NAME"_DGMGRL as Global Database Name
```



2. set db parameter

```
alter database force logging;
alter database flashback on;
alter database add standby log group 11 size 50m;
alter system set dg_broker_start=true;
alter system set standby_file_management=AUTO;
```

Standby != Dataguard

Upgrade to Dataguard

3. copy spfile & orapw file

```
scp $ORACLE_HOME/dbs/spfile${ORACLE_SID} std02:/…
scp $ORACLE_HOME/dbs/orapw${ORACLE_SID} std02:/…
```

4. duplicate

```
sql> startup nomount;

rman target sys@std01:1521/orcla \
auxiliary sys@std02:1521/orclb_dgmgrl

duplicate target database for standby \
from active database spfile \
set “orclb”
```

Standby != Dataguard

Upgrade to Dataguard

- Upgrade to Dataguard

5. create dataguard config

```
dgmgrl sys@srv01:1521/orcl
create configuration ‘DG_ORCL’ \
as primary database is ‘ORCLA’ \
connect identifier is “std01:1521/orcla_dgmgrl”;

add database ‘STANDBY’ \
as connect identifier is “std02:1521/orclb_dgmgrl”;

enable configuration

FINISH !!!
```

Standby != Dataguard

Standby+ Solutions

Es gibt tools, die eine fertige Standby-Lösung (nicht DG!) bieten, z.B.

DB-Visit: db-visit standby

Trivadis: tvd-standby

Funny things with DG

Switchover / Failover

Ein Failover ist DER Grund, warum wir überhaupt eine Standby haben… im Fehlerfall wird diese einfach aktiviert

1. ohne DG: `sql> alter database open resetlogs;`
2. mit DG: `dgmgrl> failover to 'ORCLB';`

Des weiteren haben wir mit einer Standby die Möglichkeit, für geplante Wartungsarbeiten die Rollen zu tauschen.

1. ohne DG:
    1. herunterfahren beider Datenbanken
    2. kopieren aller archivelogs und online-redo-logs auf die Standby
    3. standby recovern
    4. standby-db starten
2. mit DG: `dgmgrl> switchover to 'ORCLB';`

Funny things with DG

Table pit recover

1. vorbereitungen
   - recover auf standby stoppen

     ```
     dgmgrl> edit database .. set state='APPLY-OFF';
     ```

   - Flashback / restore der Standby zum gewünschten Zeitpunkt;

     ```
     rman target /
     run {
     set until time to_date('2016-09-27 12:00', 'YYYY-MM-DD HH24:MI');
     --allocate channel c1 TYPE 'sbt_tape'
     --flashback database;
     restore database;
     recover database;
     }
     ```

Funny things with DG

Table pit recover

1.1 über einen DB-Link

- Standby “read onle” öffnen *

```
sql> alter database open read only;
```

- DB-Link in der Primary auf die Standby anlegen

```
sql> create database link ‘standby’ as …
```

- fehlerhafte Tabelle umbenennen (oder löschen)

```
sql> rename table mytable to old_table
```

- Tabelle von Standby “kopieren”

```
sql> create table mytable as select * from mytable@standby
```

- indizes, constraints, trigger etc. erstellen

```
sql> create index …
```

Funny things with DG

Table pit recover

1.2 über “export / Import”

- Standby “read only” öffnen
- Einfach mit “exp” die Tabelle exportieren

1.3 über datapump

1.3a: standby in eine “snapshot standby” umwandeln

```
dgmgrl> convert database ‘orclb’ to snapshot standby
```

1.3b: manuelle “snapshot standby”

```
rman> create guarantee restore point x
rman> alter database open resetlogs;
```

- datapump prozess starten

Funny things with DG

Table pit recover

1.4 Nacharbeiten / Standby wieder syncronisieren

- Bei 1.1 und 1.2

```
sql> shutdown immediate;
sql> startup mount;
rman> recover database; -- optional
dgmgrl> edit database 'orclb' set state='APPLY-ON'
```

- Bei 1.3a

```
dgmgrl> convert database 'orclb' to physical database;
```

- Bei 1.3b

```
rman> flashback to restore point x;
```

Funny things with DG

incremental refresh

Sollte die Standby (z.B. wegen Fehler im Netzwerk) der Primär-DB zu weit hinterher sein, macht es evtl. sinn einen incrementellen refresh zu machen.

1.1. ermitteln der letzten Transaktion auf der Standby

```
sql> select min(current_scn) from (
select current_scn from v$database
union all
select last_change# from v$datafile);
```

Funny things with DG

incremental refresh

1.2 Backup auf Primary starten

```
rman> backup as compressed backupset \
from scn 1234 database \
include current controlfile for standby \
device type disk format '/orabackup/for_standby_%U;

bash> scp /orabackup/for_standby* srv02:/…
```

Funny things with DG

incremental refresh

1.3 recover auf standby starten

```
dgmgrl> edit database ‘standby’ set state=‘APPLY-OFF’
rman> shutdown immediate;
rman> startup nomount;
rman> restore controlfile from ‘/orabackup/…’
rman> alter database mount;
-- bei OFA / ASM
rman> catalog start with ‘+DATA/STANDBY/DATAFILE/’
rman> switch database to copy;
-- normal weiter
rman> catalog start with ‘/orabackup/’;
rman> recover database noredo;
dgmgrl> edit database ‘standby’ set state=‘APPLY-ON’
```

Funny things with DG

Auto Block-repair

1.1 Active Dataguard Lizenz kaufen

1.2 Active Dataguard aktivieren

```
sql> alter database open;
```

1.3 Warten bis etwas kaputt geht…

Funny things with DG

updates – Standby first

- reduziert Downtime
- nicht geignet für Major updates

  (11.2.0.3 -> 11.2.0.4 OK / 11.2.0.4 -> 12.1.0.2 NO)
- Im jeweiligen Patch ist explizit “standby first” freigegeben

1. update Standby (erst CRS/ASM, dann DB_HOME!)
2. switchover
3. upgrade former primary
4. sql patch installieren (datapatch / catbundle.sql psu apply)

Funny things with DG

updates – rolling upgrade

- near zero Downtime (< 60 sek)
- prinzipiell auch für major upgrade geeignet
  - funktioniert ab Oracle 12.1

1. prüfen ob nicht supportete datentypen vorhanden sind

```
SELECT COLUMN_NAME,DATA_TYPE FROM DBA_LOGSTDBY_UNSUPPORTED;
```

2. Plan initieren

```
EXECUTE DBMS_ROLLING.INIT_PLAN (future_primary => 'ORCLB');
```

3. Plan erstellen

```
EXECUTE DBMS_ROLLING.BUILD_PLAN;
```

Funny things with DG

updates – rolling upgrade

4. DG configuration deaktivieren

```
dgmgrl> disable configuration;
```

5. Prüfen:

```
sql> SELECT instid,source,target,phase,status FROM dba_rolling_plan;
sql> SELECT instid, target, phase, description FROM dba_rolling_plan WHERE instid IN (7);
```

6. Plan aktivieren (Achtung! jetzt passiert wirklich etwas!)

```
sql> EXECUTE DBMS_ROLLING.START_PLAN;
```

7. Standby DB aktualisieren (erst ASM, dann DB_HOME)
8. Switchover

```
sql> EXECUTE DBMS_ROLLING.SWITCHOVER;
```

9. ehemalige Primary patchen (binary only)
10. Datenbanken wieder syncronisieren

```
sql> EXECUTE DBMS_ROLLING.FINISH_PLAN;
```

11. DG configuration wieder aktivieren

Q & A

Benjamin Kurschies
benjamin@kurschies.de
www.xing.com/profile/Benjamin_Kurschies