

5.8.3 Switchover mit Data Guard CLI

Mit Data Guard CLI (dgmgrl) kann der Switchover wesentlich einfacher realisiert werden als über die zuvor beschriebenen manuellen Verfahren. Der Switchover erfolgt hier in vier Schritten:

Schritt 1: Prüfen des Status der Primärdatenbank

Mit dem Befehl `show database verbose` kann im Data Guard CLI geprüft werden, in welchem Zustand die Primärdatenbank ist.

```
DGMGR> show database verbose frankfurt
```

Wichtig ist, dass hier als Status der Primärdatenbank *success* zurückgegeben wird.

Schritt 2: Prüfen des Status der Standby-Datenbank

Analog zu Schritt 1 sollte der Status der Standby-Datenbank geprüft werden.

```
DGMGR> show database verbose tokiu
```

Auch für die Standby-Datenbank sollte als Status *success* zurückgemeldet werden.

Schritt 3: Switchover

Mit dem Befehl `Switchover` kann nun der Rollentausch zwischen Primär- und Standby-Datenbank initiiert werden:

```
DGMGR> SWITCHOVER TO "tokio";
```

```
Performing switchover NOW. Please wait...
```

```
Operation requires shutdown of instance "frankfurt" on database"frankfurt".
```

```
Shutting down instance "frankfurt"...
```

```
Database dismounted.
```

```
ORACLE instance shut down.
```

```
Operation requires shutdown of instance "tokio" on database "tokio".
```

```
Shutting down instance "tokio"...
```

```
database not open
```

```
ORACLE instance shut down.
```

```
Operation requires startup of instance "frankfurt" on database "frankfurt".
```

```
Starting instance "frankfurt"...
```

```
ORACLE instance started.
```

```
Database mounted.
```

```
Operation requires startup of instance "tokio" on database "tokio".
```

```
Starting instance "tokio"...
```

```
ORACLE instance started.
```

```
Database mounted.
```

```
Switchover succeeded. New primary is "tokio"
```

Listing 5.155: Switchover zur Datenbank tokiu

Schritt 4: Prüfen des Switchover

Abschließend sollte man mittels `show configuration` den Switchover überprüfen:

```
DGMGR> show configuration;

Configuration
Name:DG_frankfurt
Enabled:YES
Protection Mode: MaxPerformance
Databases:frankfurt - Physical standby database
tokio - Primary database
Current status for "DG_frankfurt": SUCCESS
```

Listing 5.156: Anzeige der Data Guard-Konfiguration

Man sieht: Der anfängliche Aufwand für die Basiskonfiguration der Data Guard Broker-Architektur lohnt sich. Letztlich besteht der Switchover mit dem Data Guard CLI nur aus einem Befehl, nämlich `switchover to <standby_db>`. Manuelle Eingriffe sind nicht nötig, die Überprüfung des Status der Datenbanken ist einfach, und Fehler eines Administrators durch manuelle Operationen werden weitestgehend ausgeschlossen. Noch einfacher ist nur der Switchover mit dem grafischen Grid Control.

5.8.4 Switchover mit Grid Control

Mit Grid Control ist der Switchover denkbar einfach. Nach Auswahl der betreffenden Standby-Datenbank wird die Option `Switchover` genutzt. Die verbleibenden Redo Logs werden dann automatisch transferiert und verarbeitet. Abschließend wird die Primärdatenbank heruntergefahren und die Standby-Datenbank aktiviert.

5.9 Failover

Während bei einem Switchover die Primär- und Standby-Datenbank die Rollen tauschen, wird bei einem Failover davon ausgegangen, dass das Primärsystem nicht verfügbar ist. Daher kann die Primärdatenbank nicht in die Rolle der Standby wechseln. Eine Folge ist, dass Redo Log-Informationen nicht übertragen werden können. Die ehemalige Primärdatenbank ist also nicht mehr auf dem aktuellen Stand. Daher ist nach einem Failover – anders als nach einem Switchover – die ehemalige Primärdatenbank verloren. Diese muss nach dem Failover reinstantiiert, also komplett neu aufgebaut werden.

Trotzdem sollte der Failover nicht erst dann durchexerziert werden, wenn der Ernstfall eingetreten ist. Er sollte zuvor schon einmal geprobt worden sein. Die wichtigsten Schritte müssen dem DBA bekannt sein. Ein falscher Befehl kann im Ernstfall einen unnötigen Datenverlust verursachen.

Zunächst einmal sollte versucht werden, alle Redo-Informationen auf die Standby-Seite zu übertragen. Dies schließt eventuell auf der Primärseite vorhandene Archived Redo Logs, die nicht oder nicht vollständig auf die Standby-Seite übertragen wurden, ein.

Handelt es sich bei der Standby-Datenbank um eine Real Application Cluster-Konfiguration, sollte dafür gesorgt werden, dass alle Instanzen bis auf eine heruntergefahren werden. Sofern ein Delay für das Log Apply implementiert wurde, sollte dieses entfernt werden. Handelt es sich um eine Konfiguration mit mehreren Standby-Datenbanken, bei der sowohl eine Physical als auch eine Logical Standby-Datenbank zur Verfügung steht, so ist die Physical Standby-Datenbank zur Übernahme der Rolle als Primärdatenbank zu bevorzugen.

Auch der Failover lässt sich manuell, mit dem Data Guard CLI oder mit dem grafischen Grid Control initiieren. Alle drei Verfahren werden nachfolgend beschrieben.

5.9.1 Failover einer Physical Standby-Datenbank

Der Failover einer Physical Standby-Datenbank vollzieht sich in sechs Schritten:

Schritt 1: Beheben von Redo-Lücken

Über die View `v$archive_gap` können eventuell vorhandene Lücken in den Redo Informationen identifiziert werden. Alle Archived Redo Logfiles der Primärdatenbank sollten – soweit möglich – auf das Standby-System kopiert werden, sofern deren Log Sequence-Nummer größer ist als jene, die zuletzt auf der Standby-Seite empfangen wurde.

Schritt 2: Registrieren der in Schritt 1 kopierten Archived Redo Logfiles

Die in Schritt 1 manuell übertragenen Files müssen nun noch registriert werden. Dies geschieht mit dem Befehl

```
alter database
register physical logfile '/arch1/tokio/arch_952.arc';
```

Schritt 3: Nachfahren der letzten Transaktionen

Werden Standby Redo Logs verwendet, so werden mit dem Befehl

```
alter database recover managed standby database finish;
```

die letzten noch offenen Transaktionen abgeschlossen. Werden keine Standby Redo Logs verwendet oder sind diese nicht aktiv, so wird der Befehl leicht abgewandelt eingesetzt:

```
alter database recover managed standby database finish
skip standby logfile;
```

Schritt 4: Übernahme der Primary Rolle

Nun kann die Standby-Datenbank die Rolle der Primärdatenbank übernehmen:

```
alter database commit to switchover to primary;
```

Schritt 5: Shutdown und Restart der neuen Primärdatenbank

Die ehemalige Standby-Datenbank muss nun einmal restartet werden, um die Rolle der Primärdatenbank zu übernehmen:

```
shutdown immediate;
startup;
```

Schritt 6: Backup der neuen Primärdatenbank

Die neue Primärdatenbank sollte sobald wie möglich ins Backup genommen werden.

Damit ist die Übernahme im Fall eines Failovers abgeschlossen. Man sieht: Es sind u.U. einige manuelle Arbeiten nötig. Mit dem grafischen Grid Control geht es etwas einfacher. Dennoch sind auch hier gelegentlich Archived Redo Log-Informationen manuell zu übertragen.

5.9.2 Failover einer Logical Standby-Datenbank

Bei einem Failover auf eine Logical Standby-Datenbank wird die ehemalige Primärdatenbank sowie jede Physical Standby-Datenbank, die an dieser Primärdatenbank hängen, unbrauchbar. Diese sind nach einem Failover auf die Logical Standby nicht mehr Bestandteil der Data Guard-Konfiguration. In den meisten Fällen können weitere, an derselben Primärdatenbank hängenden Logical Standby-Datenbanken jedoch weiter betrieben werden.

Der Failover selbst vollzieht sich in fünf Schritten, die nachfolgend beschrieben werden.

Schritt 1: Beheben von Redo-Lücken

Auch in einer Logical Standby-Konfiguration muss im Fall eines Failovers geprüft werden, ob eventuell Redo-Informationen manuell übertragen werden müssen, da diese nicht mehr von der Primärdatenbank transferiert werden konnten. Hierzu wird die View `dba_logstdby_log` abgefragt. Fehlende Archived Redo Logs, deren Log Sequence-Nummer höher als die höchste der in `dba_logstdby_log` angegebenen Log-Nummer ist, müssen manuell übertragen werden.

Schritt 2: Registrieren der in Schritt 1 kopierten Archived Redo Logfiles

Archivierte Redo Logfiles, die manuell übertragen wurden, müssen nun noch auf Seite der Standby-Datenbank registriert werden.

```
alter database
register logical logfile '/arch1/tokio/arch_952.arc';
```

Durch die Registrierung der manuell übertragenen Archived Redo Logfiles erkennt die Standby-Datenbank, dass deren Informationen noch zu verarbeiten sind. Das Nachfahren der darin enthaltenen Transaktionen wird nun automatisch gestartet.

Schritt 3: Prüfen des Redo Apply

Über die View `dba_logstdby_progress` kann nun geprüft werden, ob alle Transaktionen der manuell übertragenen Archived Redo Logs abgeschlossen wurden. Die Spalten `[applied_scn]` und `[newest_scn]` sind dann identisch:

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

Schritt 4: Aktivieren der Logical Standby-Datenbank

Abschließend wird die Logical Standby-Datenbank in der neuen Rolle als Primärdatenbank aktiviert:

```
alter database stop logical apply;
alter database activate logical standby database;
```

Schritt 5: Backup der neuen Primärdatenbank

Die neue Primärdatenbank sollte sobald wie möglich ins Backup genommen werden.

5.9.3 Failover mit Data Guard CLI

Der Failover mit Data Guard CLI erfolgt in nur zwei Schritten.

Schritt 1: Verbindung zur Standby-Datenbank und Absetzen des Failover-Befehls

Um den Failover einzuleiten, muss eine Verbindung zur Standby-Datenbank erstellt und der Failover-Befehl abgesetzt werden:

```
DGMGRL> CONNECT sys@tokio
Connected.

DGMGRL> FAILOVER TO "tokio";
Performing failover NOW. Please wait...
Operation requires shutdown of instance "tokio" on database
"tokio".
Shutting down instance "tokio"...
database not mounted
ORACLE instance shut down.

Operation requires startup of instance "tokio" on database "tokio".
Starting instance "tokio"...
ORACLE instance started.
Database mounted.

Failover succeeded. New primary is "tokio"
```

Listing 5.157: Failover zur Datenbank tokio mit dgmgrl

Dieser Failover kann – je nachdem welche Menge an Redo Log-Informationen noch zu verarbeiten ist und wie lange der Startup der Standby-Datenbank benötigt – zwischen 30 Sekunden und einigen Minuten in Anspruch nehmen. Danach steht die ehemalige Standby-Datenbank als Primärdatenbank zur Verfügung.

Schritt 2: Überprüfen der Stati nach dem Failover

Abschließend sollte man den Status der Standby-Konfiguration und der beteiligten Datenbanken überprüfen:

```
DGMGRL> SHOW CONFIGURATION;

Configuration
Name:                DG_frankfurt
```

```

Enabled:          YES
Protection Mode: MaxPerformance
Databases:
  frankfurt - Physical standby database
  tokió      - Primary database
Current status for "DG_frankfurt":
SUCCESS

```

Listing 5.158: Anzeige des aktuellen Status mit dgmgrl

Die ehemalige Standby-Datenbank steht nun voll für Benutzerzugriffe zur Verfügung.

Nach einem Failover ist die ehemalige Primärdatenbank allerdings verloren. Diese muss reinstantiiert, also komplett neu aufgebaut werden. Zu sehen ist dies auch, wenn man den Status der ehemaligen Primärdatenbank im Data Guard CLI überprüft:

```
DGMGRL> SHOW DATABASE 'frankfurt';
```

```

Database
Name:          frankfurt
Role:          PHYSICAL STANDBY
Enabled:       NO
Intended State: ONLINE
Instance(s):   frankfurt

```

```
Current status for "frankfurt":
```

```
Error: ORA-16795: Database resource guard detects that database reinstantiation is required
```

Listing 5.159: Prüfen der Datenbank frankfurt

Für die Reinstantiierung wird im Prinzip eine neue Standby-Datenbank auf dem ehemaligen Primärsystem aufgebaut. Im genannten Beispiel müsste also aus einem Backup der Datenbank *tokio* eine neue Standby-Datenbank auf dem Host *frankfurt* erstellt werden. Diese wird anschließend wieder in die Data Guard-Konfiguration aufgenommen. So man möchte, kann über einen Switchover die Datenbankinstanz *frankfurt* wieder zur Primärdatenbank gemacht werden.

5.10 Grid Control und Data Guard

Viele der Verwaltungsoperationen können auch mit Grid Control (siehe Abschnitt 13.6) erledigt werden. Der Vorteil ist vor allem in der intuitiven Benutzerführung zu sehen. Doch auch der Failover und Switchover ist mit Grid Control recht einfach. Zudem steht ein Cloning Wizard zur Verfügung, der die Erstellung eines Datenbank-Klones enorm vereinfacht (siehe auch Abschnitt 13.6).

5.11 Transparent Application Failover der Clients

Um einen automatisierten Client Failover in einer Data Guard-Konfiguration zu erreichen, kann *Transparent Application Failover (TAF)* eingesetzt werden (siehe Abschnitt 6.10.2). Sowohl Connect Time Failover als auch Application Failover sind hier nutzbar. Allerdings gibt es keine Übernahme von Sessioninformationen, wie dies bei Einsatz des Oracle Real Application Clusters der Fall ist. Dennoch ist eine Automatisierung des Failovers der Clients via TAF möglich. Unter Clients ist hier jeder »Kunde« der Primärdatenbank zu verstehen. Dabei kann es sich um eine klassische Client-Anwendung, aber durchaus auch um einen Application Server handeln. Die Konfiguration ist in beiden Fällen im Wesentlichen identisch. Einige Application Server werden aber möglicherweise zusätzliche Funktionen anbieten, die über die Möglichkeiten von Oracles TAF hinausgehen.

Auch bei Data Guard kann grundsätzlich sowohl Connect Time Failover bei der Verbindungsaufnahme als auch Application Failover für bestehende Verbindungen eingesetzt werden. Fraglich ist dennoch, wie weit dies sinnvoll ist. Oft ist nicht bekannt, wie viel Zeit ein Failover auf die Standby-Seite benötigen wird. U.U. kann es daher sinnvoller sein, je einen gesonderten Datenbankalias für die Primär- und die Standby-Datenbank zu konfigurieren. Bei Bedarf muss dann ein Application Server auf den jeweils anderen Datenbankalias umkonfiguriert werden. Benutzer, die sich mit ihren Anwendungen direkt auf die Datenbank verbinden, müssten dann bei einem Switchover bzw. Failover kurz informiert werden. Doch diese Information wird in den meisten Anwendungsumgebungen ohnehin nötig sein.

Auch wenn der Einsatz von TAF in Kombination mit Data Guard unter den derzeitigen Möglichkeiten nicht immer sinnvoll ist, wird die Konfiguration allein schon der Vollständigkeit halber hier trotzdem erörtert.

Connect Time Failover

Oracle Net nutzt bei der Verbindungsaufnahme zum Datenbankserver einen einfachen TCP-Timeout-Mechanismus. Dieser Mechanismus gestattet, nach einer Expire Time auf eine zweite Verbindung auszuweichen.

Connect Time Failover zieht in folgenden Fällen:

- ▶ Die Primärdatenbank steht nicht zur Verfügung.
- ▶ Der Oracle Listener der Primärdatenbank steht nicht zur Verfügung.
- ▶ Die Netzwerkverbindung zur Primärdatenbank funktioniert nicht.

Um Connect Time Failover zu implementieren, wird für die Adressliste des Datenbankalias in der `tnsnames.ora` ein zweiter Host konfiguriert:

```
DGD =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS=(PROTOCOL=TCP)(HOST=frankfurt)(PORT=521))
      (ADDRESS=(PROTOCOL=TCP)(HOST=tokio)(PORT=1521))
    )
  )
```

```

(CONNECT_DATA =
  (SERVICE_NAME = DGD)
)
)

```

Listing 5.160: Konfiguration des Connect Time Failover

Der Service Name beider Datenbanken muss hierfür identisch sein. Verbindet man sich wie im Beispiel mit dem Alias DGD, so wird automatisch der erste Adresseintrag für eine Verbindungsanforderung verwendet. Läuft Oracle Net hier in einen Fehler, wird anschließend der Host des zweiten Adresseintrages kontaktiert.

In obiger Konfiguration wird also zunächst der Host frankfurt kontaktiert. Ist dieser nicht erreichbar, wird versucht, eine Verbindung zu Host tokiyo herzustellen. Schlägt dies fehl, gibt Oracle Net eine von der Ursache abhängige Fehlermeldung zurück.

Allerdings ist es fraglich, ob eine solche Konfiguration von Connect Time Failover wirklich Sinn macht. Haben Primär- und Standby-Datenbank ihre Rollen aufgrund eines Switchover getauscht, so ist die ehemalige Primärdatenbank auch weiterhin erreichbar. Allerdings befindet sie sich im Managed Recover Modus und steht – im Fall einer Physical Standby Database – nicht für Benutzerverbindungen zur Verfügung. Die Fehlermeldung

```
ORA-01033: ORACLE initialization or shutdown in progress
```

ist die Folge.

Application Failover

Anders als bei Verwendung des Real Application Clusters steht bei einem Failover oder einem Switchover einer Standby-Datenbank diese nicht unmittelbar als Primärdatenbank zur Verfügung. Im Regelfall sind einige Minuten erforderlich, die für das Nachfahren der letzten Redo-Informationen und den anschließenden Restart der (ehemaligen) Standby-Datenbank nötig sind.

Fällt die Primärdatenbank unerwartet aus, so versucht Oracle Net im Fall einer TAF-Konfiguration den zweiten Knoten zu erreichen. Schlägt dies fehl, so kann über die Parameter `retries` und `delay` die Anzahl und die Verzögerung weiterer Verbindungsversuche konfiguriert werden.

```

DGD_TAF=
  (DESCRIPTION=
    (address_list=
      (load_balance=off)
      (failover=on)
    )
    (ADDRESS=(PROTOCOL=TCP)(HOST=frankfurt)(PORT=521))
    (ADDRESS=(PROTOCOL=TCP)(HOST=tokio)(PORT=1521))
  )
  (CONNECT_DATA=
    (SERVICE_NAME = DGD)
  )
)

```