EDITION ORACLE®

Christine Gschoßmann, Klaus Langenegger

Oracle Backup und Recovery
Das Praxisbuch

Für alle Versionen bis einschließlich Oracle 11g

ADDISON-WESLEY

3 Grundlagen des physischen Backups

3.1 Einführung

Das physische Backup einer Oracle-Datenbank ist im Produktivbetrieb eine zwingende Notwendigkeit, um im Fehlerfall beispielsweise bei Ausfall des Betriebssystems, einer oder mehrerer Festplatten oder des Rechners einen konsistenten Datenbestand rekonstruieren zu können.

Physische Backups sind Sicherungen der Dateien, die für die Speicherung und Wiederherstellung der Datenbank benutzt werden, wie Datendateien, Control-Dateien und Offline-Redolog-Dateien. Die Datenbank lässt sich je nach gewählter Sicherungsstrategie und Fehlertyp vollständig oder nur mit Datenverlust wiederherstellen.

Dazu gibt es zwei grundsätzliche Sicherungsmöglichkeiten, die hier im Einzelnen beschrieben werden sollen.

- Offline-Backup: Die Datenbank ist geschlossen; kein Zugriff auf Daten möglich.
- Online-Backup: Die Datenbank ist geöffnet und steht uneingeschränkt zur Verfügung. Die Datenbank muss sich im ARCHIVELOG-Modus befinden.

Ein Offline-Backup ist oft nötig, wenn mehrere Datenbanken oder Anwendungen, die ändernd auf Daten zugreifen, in einem Systemverbund stehen und ein konsistenter Aufsetzpunkt innerhalb dieses Verbunds sichergestellt werden muss. Dazu müssen alle beteiligten Systeme zur gleichen Zeit konsistent angehalten oder beendet und anschließend gesichert werden. So wird gewährleistet, dass im Fall eines Restores der gesamten Systemlandschaft keine Inkonsistenzen auftreten.

Ein Online-Backup benötigt zur Wiederherstellung der Daten alle während des Backups angefallenen Redolog-Dateien. Die während des Backups angefallenen Offline-Redolog-Dateien sollten deshalb ebenso lange wie das Online-Backup selbst aufbewahrt werden. Zusätzlich sollten Offline-Redolog-Dateien unabhängig vom Backup-Zyklus laufend gesichert werden. Für Produktivsysteme empfiehlt es sich, die Offline-Redolog-Dateien mehrfach beispielsweise auf zwei unterschiedliche Bänder zu sichern, um bei eventuellen Fehlern auf den Sicherungsmedien (zum Beispiel Bandlesefehler) eine Ausfallsicherheit zu gewährleisten.

Die Sicherung der Dateien kann auf verschiedene Medien erfolgen und wird entweder mit Betriebssystemmitteln, über den Oracle Recovery Manager oder mit Tools von Drittanbietern auf das jeweilige Backup-Medium kopiert.

Die Backup-Medien (zum Beispiel Platten oder Bandlaufwerke) sollten dabei direkt am Datenbankserver angeschlossen sein.

3.2 Offline-Backup

Unter einem vollständigen Offline-Backup versteht man die physische Sicherung aller für das Wiederherstellen einer Datenbank notwendigen Dateien im ‚kalten' Zustand. Das bedeutet, die Datenbank muss vor dem Backup geschlossen werden. Dadurch wird gewährleistet, dass während dieser Zeit keine Änderungen in der Datenbank stattfinden. Nach einem Offline-Backup der gesamten Datenbank verfügt man über eine konsistente Sicherung der Datenbank.

Es können aber während eines Offline-Backups auch nur einzelne Tablespaces oder Datendateien gemeinsam mit der Control-Datei gesichert werden. Diese partielle Sicherung einer Datenbank kann nicht für eine konsistente Wiederherstellung der gesamten Datenbank verwendet werden. Hierfür werden zusätzlich die Offline-Redolog-Dateien zwischen den einzelnen partiellen Sicherungen benötigt.

Ein vollständiges Offline-Backup muss folgende Dateien beinhalten:

1. alle Datendateien sämtlicher Tablespaces (Temp-Files müssen dabei nicht gesichert werden)
2. mindestens eine Kopie der Control-Datei
3. die Konfigurationsdateien der Datenbank, wie die Parameterdatei (PFILE beziehungsweise SPFILE), tnsnames.ora und listener.ora sowie die Passwortdatei
4. die Online-Redolog-Dateien, um das Öffnen der Datenbank nach einem Restore der kompletten Datenbank ohne weiteres Recovery und ohne die Option `RESETLOGS` zu ermöglichen sowie um das Crash Recovery zu vereinfachen, falls die Datenbank nicht konsistent gestoppt werden konnte.

Die Sicherung der Online-Redolog-Dateien ist jedoch nicht zwingend erforderlich. In vielen Umgebungen werden diese bei einem Offline-Backup nicht gesichert.

Vor einem Offline-Backup muss die Datenbank möglichst konsistent gestoppt werden.

> **Achtung!**
> Wird die Datenbank im NOARCHIVELOG-Modus betrieben, so ist das konsistente Beenden zwingend erforderlich! Dies bedeutet, dass die Datenbank mit `SHUTDOWN NORMAL | IMMEDIATE | TRANSACTIONAL` beendet werden muss.

Dadurch wird sichergestellt, dass keine offenen Transaktionen in der Datenbank existieren. Wird die Datenbank automatisch für das Backup beendet, was beispielsweise bei einem skriptgesteuerten Backup der Fall ist, so sollte immer `SHUTDOWN IMMEDIATE` verwendet werden, um die Datenbank so schnell wie möglich konsistent zu beenden.

`SHUTDOWN IMMEDIATE` lässt keine Neuanmeldungen zu, und es können keine neuen Transaktionen angestoßen werden. Alle offenen Transaktionen werden abgebrochen (Rollback). Wenn noch lang laufende, offene Transaktionen existieren, kann es sein, dass auch diese Methode zur Beendigung der Datenbank länger dauert. Außerdem werden alle Verbindungen der angemeldeten Benutzer getrennt.

Beim `SHUTDOWN NORMAL` können sich zwar keine Benutzer mehr anmelden, die Datenbank wird aber erst beendet, wenn sich alle User abgemeldet haben.

Durch das Stoppen mit SHUTDOWN TRANSACTIONAL wird verhindert, dass neue Transaktionen gestartet werden, allerdings wartet die Datenbank so lange, bis alle offenen Transaktionen abgeschlossen wurden, egal ob erfolgreich (Commit) oder nicht (Rollback). Es sind keine Neuanmeldungen möglich, und nach Beendigung der letzten Transaktion werden alle noch verbundenen Benutzer abgemeldet.

3.2.1 Ermittlung der für das Offline-Backup benötigten Dateien

Um die Dateien zu bestimmen, die für eine Komplettsicherung benötigt werden, stehen unter anderem folgende Möglichkeiten zur Verfügung.

Datendateien

Für die Sicherung der Datenbank müssen nur die aktuellen Datendateien gesichert werden. Manchmal kann es vorkommen, dass sich im File-System noch ‚Leichen' befinden, die vom Löschen nicht mehr benötigter Tablespaces herrühren. Diese Dateien werden für die Sicherung nicht benötigt. Den aktuellen Bestand mit Pfad, Namen und Größe der Dateien kann man über die Views DBA_DATA_FILES oder V$DATAFILE bestimmen.

Pfad und Dateinamen sowie Größe der Datendateien:

```
SQL> SELECT FILE_NAME, BYTES FROM DBA_DATA_FILES;

FILE_NAME                  BYTES
-------------------------- ------------------
C:\ORACLE\GC\USERS02.DBF   5242880
C:\ORACLE\GC\USERS01.DBF   5242880
C:\ORACLE\GC\SYSAUX01.DBF  262144000
...
SQL> SELECT NAME, BYTES FROM V$DATAFILE;

NAME                         BYTES
---------------------------- -----------------
C:\ORACLE\GC\SYSTEM01.DBF    503316480
C:\ORACLE\GC\UNDOTBS01.DBF   26214400
C:\ORACLE\GC\SYSAUX01.DBF    262144000
...
```

Um sich alle Tablespaces – außer den temporären – und ihre zugehörigen Datendateien anzeigen zu lassen, kann folgendes SQL-Statement abgesetzt werden:

```
SQL> SELECT t.NAME "Tablespace", f.NAME "Datafile"
FROM V$TABLESPACE t, V$DATAFILE f
WHERE t.TS# = f.TS#
ORDER BY t.NAME;

Tablespace       Datafile
-------------------------------------------------
SYSAUX           C:\ORACLE\GC\SYSAUX01.DBF
SYSTEM           C:\ORACLE\GC\SYSTEM01.DBF
UNDOTBS1         C:\ORACLE\GC\UNDOTBS01.DBF
USERS            C:\ORACLE\GC\USERS01.DBF
```

Dateien von Temporary Tablespaces/Temp-Files

Temp-Files gehören immer zu Temporary Tablespaces. Wie der Name schon sagt, können in ihnen keine permanenten Objekte wie Tabellen abgelegt werden. Sie werden typischerweise für die Ablage temporärer Datenbankobjekte, die zum Beispiel beim Sortieren entstehen, genutzt.

Nach Abschluss eines Sortiervorgangs, wie zum Beispiel einer `ORDER BY`-Anweisung in einem SQL-Statement, werden die belegten Extents wieder freigegeben. Somit werden diese Dateien nicht für das Backup benötigt.

Allerdings müssen sie bei Datenbanken bis einschließlich 9i nach einem Restore der Datenbank mit dem Befehl `ALTER TABLESPACE ADD TEMPFILE` wieder neu angelegt werden. Temp-Files werden dabei nicht auf allen Betriebssystemen initial mit der Größe aus der Storage-Anweisung des `CREATE/ALTER TABLESPACE`-Kommandos erzeugt, sondern wachsen erst, wenn in ihnen temporäre Objekte abgelegt werden Dies sollte auch bei der Dateisystemüberwachung berücksichtigt werden. Der Vollständigkeit halber soll auch die Bestimmung dieser Dateien hier aufgezeigt werden. Die Views `DBA_TEMP_FILES` und `V$TEMP-FILE` enthalten darüber die notwendigen Informationen.

Pfad und Dateinamen sowie Größe der Temp-Files:

```
SQL> SELECT FILE_NAME, BYTES FROM DBA_TEMP_FILES;
FILE_NAME                      BYTES
------------------------------ --------------------
C:\ORACLE\GC\TEMP01.DBF        20971520
SQL> SELECT NAME, BYTES FROM V$TEMPFILE;
NAME                           BYTES
------------------------------ --------------------
C:\ORACLE\GC\TEMP01.DBF        20971520
```

Konfigurationsdateien

Es gibt zwei unterschiedliche Arten der Parameterdatei: eine sogenannte Text-Initialisierungsparameterdatei init<SID>.ora und eine Serverparameterdatei spfile[<SID>].ora. Eine dieser Dateien muss vorhanden sein. Oracle liest beim Starten der Datenbankinstanz dabei das Verzeichnis `$ORACLE_HOME/dbs` beziehungsweise `%ORACLE_HOME%\database` aus. Finden sich in diesem Verzeichnis mehrere Parameterdateien, entscheidet sich Oracle in folgender Reihenfolge für eine dieser Dateien:

spfile<SID>.ora

spfile.ora

init<SID>.ora

Wird eine Serverparameterdatei *spfile[<SID>].ora* genutzt, kann diese wie folgt bestimmt werden:

```
SQL> SHOW PARAMETER SPFILE
NAME          TYPE        VALUE
------------- ----------- ------------------------------------------------
spfile        string      C:\ORACLE\DB_1\DATABASE\SPFILEGC.ORA
```

Bei Nutzung der Parameterdatei *init<SID>.ora* liegt diese unter `$ORACLE_HOME/dbs` (Unix) oder `%ORACLE_HOME%\database` (Windows).

Achtung!
Eventuell ist in der Datei *init<SID>.ora* nur der Speicherort der Parameterdatei über die Parameter `IFILE = <Pfad>/init<SID>.ora` hinterlegt!

Im selben Verzeichnis wie die Parameterdatei befindet sich auch die Passwortdatei *pwd<SID>.ora*.

tnsnames.ora, listener.ora und sqlnet.ora liegen im Verzeichnis `$ORACLE_HOME/network/admin` beziehungsweise `%ORACLE_HOME%\network\admin` (Windows).

Control-Dateien

Pfade und Namen der Control-Dateien lassen sich über den Parameter `CONTROL_FILES` oder die View `V$CONTROLFILE` ermitteln.

Pfad und Dateinamen der Control-Dateien:

```
SQL> SHOW PARAMETER CONTROL_FILES
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
control_files                        string      C:\ORACLE\PRODUCT\10.2.0\ORADA
                                                 TA\GC\CONTROL01.CTL, C:\ORACLE
                                                 \PRODUCT\10.2.0\ORADATA\GC\CON
                                                 TROL02.CTL, C:\ORACLE\PRODUCT\
                                                 10.2.0\ORADATA\GC\CONTROL03.CTL
```

oder

```
SQL> SELECT NAME FROM V$CONTROLFILE;
NAME
------------------------------------------------------------
C:\ORACLE\PRODUCT\10.2.0\ORADATA\GC\CONTROL01.CTL
C:\ORACLE\PRODUCT\10.2.0\ORADATA\GC\CONTROL02.CTL
C:\ORACLE\PRODUCT\10.2.0\ORADATA\GC\CONTROL03.CTL
```

Online-Redolog-Dateien

Die Online-Redolog-Dateien werden nicht benötigt, um die Datenbank aus einem Offline-Backup wiederherzustellen, vorausgesetzt die Datenbank wurde konsistent geschlossen. Soll ein Offline-Backup zurückgespielt und die Datenbank ohne weiteres Recovery geöffnet werden, muss die Option `RESETLOGS` des `ALTER DATABASE OPEN`-Befehls verwendet werden. Dabei wird die Log-Sequence-Nummer zurückge-

setzt und eine neue Datenbankinkarnation erzeugt. Falls dies vermieden werden soll, müssen auch die Online-Redolog-Dateien gesichert werden.

Die Online-Redolog-Dateien lassen sich am einfachsten über die View `V$LOGFILE` ermitteln.

Pfad und Dateinamen der Online-Redolog-Dateien:

```
SQL> SELECT MEMBER FROM V$LOGFILE;

MEMBER
--------------------------------------------------
C:\ORACLE\GC\REDO02.LOG
C:\ORACLE\GC\REDO01.LOG
C:\ORACLE\GC\REDO03.LOG
```

3.2.2 Ablauf eines Offline-Backups

Ein Offline-Backup gliedert sich grundsätzlich in folgende drei Phasen, wobei sich das Kopieren der Dateien auf das Backup-Medium recht unterschiedlich gestalten kann, je nachdem, welche Backup-Medien verwendet werden (Band, Platte etc.) und mit welchen Mitteln das Kopieren stattfindet (Betriebssystemmittel, RMAN oder Drittanbietertools).

1. Konsistentes Beenden der Datenbank: `SHUTDOWN [ NORMAL ] | [ TRANSACTIONAL ] | [ IMMEDIATE ]`
2. Kopieren der Datendateien, Control-Datei und der Konfigurationsdateien (Betriebssystemmittel, RMAN oder Drittanbietertools)
3. Optional auch die Online-Redolog-Dateien kopieren
4. Starten der Datenbank

Der Betrieb der Datenbank ist nun wieder uneingeschränkt möglich.

3.2.3 Offline-Backup im NOARCHIVELOG-Modus

Wird die Datenbank im NOARCHIVELOG-Modus betrieben, sollte zusätzlich noch Folgendes beachtet werden:

Beim konsistenten Beenden der Datenbank mit `SHUTDOWN NORMAL`, `TRANSACTIONAL` oder `IMMEDIATE` werden alle offenen Transaktionen beendet. Die Datenbank befindet sich dann in einem konsistenten Zustand.

Sollte aus irgendwelchen Gründen ein konsistentes Stoppen der Datenbank nicht beziehungsweise nicht mehr möglich sein, so müssen in diesem Fall zusätzlich auch die Online-Redolog-Dateien gesichert werden, um bei der Wiederherstellung der Datenbank aus diesem Backup ein Crash Recovery zu ermöglichen. Nur so kann die Datenbank alle Transaktionen, die beim Stoppen der Datenbank noch offen waren, konsistent beenden.

Dabei werden die bei erfolgreich beendeten Transaktionen (Commit) geänderten Daten in die Datenbank übernommen und Datenänderungen abgebrochener Transaktionen verworfen (Rollback). Die hierzu erforderlichen Informationen befinden sich in den Online-Redolog-Dateien.

3.3 Online-Backup

Der Hauptvorteil des Online-Backups liegt sicherlich darin, dass die Datenbank im laufenden Betrieb gesichert wird und somit der Betrieb uneingeschränkt möglich ist. Im Gegensatz zum Offline-Backup kann eine Sicherung im laufenden Betrieb natürlich nicht konsistent sein. Um trotzdem im Fehlerfall auf ein Online-Backup zurückgreifen zu können, benötigt man zusätzlich die Offline-Redolog-Dateien, die seit Beginn der Sicherung geschrieben wurden.

Während eines Online-Backups, also zwischen dem Setzen (`BEGIN BACKUP`) und dem Zurücknehmen (`END BACKUP`) des Backup-Modus, werden die Checkpoint-Zeitstempel nicht mehr in die Header der Datendateien, für die das Backup initiiert wurde, geschrieben. Das bedeutet, dass die Header der gesicherten Dateien den Zeitstempel des letzten Checkpoints vor Beginn des Backups aufweisen und Oracle somit bei einem Wiederherstellen dieser Dateien aus der Sicherung feststellen kann, welche Redolog-Dateien nachgefahren werden müssen.

Jede Redolog-Datei, die archiviert wird, erhält eine Log-Sequence-Nummer, die fortlaufend nummeriert ist. Für ein Online-Backup beziehungsweise ein Recovery dieses Backups ist diese Nummer unentbehrlich. Sie sollte deshalb mit dem Online-Backup protokolliert werden.

Ein entscheidender Unterschied zum Offline-Backup ist, dass bei Wiederherstellung der Datenbank aus einem Online-Backup immer ein Recovery der zurückgespielten Datendateien nötig ist, um die Datenbank auf einen konsistenten Stand zu bringen.

Achtung!
Online-Backups sind grundsätzlich nur dann möglich, wenn die Datenbank im ARCHIVELOG-Modus betrieben wird.

Auch für das Online-Backup gilt, dass sämtliche Dateien gesichert werden müssen, die für das Wiederherstellen der Datenbank benötigt werden.

Ein vollständiges Online-Backup besteht aus folgenden Dateien:

- alle Datendateien sämtlicher Tablespaces (Temp-Files müssen dabei nicht gesichert werden)
- alle während des Backups erzeugten Offline-Redolog-Dateien
- mindestens eine Kopie der Control-Datei
- die Konfigurationsdateien der Datenbank, wie die Parameterdatei (pfile beziehungsweise spfile), tnsnames.ora, listener.ora, sqlnet.ora sowie die Passwortdatei

Ebenso wie beim Offline-Backup kann die Sicherung der Dateien auf verschiedene Medien erfolgen und wird entweder mit Betriebssystemmitteln, über den Oracle Recovery Manager oder mit Tools von Drittanbietern auf das jeweilige Backup-Medium kopiert.

3.3.1 Ermittlung der für das Online-Backup benötigten Dateien

Um die Dateien zu ermitteln, die für ein vollständiges Online-Backup benötigt werden, stehen unter anderem folgende Möglichkeiten zur Verfügung.

Datendateien

Informationen über alle Datendateien inklusive Pfad, Name und Größe der Dateien kann man über die Views `DBA_DATA_FILES` oder `V$DATAFILE` bestimmen.

Pfad und Dateinamen sowie Größe der Datendateien:

```
SQL> SELECT FILE_NAME, BYTES FROM DBA_DATA_FILES;
FILE_NAME                  BYTES
-------------------------- ------------------
C:\ORACLE\GC\USERS02.DBF   5242880
C:\ORACLE\GC\USERS01.DBF   5242880
C:\ORACLE\GC\SYSAUX01.DBF  262144000
...
SQL> SELECT NAME, BYTES FROM V$DATAFILE;
NAME                         BYTES
---------------------------- -----------------
C:\ORACLE\GC\SYSTEM01.DBF    503316480
C:\ORACLE\GC\UNDOTBS01.DBF   26214400
C:\ORACLE\GC\SYSAUX01.DBF    262144000
...
```

Um sich alle Tablespaces – außer den temporären – und ihre zugehörigen Datendateien anzeigen zu lassen, kann folgendes SQL-Statement verwendet werden:

```
SQL> SELECT t.NAME "Tablespace", f.NAME "Datafile"
FROM V$TABLESPACE t, V$DATAFILE f
WHERE t.TS# = f.TS#
ORDER BY t.NAME;
Tablespace       Datafile
-------------------------------------------------
SYSAUX           C:\ORACLE\GC\SYSAUX01.DBF
SYSTEM           C:\ORACLE\GC\SYSTEM01.DBF
UNDOTBS1         C:\ORACLE\GC\UNDOTBS01.DBF
USERS            C:\ORACLE\GC\USERS01.DBF
```

Offline-Redolog-Dateien

Die Offline-Redolog-Dateien, die für das Online-Backup benötigt werden, können erst nach Ende des Backups der Datendateien bestimmt werden.

Das Verzeichnis und das Format der Offline-Redolog-Dateien werden dabei am einfachsten über die Parameter `LOG_ARCHIVE_DEST`, `LOG_ARCHIVE_DUPLEX_DEST` oder `LOG_ARCHIVE_DEST_n` sowie `LOG_ARCHIVE_FORMAT` ermittelt.

Wurde eine Flash Recovery Area definiert (Parameter `DB_RECOVERY_FILE_DEST`), können sich die Redolog-Dateien auch in den Verzeichnissen unterhalb des Verzeich-

nisses DB_RECOVERY_FILE_DEST\<SID>\ARCHIVELOG befinden (Details siehe Kapitel 1.1.3, Abschnitt »Relevante Parameter«).

Der Parameter LOG_ARCHIVE_FORMAT kann sich dabei aus folgenden Variablen zusammensetzen:

Variable	Bedeutung
%s	Log-Sequence-Nummer
%t	Thread-Nummer
%a	Aktivierungs-ID
%d	Datenbank-ID
%r	Resetlogs-ID, die sicherstellt, dass eindeutige Namen für die Redolog-Dateien auch über mehrere Inkarnationen der Datenbank erzeugt werden

Tabelle 3.1: Variablen des Parameters LOG_ARCHIVE_FORMAT

Werden dabei Großbuchstaben für die Variablen, zum Beispiel *%S*, verwendet, werden die Werte auf eine (betriebssystemabhängige) Länge festgelegt und mit führenden Nullen aufgefüllt.

Der Name der Datei setzt sich aus diesen beiden Parametern zusammen.

Beispiel:

```
LOG_ARCHIVE_DEST = 'F:\ORACLE\GC1\arch\GC1'
LOG_ARCHIVE_FORMAT = ARC%S.%T
```

Dadurch würde sich für die Datei mit der Log-Sequence-Nummer 787 folgender Name ergeben:

```
F:\ORACLE\GC1\arch\GC1ARC00787.001
```

Die Log-Sequence-Nummer der ersten für die Sicherung notwendigen Offline-Redolog-Datei lässt sich bereits zu Beginn des Backups über folgende Datenbankabfrage feststellen.

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';

SEQUENCE#
------------
787
```

Dies ist die zurzeit aktive Log-Sequence-Nummer, deren Redo-Daten daher noch nicht archiviert sind. Die jüngste archivierte Redolog-Datei ist demnach die mit der Sequence-Nummer 786. In der Alert-Datei der Datenbank findet man dazu den letzten Log-Switch:

```
Tue Apr 18 19:51:38 2008
Thread 1 advanced to log sequence 787
  Current log# 13 seq# 787 mem# 0: F:\...\LOG_G13M1.DBF
  Current log# 13 seq# 787 mem# 1: G:\...\LOG_G13M2.DBF
```

Nach Ende des Online-Backups muss die aktuelle Log-Sequence-Nummer erneut abgefragt werden. Damit erhält man die letzte für ein konsistentes Backup erforderliche Redolog-Datei.

Konfigurationsdateien

Es gibt zwei unterschiedliche Arten der Parameterdatei: eine sogenannte Text-Initialisierungsparameterdatei init<SID>.ora und eine Serverparameterdatei spfile[<SID>].ora.

Wird eine Serverparameterdatei *spfile[<SID>].ora* genutzt, kann diese wie folgt bestimmt werden:

```
SQL> SHOW PARAMETER SPFILE
NAME          TYPE        VALUE
------------- ----------- ------------------------------------------------
spfile        string      C:\ORACLE\DB_1\DATABASE\SPFILEGC.ORA
```

Bei Nutzung der Parameterdatei *init<SID>.ora* liegt diese unter `$ORACLE_HOME/dbs` (Unix) oder `%ORACLE_HOME%/database` (Windows).

Achtung!
Eventuell ist in der Datei *init<SID>.ora* nur der Speicherort der Parameterdatei über die Parameter `IFILE = <Pfad>/init<SID>.ora` hinterlegt!

In demselben Verzeichnis wie die Parameterdatei befindet sich auch die Passwortdatei *pwd<SID>.ora*.

tnsnames.ora, *listener.ora* sowie *sqlnet.ora* liegen im Verzeichnis `$ORACLE_HOME/network/admin` (Unix) beziehungsweise `%ORACLE_HOME%\network\admin` (Windows).

Control-Dateien

Die Control-Datei muss für ein Online-Backup gesondert betrachtet werden. Anders als bei einem Offline-Backup kann keine Kopie der vorhandenen Control-Dateien verwendet werden, da diese fortlaufend aktualisiert werden (SCN etc.). Stattdessen muss zuerst ein Backup der Control-Datei erzeugt werden.

Mit folgendem Befehl wird eine binäre Datei erzeugt, die anschließend für das Online-Backup verwendet werden kann.

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO '<pfad>/<dateiname>';
```

Es kann aber auch eine Textdatei generiert werden, mittels derer eine neue Control-Datei erzeugt werden kann.

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO TRACE;
```

Die Datei wird in dem Verzeichnis angelegt, das durch den Initialisierungsparameter `USER_DUMP_DEST` festgelegt wurde. Ab Oracle 11g wird die Textdatei in das Verzeichnis `Diag Trace` geschrieben. Das Verzeichnis `Diag Trace` kann über die View `V$DIAG_INFO` (`SELECT * FROM V$DIAG_INFO`) ermittelt werden.

Nähere Informationen befinden sich in Abschnitt 1.1.4, Control-Dateien.

3.3.2 Durchführung eines Online-Backups

Anders als beim Offline-Backup wird für das Online-Backup die Datenbank nicht geschlossen. Deshalb muss der Datenbank der Beginn der Sicherung mitgeteilt werden. Dies erfolgt durch den Befehl `BEGIN BACKUP`. Bis Oracle 9i konnte das Kommando nur für die einzelnen Tablespaces abgesetzt werden. Ab Oracle 10g kann auch die gesamte Datenbank mit allen Datendateien in den Online-Backup-Modus gesetzt werden.

Aus diesem Grund werden beide Möglichkeiten im Folgenden betrachtet.

Online-Backup eines Tablespace

1. Bestimmen der aktuellen Log-Sequence-Nummer

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';
```

   Alle Offline-Redolog-Dateien, die von nun an während des Backups erzeugt werden, also alle größer oder gleich dieser Log-Sequence-Nummer, werden für die Sicherung benötigt.

2. Tablespace in den Backup-Modus setzen

```
SQL> ALTER TABLESPACE <tablespace_name> BEGIN BACKUP;
```

3. Kopieren beziehungsweise Sichern aller zum Tablespace gehörenden Datendateien
4. Tablespace wieder aus dem Backup-Modus nehmen

```
SQL> ALTER TABLESPACE <tablespace_name> END BACKUP;
```

   Sollen mehrere Tablespaces innerhalb dieses Backups gesichert werden, müssen die Aktionen *Tablespace in den Backup-Modus* setzen, *Kopieren der zugehörigen Datendateien* und *Tablespace aus Backup-Modus nehmen* für diese Tablespaces wiederholt werden.

5. Backup der Control-Datei erzeugen

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO '<pfad>/<dateiname>' [REUSE];
```

   Die Angabe von `REUSE` ist nötig, wenn die Datei bereits existiert und überschrieben werden soll. Dies ist beispielsweise dann der Fall, wenn für das Online-Backup immer wieder derselbe Ablageort zur Sicherung der Control-Datei verwendet wird.

6. Bestimmen der aktuellen Log-Sequence-Nummer

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';
```

   Die Redolog-Datei mit dieser Log-Sequence-Nummer ist damit die letzte, die für ein konsistentes Backup erforderlich ist. Da es sich dabei noch um eine Online-Redolog-Datei handelt, in welche die aktuellen Datenbankänderungen ge-

schrieben werden, muss die Datei archiviert werden, um sie sichern zu können. Dies wird durch einen Log-Switch veranlasst.

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
```

Dadurch wird auf die nächste Log-Sequence-Nummer umgeschaltet, und einer der Archiver-Prozesse schreibt die Informationen dieser Online-Redolog-Datei in das ARCHIVELOG-Verzeichnis. Nach Abschluss dieser Aktion können alle Redolog-Dateien, beginnend mit der anfangs abgefragten Log-Sequence-Nummer bis einschließlich der zuletzt archivierten Datei, auf das Backup-Medium kopiert werden.

7. Kopieren der Konfigurationsdateien und der Sicherung der Control-Datei

Damit ist das Online-Backup des Tablespace abgeschlossen. Trotzdem sollte zur Sicherheit noch kontrolliert werden, ob wirklich alle Datendateien des gesicherten Tablespace wieder aus dem Backup-Modus genommen wurden. Dazu wird die View V$BACKUP verwendet. Der Status der Dateien muss den Wert NOT ACTIVE aufweisen.

```
SQL> SELECT f.FILE_NAME "Dateiname", b.STATUS "Status"
     FROM DBA_DATA_FILES f, V$BACKUP b
     WHERE f.TABLESPACE_NAME='<tablespace_name>'
     AND f.FILE_ID = b.FILE#
     ORDER BY f.FILE_NAME;
```

Werden hier noch Dateien ausgegeben, die den Status ACTIVE besitzen, so kann entweder für jede Datei einzeln, für alle Dateien eines Tablespace oder für alle Datendateien der Datenbank der Backup-Modus zurückgesetzt werden. Bevor man jedoch den Status für alle Datendateien zurücksetzt, sollte sichergestellt sein, dass kein weiteres Online-Backup läuft.

```
SQL> ALTER DATABASE DATAFILE '<pfad>/<dateiname>' END BACKUP;
SQL> ALTER TABLESPACE END BACKUP;
SQL> ALTER DATABASE END BACKUP;
```

Damit haben alle Datendateien den Backup-Modus wieder verlassen, und die Zeitstempel der Checkpoints werden wieder in die Header der Datendateien geschrieben.

Online-Backup der gesamten Datenbank

Eine konsistente Online-Sicherung der gesamten Datenbank besteht aus den Kopien aller Datendateien sowie sämtlichen Änderungen, die während dieser Sicherung in der Datenbank durchgeführt wurden. Das bedeutet, dass alle Redolog-Dateien, die im Laufe der Sicherung erzeugt werden, zwingend für die Wiederherstellung eines konsistenten Zustands der Datenbank aus diesem Backup erforderlich sind. Außerdem werden die Informationen über die Datenbankstruktur zu diesem Zeitpunkt benötigt. Deshalb muss auch eine Kopie der Control-Datei erzeugt werden, die ebenso Bestandteil dieser Sicherung ist.

Zu Beginn der Sicherung muss also erst einmal die aktuelle Log-Sequence-Nummer ermittelt werden.

1. Bestimmen der aktuellen Log-Sequence-Nummer

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';
```

   Alle Offline-Redolog-Dateien, die von nun an während des Backups erzeugt werden, also alle größer oder gleich dieser Nummer, werden für die Sicherung benötigt.

2. Datenbank in den Backup-Modus setzen

```
SQL> ALTER DATABASE BEGIN BACKUP;
```

3. Kopieren aller Datendateien der Datenbank (keine Temp-Files)
4. Datenbank wieder aus dem Backup-Modus nehmen

```
SQL> ALTER DATABASE END BACKUP;
```

5. Backup der Control-Datei erzeugen

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO '<pfad>/<dateiname>' [REUSE];
```

   Die Angabe von `REUSE` ist nötig, wenn die Datei bereits existiert und überschrieben werden soll. Dies ist beispielsweise dann der Fall, wenn für das Online-Backup immer wieder derselbe Ablageort zur Sicherung der Control-Datei verwendet wird.

6. Bestimmen der aktuellen Log-Sequence-Nummer

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';
```

   Die Redolog-Datei mit dieser Sequence-Nummer ist damit die letzte, die für ein konsistentes Backup erforderlich ist. Da es sich dabei noch um eine Online-Redolog-Datei handelt, in welche die aktuellen Datenbankänderungen geschrieben werden, muss die Datei archiviert werden, um sie sichern zu können. Dies wird durch einen Log Switch veranlasst.

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
```

   Dadurch wird auf die nächste Log-Sequence-Nummer umgeschaltet, und einer der Archiver-Prozesse schreibt die Informationen dieser Online-Redolog-Dateien in das ARCHIVELOG-Verzeichnis. Nach Abschluss dieser Aktion können alle Redolog-Dateien, beginnend mit der anfangs abgefragten Log-Sequence-Nummer bis einschließlich der zuletzt archivierten Datei, auf das Backup-Medium kopiert werden.

7. Kopieren der Konfigurationsdateien und der Sicherung der Control-Datei

Damit ist das Online-Backup der Datenbank abgeschlossen. Trotzdem sollte zur Sicherheit noch kontrolliert werden, ob wirklich alle Datendateien wieder aus dem Backup-Modus genommen wurden. Dazu wird die View `V$BACKUP` verwendet. Der Status der Dateien muss den Wert *NOT* ACTIVE aufweisen.

```
SQL> SELECT f.FILE_NAME "Dateiname", b.STATUS "Status"
     FROM DBA_DATA_FILES f, V$BACKUP b
     WHERE f.FILE_ID = b.FILE#
     ORDER BY f.FILE_NAME;
```

Oder um nur die unterschiedlichen Status der Dateien auszugeben:

```
SQL> SELECT distinct b.STATUS "Status"
     FROM DBA_DATA_FILES f, V$BACKUP b
     WHERE f.FILE_ID = b.FILE#;
```

Werden hier noch Dateien ausgegeben, die den Status ACTIVE besitzen, beziehungsweise wird bei der Ausgabe des zweiten SQL-Statements auch noch der Status ACTIVE zurückgeliefert, so kann entweder für jede Datei einzeln, für alle Dateien eines Tablespace oder für alle Datendateien der Datenbank der Backup-Modus zurückgesetzt werden.

```
SQL> ALTER DATABASE DATAFILE 'dateiname' END BACKUP;

SQL> ALTER TABLESPACE END BACKUP;
```

Ab Oracle 9i:

```
SQL> ALTER DATABASE END BACKUP;
```

Damit haben alle Datendateien den Backup-Modus wieder verlassen, und die Zeitstempel der Checkpoints werden wieder in die Header der Datendateien geschrieben.

3.4 Backup großer Datenbanken

Waren noch vor wenigen Jahren Datenbanken mit mehr als 100 Gigabyte eine Ausnahme, so sind heute selbst Datenbanken im Terabyte-Bereich nichts Ungewöhnliches mehr. Das Sicherungskonzept solch großer Datenbanken will aber wohl überlegt sein. Oft treten erst Probleme auf, wenn sich so ein Monstrum im Laufe der Jahre entwickelt hat und die Sicherung mit den bisherigen Mitteln und Methoden nicht mehr handhabbar ist. So dauert die Durchführung eines vollständigen Backups zu lange, oder es lassen sich die in den SLAs (Service Level Agreements) vereinbarten Restore-Zeiten mit den bestehenden Möglichkeiten und Ressourcen nicht mehr einhalten.

Ansatzweise sollen hier einige Möglichkeiten aufgezeigt werden, die das Sichern und Wiederherstellen großer Datenbanken beschleunigen und erleichtern können.

Dazu muss die im Rahmen von Backup und Restore verwendete Hardware (zum Beispiel Bandlaufwerke, Plattensysteme inklusive Controller und I/O-Busse) so weit wie möglich optimiert werden. Die Möglichkeiten, die sich hier bieten, sollten mit dem Hardware-Hersteller erörtert und abgestimmt werden.

Des Weiteren bieten auch noch andere Bereiche zum Teil erhebliches Verbesserungspotenzial, um Backup und Restore zu beschleunigen.

3.4.1 Parallelisierung

Eine deutliche Reduzierung der Backup-Laufzeiten kann durch die parallele Sicherung auf mehrere Band- oder Plattenlaufwerke gleichzeitig erzielt werden. Außerdem sollten die Band- oder Plattenlaufwerke direkt am Datenbankserver angeschlossen sein, das heißt, die Sicherung sollte nicht über das Netzwerk erfolgen.

Bei der Planung paralleler Backups spielen auch die Kapazität sowie der Durchsatz der Bandlaufwerke und die Zugriffszeiten der Festplatten eine Rolle. Weiterhin muss auch der maximale Durchsatz der System- und I/O-Busse berücksichtigt werden. Auch die Anzahl und Leistung der CPUs sollten beim Parallelisierungsgrad des Backups berücksichtigt werden.

3.4.2 Inkrementelle Backups

Die Verwendung von RMAN, dem Oracle Recovery Manager, bietet die Möglichkeit eines inkrementellen Backups (Level 1 Backup). Dabei werden nur die seit der letzten Vollsicherung (Level 0 Backup) oder dem letzten inkrementellen Backup geänderten Blöcke gesichert. Dadurch verringert sich das zu sichernde Datenvolumen.

Ab Oracle 10g kann zusätzlich das Block Change Tracking aktiviert werden, bei dem eine Betriebssystemdatei angelegt wird, in der Oracle die Datenbankblöcke protokolliert, die seit der letzten Vollsicherung geändert wurden. Bei einer inkrementellen Sicherung liest RMAN dann diese Datei aus, um die zu sichernden Blöcke zu bestimmen. Bis einschließlich Oracle 9i musste RMAN noch alle Blöcke der Datendateien lesen, um die Blöcke für das Backup zu bestimmen – auch die nicht veränderten. Durch die Verwendung des Block Change Trackings ergibt sich in Oracle 10g eine deutliche Laufzeitverbesserung für das inkrementelle Backup.

Der Nachteil inkrementeller Sicherungen ist der, dass bei einem Restore eine Vollsicherung und eine oder mehrere inkrementelle Sicherungen zurückgespielt werden müssen. Da bei einer Vollsicherung mit RMAN aber nur die gefüllten Datenbankblöcke gesichert werden, ist dies nicht unbedingt mit längeren Restore-Zeiten als bei anderen Sicherungsmethoden verbunden. Dazu mehr in Kapitel 9, Backup und Recovery über RMAN.

3.4.3 Partielle Backups

Wenn es nicht möglich ist, die gesamte Datenbank innerhalb eines für das Backup zur Verfügung stehenden Zeitfensters zu sichern, so besteht die Möglichkeit, nur Teile der Datenbank zu sichern, wie einzelne Tablespaces oder gar nur eine oder mehrere Datendateien. Für ein vollständiges Backup der Datenbank wird das Backup auf mehrere Zeitfenster wie zum Beispiel Nächte verteilt. Pro Nacht wird dann nur eine bestimmte Teilmenge der Datenbank gesichert.

Dies hat allerdings zur Folge, dass im Falle eines Restores Dateien aus unterschiedlichen Sicherungen zurückgespielt werden müssen und sich somit die Auszeit für das Recovery verlängert, da mehr Offline-Redolog-Dateien eingespielt werden müssen.

Ein partielles Backup kann offline oder online erfolgen und wird analog zu einer Vollsicherung durchgeführt mit dem Unterschied, dass eben nicht alle, sondern nur bestimmte Datendateien gesichert werden.

Ein gravierender Nachteil von partiellen Backups ist die Überwachung und Anpassung der einzelnen Backups, die in ihrer Gesamtheit eine vollständige Sicherung der Datenbank ergeben. Nur wenn ALLE partiellen Backups erfolgreich abgeschlossen und ALLE notwendigen Dateien der Datenbank gesichert wurden, ist eine vollständige Sicherung der Datenbank vorhanden. Bei einer Wiederherstellung aus einem partiellen Backup werden zusätzlich noch Offline Redologs benötigt, um die Datendateien unterschiedlicher Sicherungszeitpunkte wieder auf einen einheitlichen konsistenten Stand zu bringen. Auch Strukturänderungen an der Datenbank wie beispielsweise neue Datendateien müssen berücksichtigt und gegebenenfalls in die Definition der partiellen Backup-Jobs aufgenommen werden.

Eine sinnvolle Nutzung von partiellen Backups besteht in der Sicherung einzelner Tablespaces mit sehr häufigen und vielen Datenänderungen zusätzlich zu vollständigen Backups. Bei einem Recovery einzelner oder aller Datendateien dieser Tablespaces müssen so nur die seit diesem partiellen Backup angefallenen Redolog-Dateien zurückgespielt werden, also weniger Offline-Redolog-Dateien als seit der letzten vollständigen Datenbanksicherung erzeugt wurden.

3.4.4 Zwei-Phasen-Backup

Unter einem Zwei-Phasen-Backup versteht man die Aufteilung der Sicherung in zwei Schritte:

Im ersten Schritt wird eine Sicherung auf einen Plattenbereich durchgeführt, die wesentlich schneller ist als eine Bandsicherung. Anschließend wird das Backup auf Band gesichert.

Die Vorteile ergeben sich zum einen aus der verkürzten Sicherungsdauer und zum anderen durch die Möglichkeit eines schnellen Restore vom Plattenbereich, sofern das zuletzt erstellte Backup zurückgespielt werden muss.

Als Nachteil ist hier der zusätzlich nötige Plattenplatz zu sehen. Außerdem müssen hier zwei Backups statt wie bei einem konventionellen Backup eines durchgeführt werden. Ein spezieller Fall des Zwei-Phasen-Backups ist das Split-Mirror-Backup (siehe Kapitel 3.5).

3.4.5 Snapshots

Verschiedene Hardware-Hersteller bieten eine sogenannte Snapshot-Methode an. Dabei werden alle Daten zu einem bestimmten Zeitpunkt auf I/O-Ebene eingefroren.

Snapshots werden entweder bei geschlossener Datenbank (offline) oder im Backup-Modus (online) erzeugt. Werden anschließend wieder Daten geändert, werden diese beim nächsten Snapshot zusätzlich abgespeichert.

Mithilfe von Snapshots können sehr schnell Backups (meist in wenigen Sekunden) auf Platte erzeugt werden. Dabei ist es möglich, mehrere Snapshots von unterschiedlichen Zeitpunkten zu erzeugen. Der zusätzlich dafür benötigte Plattenplatz hält sich in Grenzen, da jeweils nur die geänderten Blöcke gesichert werden. Bei längerem Vorhalten der Snapshots ist zudem ein schneller Restore der Datenbank möglich.

Während der Snapshots kann es zu Performance-Einbußen beziehungsweise Ressourcenengpässen kommen.

3.4.6 Multisection-Backups

Seit Oracle 11g bietet RMAN auch die Möglichkeit, große Datendateien stückweise zu sichern. Das Multisection-Backup bietet den Vorteil, dass große Datendateien parallel gesichert werden können. Dabei wird die Datendatei in einzelne Bereiche aufgeteilt, die aus einer Reihe aufeinanderfolgender Datenbankblöcke bestehen. Ein Backup-Set (siehe Abschnitt 9.3) beinhaltet stets alle Sektionen einer Datendatei.

3.5 Split-Mirror-Backup

Diese Technologie ermöglicht eine sehr schnelle Durchführung eines Backups. Bei einer Split-Mirror-Lösung wird zunächst blockweise eine vollständige Kopie eines oder mehrerer Volumes angelegt. Unter Volumes (unter Windows auch Partitionen oder Laufwerke) versteht man das Zusammenfassen einer oder mehrerer Platten zu einer logischen Einheit.

Dabei werden die gespiegelten Daten mit den Originalen synchronisiert. Anschließend kann der (Platten-)Spiegel geteilt werden, um ein identisches Abbild des Originals zu diesem Zeitpunkt festzuhalten.

Im Datenbankumfeld kann dieses Verfahren für eine schnelle Datensicherung, online wie offline, auch sehr großer Datenbanken genutzt werden. Selbst für eine Offline-Sicherung beträgt die Ausfallzeit für das Backup, je nachdem, wie lange das Auftrennen des Spiegels dauert, im Regelfall nur wenige Minuten. Auch die Last auf dem Datenbankserver durch die Sicherung reduziert sich bei Verwendung eines eigenen Sicherungsservers, der die gespiegelten Daten übernimmt. Allerdings stellt eine Split-Mirror-Lösung auch einen hohen Kostenfaktor dar.

Für die weitere Betrachtung einer Split-Mirror-Lösung wird vorausgesetzt, dass folgende Dateien gespiegelt werden:

- alle Datendateien
- mindestens eine Kopie der Control-Datei
- die Konfigurationsdateien
- die Offline-Redolog-Dateien

Die Online-Redolog-Dateien spielen dabei nur eine untergeordnete Rolle. Sie werden auf der Spiegelseite nicht benötigt. Dabei wird natürlich vorausgesetzt, dass die Datenbank im ARCHIVELOG-Modus betrieben wird (siehe Kapitel 1.6.1).

Für den Fall, dass ein Offline-Backup zurückgespielt wird und die Datenbank ohne weiteres Recovery geöffnet werden soll, muss die Option `RESETLOGS` des `ALTER DATABASE OPEN`-Befehls verwendet werden. Dabei wird die Log-Sequence-Nummer zurückgesetzt und eine neue Datenbankinkarnation erzeugt. Soll dies vermieden werden, müssen auch die Online-Redolog-Dateien gespiegelt werden.

Ein weiterer Vorteil der Plattenspiegelung ist die Möglichkeit eines schnellen Restores aus der letzten Sicherung, die sich ja noch auf der »Spiegelseite« befindet. Muss aufgrund eines aufgetretenen Problems die Datenbank aus dem letzten Backup wieder hergestellt werden, kann dies schnell und ohne großen Aufwand durchgeführt werden. Der Zeitaufwand für einen Restore der Datenbank und das anschließende Recovery verkürzt sich dadurch auf ein Minimum.

Folgende Punkte sind jedoch im Fall eines Restores der Datenbank zu beachten:

- Die Offline-Redolog-Dateien müssen aus der Redolog-Sicherung zurückgespielt werden, da die neueren Dateien des Originals durch die auf der Spiegelseite befindlichen älteren Dateien ersetzt werden.

  Dies kann dadurch vermieden werden, dass entweder dieser Bereich nicht überschrieben oder die Offline-Redolog-Dateien in mehrere Archivelog-Verzeichnisse archiviert werden (Kapitel 1.1.3, Parameter `LOG_ARCHIVE_DEST_n` oder `LOG_ARCHIVE_DUPLEX_DEST`).

- Die Konfigurationsdateien, sollten sie auf Originalseite seit dem Auftrennen des Spiegels geändert worden sein, dürfen ebenfalls nicht durch die Kopien ersetzt werden. Gegebenenfalls müssen sie nach jeder Änderung separat gesichert werden.

- Auch die Control-Dateien sollten, je nach Szenario, nicht durch die Kopien ersetzt werden.

Die Durchführung des Backups unterscheidet sich von der konventionellen Methode im Großen und Ganzen nur dadurch, dass statt der Sicherung der Datendateien auf der Originaldatenbank erst der Spiegel geteilt wird und anschließend auf der Spiegelseite das Backup erfolgt. Somit kann ein Backup fast wie gewohnt durchgeführt werden.

Nach Auftrennen des Spiegels müssen die Dateien noch auf ein entsprechendes Backup-Medium zur längeren Aufbewahrung kopiert werden, da ja beim nächsten Synchronisieren der Spiegelseite die Dateien wieder überschrieben werden. Das Synchronisieren der beiden Spiegelhälften wird immer erst kurz vor der Sicherung durchgeführt, wodurch für die restliche Zeit ein vollständiges Backup der Datenbank auf Platte vorgehalten wird.

Im weiteren Verlauf dieses Kapitels wird vorausgesetzt, dass sich die Datenbank im ARCHIVELOG-Modus befindet und die Offline-Redolog-Dateien unter Berücksichtigung der Aufbewahrungsfrist laufend gesichert und auf den Backup-Medien vorgehalten werden.

3.5.1 Split-Mirror-Offline-Backup

Wie beim herkömmlichen Offline-Backup besteht auch dieses Backup grundsätzlich aus drei Schritten. Zusätzlich muss die Datenbankkopie, die sich nach dem Auftrennen der Volumes auf der Spiegelseite befindet, noch auf entsprechende Backup-Medien kopiert werden.

Achtung!
Bevor das Backup durchgeführt wird, muss sichergestellt sein, dass die Spiegelseite vollständig synchronisiert wurde, die Blöcke von Original und Spiegel also identisch sind.

Vorausgesetzt, dass sich sowohl die Datendateien als auch die Konfigurationsdateien und zumindest eine Kopie der Control-Datei auf den gespiegelten Volumes befinden, läuft ein Backup folgendermaßen ab:

1. Konsistentes Beenden der Originaldatenbank: `SHUTDOWN [ NORMAL ] | [ TRANSACTIONAL ] | [ IMMEDIATE ]`
2. Auftrennen des Spiegels
3. Starten der Originaldatenbank

Nach dem Auftrennen des Spiegels befindet sich eine komplette Sicherungskopie auf der Spiegelseite. Diese kann nun auf entsprechende Backup-Medien gesichert werden.

3.5.2 Split-Mirror-Online-Backup

Ein Online-Backup ist auch in diesem Umfeld etwas aufwendiger als ein Offline-Backup. Trotzdem bietet eine Split-Mirror-Umgebung auch für eine Online-Sicherung einige Vorteile. Der Zeitaufwand ist sehr gering, und das Kopieren der Dateien wird wesentlich erleichtert.

Im Unterschied zum Offline-Backup muss in dieser Umgebung eine Kopie der Control-Datei manuell erzeugt werden. Vorausgesetzt, dass sich sowohl die Datendateien als auch die Konfigurationsdateien auf den gespiegelten Volumes befinden, wird ein Online-Backup vereinfacht wie folgt durchgeführt:

1. Bestimmen der aktuellen Log-Sequence-Nummer

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';
```

   Alle Offline-Redolog-Dateien, die von nun an während des Backups erzeugt werden, also alle größer oder gleich dieser Log-Sequence-Nummer, werden für die Sicherung benötigt. In diesem Fall werden dabei durch die stark verkürzte Backup-Dauer aber wesentlich weniger Redolog-Dateien erzeugt.

2. Datenbank in den Online-Backup-Modus setzen

```
SQL> ALTER DATABASE BEGIN BACKUP;
```

3. Auftrennen des Spiegels

4. Datenbank wieder aus dem Backup-Modus nehmen

```
SQL> ALTER DATABASE END BACKUP;
```

5. Backup der Control-Datei erzeugen

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO '<pfad>/<dateiname>' [REUSE];
```

   Die Angabe von `REUSE` ist nötig, wenn die Datei bereits existiert und überschrieben werden soll, beispielsweise wenn für das Online-Backup immer wieder derselbe Ablageort zur Sicherung der Control-Datei verwendet wird. Die dadurch erzeugte Datei ist ein wichtiger Bestandteil des Backups und muss auch auf der Spiegelseite verfügbar sein.

6. Bestimmen der aktuellen Log-Sequence-Nummer

```
SQL> SELECT SEQUENCE# FROM V$LOG WHERE STATUS='CURRENT';
```

   Die Redolog-Datei mit dieser Log-Sequence-Nummer ist damit die letzte, die für ein konsistentes Backup erforderlich ist. Da es sich dabei um eine Online-Redolog-Datei handelt, in welche die aktuellen Datenbankänderungen geschrieben werden, muss die Datei archiviert werden, um sie sichern zu können. Dies wird durch einen Log-Switch veranlasst.

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
```

   Dadurch wird auf die nächste Log-Sequence-Nummer umgeschaltet, und einer der Archiver-Prozesse schreibt die Informationen dieser Online-Redolog-Datei in das ARCHIVELOG-Verzeichnis. Nach Abschluss dieser Aktion können alle Redolog-Dateien, beginnend mit der anfangs abgefragten Log-Sequence-Nummer bis einschließlich der zuletzt archivierten Datei, auf das Backup-Medium kopiert werden. Dieses Backup der Redolog-Dateien ist gemeinsam mit der Datenbankkopie auf Spiegelseite vorzuhalten.

7. Abschließend müssen auch hier alle für das Backup relevanten Dateien (Datendateien, Offline-Redolog-Dateien, Kopie der Control-Datei und die Konfigurationsdateien), die sich jetzt auf der abgetrennten Spiegelseite befinden, auf die dafür vorgesehenen Backup-Medien kopiert werden, da sie sonst durch das nächste Backup überschrieben werden.

Das Backup ist damit erfolgreich beendet.

3.6 Standby-Datenbank und Backup

Oracle bietet zwei unterschiedliche Typen von Standby-Datenbanken an: die logische und die physische. Bei der hier im Zusammenhang mit Backup und Recovery beschriebenen Datenbank handelt es sich stets um eine physische Standby-Datenbank.

Eine physische Standby-Datenbank ist eine Kopie einer produktiven Datenbank, die aus einem Backup der Primärdatenbank erzeugt wurde und die transaktional

konsistent in Echtzeit oder mit Zeitversatz durch das Nachfahren der Offline-Redolog-Dateien (Redo-Apply) aus der Primärdatenbank aktualisiert wird.

Die Standby-Datenbank befindet sich dabei im MOUNT-Status und wird sinnvollerweise auf einem separaten Server betrieben, da sie meist der Ausfallsicherheit der Produktionsdatenbank dient. Voraussetzung für den Betrieb einer Standby-Datenbank ist deshalb der aktivierte ARCHIVELOG-Modus. Außerdem muss die Übertragung der Offline-Redolog-Dateien von der Primär- zur Standby-Datenbank sichergestellt sein. Dies kann beispielsweise durch ein gemeinsames ARCHIVELOG-Verzeichnis oder durch automatisiertes, skriptgesteuertes Kopieren erfolgen.

Dabei müssen in den Oracle-Versionen bis einschließlich 10g die beiden Server die gleiche Betriebsystem-Plattform besitzen sowie ein identisches Oracle-Release inklusive Patch-Level. Seit Oracle 11g unterstützt Data Guard auch unterschiedliche Prozessorarchitekturen, Betriebssysteme und 32- und 64-Bit-Kombinationen.

Bei Ausfall der Produktionsdatenbank kann die Standby-Datenbank aktiviert werden und so die Funktionen der produktiven Datenbank übernehmen.

Des Weiteren werden Standby-Datenbanken auch zum Abfangen logischer Fehler genutzt. Dies wird durch das zeitversetzte Einspielen der Änderungen der Primärdatenbank in die Standby-Datenbank erreicht. Damit kann beispielsweise nach dem versehentlichen Löschen einer Tabelle in der Primärdatenbank das Einspielen der Offline-Redolog-Dateien angehalten werden. Wurde die Änderung (Löschen der Tabelle) noch nicht in die Standby-Datenbank eingespielt, kann die Standby-Datenbank durch ein Point-in-Time-Recovery bis kurz vor diesem Zeitpunkt aktualisiert werden. Anschließend übernimmt die Standby-Datenbank die Funktion der Primärdatenbank (Switchover). Die Änderungen seit dem Löschen der Tabelle in der Primärdatenbank sind dann allerdings verloren.

Außerdem kann eine Standby-Datenbank auch für das Reporting und weitere Auswertungen genutzt werden. Dazu muss sie allerdings im Read-only-Modus geöffnet und kann während dieser Zeit nicht aktualisiert werden.

Mit Oracle 11g steht auch eine sogenannte Snapshot-Standby-Datenbank zur Verfügung. Eine physische Standby-Datenbank kann schnell zu einer Snapshot-Standby-Datenbank konvertiert werden.

Die Snapshot-Standby-Datenbank wird im Read-Write-Modus geöffnet und kann dann zum Beispiel für Testzwecke genutzt werden. Während die Datenbank im Read-Write-Modus geöffnet wird, werden weiterhin die Offline-Redolog-Dateien der Primärdatenbank übertragen, aber es findet kein Redo-Apply statt.

Die Snapshot-Standby-Datenbank kann mit nur einem SQL-Befehl wieder in eine physische Standby-Datenbank konvertiert werden (`ALTER DATABASE CONVERT TO PHYSICAL STANDBY;`). Dabei wird Flashback Database genutzt (siehe Abschnitt 7.2.4). Während die Snapshot-Standby-Datenbank geöffnet ist, kann kein Switchover stattfinden.

Oracle Data Guard bietet ein reichhaltiges Angebot für den Betrieb von Standby-Datenbanken vom Erzeugen über die Verwaltung bis hin zum Monitoring. Oracle Data Guard ist ein Feature der Oracle Database Enterprise Edition und bedarf wie

RMAN keiner zusätzlichen Installation. RMAN und Data Guard können dabei gemeinsam für die Administration einer Data Guard-Konfiguration genutzt werden.

Näheres zum Thema Standby-Datenbanken und Oracle Data Guard findet man in der Oracle-Dokumentation oder in entsprechender Fachliteratur wie zum Beispiel *Oracle 10g Hochverfügbarkeit* von Andrea Held, erschienen im Addison-Wesley-Verlag.

Das Hauptaugenmerk in diesem Buch liegt aber auf den Möglichkeiten, die eine Standby-Datenbank im Hinblick auf Backup und Restore bietet.

Ein großer Nutzen der Standby-Datenbank liegt darin, dass sie anstelle der Primärdatenbank gesichert werden kann. Außerdem stellt sie auch ein immer aktuelles Backup der produktiven Datenbank dar.

Dadurch ergeben sich folgende Vorteile:

- Die primäre Datenbank steht bei einem Offline-Backup der Standby-Datenbank während der Sicherung uneingeschränkt zur Verfügung.
- Der Datenbankserver der primären Datenbank wird nicht durch die I/O-Operationen des Backups belastet.
- Sämtliche Systemressourcen des produktiven Servers stehen auch während der Sicherung für den Online-Betrieb zur Verfügung.
- Alle Systemressourcen des Standby-Servers können für das Backup genutzt werden.
- Die Server von Primär- und Standby-Datenbank können räumlich voneinander getrennt sein und bieten dadurch eine sehr hohe Ausfallsicherheit.
- Kurze Ausfallzeiten in Fehlersituationen, bei denen ein Restore der Datenbank erforderlich ist. Dabei kann die Standby-Datenbank nach dem Recovery der letzten Redolog-Dateien sofort die Aufgaben der Primärdatenbank übernehmen. Es müssen weder Datendateien noch Offline-Redolog-Dateien von Band zurückgespielt werden, da sich alle für das Recovery notwendigen Dateien bereits auf dem Standby-Server befinden.
- Die Offline-Redolog-Dateien werden durch das Einspielen in die Standby-Datenbank auf Korruptionen und Lesbarkeit überprüft.
- Zusätzlich lassen sich durch eine Standby-Datenbank auch Wartungsarbeiten ohne längere Auszeiten durchführen, da die Standby-Datenbank während dieser Zeit die Aufgaben der Primärdatenbank übernehmen kann.
- Bei logischen Fehlern kann auf der Standby-Datenbank temporär ein `FLASHBACK DATABASE` (Beschreibung siehe Kapitel 7.2.4) auf einen Zeitpunkt vor dem Fehler durchgeführt werden. Dadurch können die Daten auf der Primärdatenbank wesentlich einfacher wiederhergestellt werden.

Nachteil einer Standby-Datenbank ist neben den hohen Kosten für die meist doppelten Komponenten auch der erhöhte Aufwand für die Administration dieser Lösung.

Vorgehensweise für die Komplettsicherung

Ein Offline-Backup einer Standby-Datenbank, ohne dabei näher auf Data Guard und RMAN einzugehen, unterscheidet sich in einigen Punkten von einem normalen Offline-Backup.

Zuerst müssen die Dateien bestimmt werden, die für eine Komplettsicherung der Datenbank benötigt werden. Für das Backup müssen die Datendateien der Standby-Datenbank und die Control-Datei sowie die Konfigurationsdateien der Primärdatenbank gesichert werden, da ja im schlimmsten Fall die Primärdatenbank wiederhergestellt werden soll.

1. Datendateien:

   Informationen über alle Datendateien inklusive Pfad, Namen und Größe der Dateien kann man über die View `V$DATAFILE` bestimmen. Die View `DBA_DATA_FILES` ist bei nicht geöffneter Datenbank – also auch im MOUNT-Status, in dem sich die Standby-Datenbank meist befindet – nicht verfügbar.

   Pfad und Dateinamen sowie Größe der Datendateien:

   Standby-Datenbank:

```
SQL> SELECT NAME, BYTES FROM V$DATAFILE;

NAME                          BYTES
----------------------------  -----------------
C:\ORACLE\STBY\SYSTEM01.DBF   503316480
C:\ORACLE\STBY\UNDOTBS01.DBF  26214400
C:\ORACLE\STBY\SYSAUX01.DBF   262144000
...
```

   Um sich alle Tablespaces – außer den temporären – und ihre zugehörigen Datendateien anzeigen zu lassen, kann folgendes SQL-Statement abgesetzt werden:

```
SQL> SELECT t.NAME "Tablespace", f.NAME "Datafile"
     FROM V$TABLESPACE t, V$DATAFILE f
     WHERE t.TS# = f.TS#
     ORDER BY t.NAME;

Tablespace       Datafile
-------------------------------------------------
SYSAUX           C:\ORACLE\STBY\SYSAUX01.DBF
SYSTEM           C:\ORACLE\STBY\SYSTEM01.DBF
UNDOTBS1         C:\ORACLE\STBY\UNDOTBS01.DBF
USERS            C:\ORACLE\STBY\USERS01.DBF
```

   Falls sich die Verzeichnisstrukturen von Primär- und Standby-Datenbank unterscheiden, sind die Initialisierungsparameter `LOG_FILE_NAME_CONVERT` und/oder `DB_FILE_NAME_CONVERT` in der Primär- und der Standby-Datenbank zu setzen.

   Zum Beispiel:

```
LOG_FILE_NAME_CONVERT = 'C:\ORACLE\GC\', 'C:\ORACLE\STBY\'
DB_FILE_NAME_CONVERT = 'C:\ORACLE\GC\', 'C:\ORACLE\STBY\'
```

An erster Stelle steht dabei der Pfad der Primärdatenbank gefolgt vom Pfad auf der Standby-Seite.

Bei einem Restore des Backups zum Wiederherstellen der Primärdatenbank muss darauf geachtet werden, dass die Dateien in das richtige Verzeichnis zurückgesichert werden.

2. Control-Datei:

   Das Backup der Control-Datei wird als binäre Kopie aus der **Primär**datenbank erzeugt.

3. Konfigurationsdateien:

   Für das Backup werden die Konfigurationsdateien von der **Primär**datenbank benötigt. Es gibt zwei unterschiedliche Arten der Parameterdatei: eine sogenannte Text-Initialisierungsparameterdatei INIT.ORA und eine Serverparameterdatei SPFILE.

   Wird eine Serverparameterdatei *SPFILE* genutzt, kann diese wie folgt bestimmt werden:

Primärdatenbank:

```
SQL> SHOW PARAMETER SPFILE

NAME         TYPE        VALUE
------------ ----------- ------------------------------------------------
spfile       string      C:\ORACLE\DB_1\DATABASE\SPFILEGC.ORA
```

Bei Nutzung der Parameterdatei *INIT.ORA* liegt diese unter `$ORACLE_HOME/dbs` (Unix) oder `%ORACLE_HOME%/database` (Windows).

Achtung!
Eventuell ist in der Datei *INIT.ORA* nur der Speicherort der Parameterdatei über die Parameter `IFILE = <Pfad>/init<SID>.ora` hinterlegt!

Im selben Verzeichnis wie die Parameterdatei befindet sich auch die Passwortdatei *pwd<SID>.ora*.

tnsnames.ora, *listener.ora* und *sqlnet.ora* liegen im Verzeichnis `$ORACLE_HOME/network/admin` beziehungsweise `%ORACLE_HOME%\network\admin` (Windows).

Da die Standby-Datenbank laufend durch das Einspielen der Offline-Redolog-Dateien der Primärdatenbank aktualisiert wird, muss zuerst das Recovery durch folgenden Befehl angehalten werden.

Standby-Datenbank:

```
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
```

Dies kann unter Umständen einige Zeit dauern, falls gerade eine Offline-Redolog-Datei eingespielt wird.

Nun wird eine Kopie der Control-Datei der Primärdatenbank erzeugt. Dazu wird folgendes Statement auf der **Primär**datenbank abgesetzt:

Primärdatenbank:

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO '<pfad>/<dateiname>' [REUSE];
```

Die Option `REUSE` ist dabei wieder für den Fall, dass die Datei bereits vorhanden ist und überschrieben werden soll.

Jetzt befindet sich die Datenbank in konsistentem Zustand und kann mittels `SHUT DOWN` gestoppt werden:

Standby-Datenbank:

```
SQL> SHUTDOWN IMMEDIATE;
```

Das vollständige Backup besteht nun aus folgenden Dateien:

- Datendateien der **Standby**-Datenbank
- Konfigurationsdateien der **Primär**datenbank
- Kopie der Control-Datei der **Primär**datenbank

Wurden die Dateien gesichert, kann die Standby-Datenbank wieder gestartet und in den Recovery-Modus gebracht werden. Die Vorgehensweise hierfür unterscheidet sich in den Oracle-Versionen 9i und den Nachfolgeversionen von 9i.

- Oracle 9i

```
SQL> STARTUP NOMOUNT;
SQL> ALTER DATABASE MOUNT STANDBY DATABASE;
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

- Ab Oracle 10g

```
SQL> STARTUP MOUNT;
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

Durch die Angabe von `DISCONNECT FROM SESSION` wird der Prozess in den Hintergrund geschickt, und man gelangt wieder zum SQL-Prompt zurück.

Die Standby-Datenbank wird jetzt wieder mit den Offline-Redolog-Dateien der Primärdatenbank aktualisiert. Da während der Sicherung in der Primärdatenbank aber neue Redolog-Dateien erzeugt wurden, ist die Standby-Datenbank noch nicht auf dem aktuellen Stand und muss den Rückstand erst wieder aufholen.