Physische Backups

Ein gültiges Backup ist, einfach und allgemein formuliert, eine Kopie aller erforderlichen Informationen in einer Datenbank, mit deren Hilfe sich die Datenbank wiederherstellen lässt, falls sie unbrauchbar wurde. Der Verlust eines Laufwerks und das zufällige Löschen einer Datenbankdatei oder -tabelle sind zwei Beispiele dafür, wann eine Datenbank unbrauchbar wird. Stützt sich das Backup-Schema auf Backup-Images der Datenbank und die Archivierung von Logdateien, sind natürlich Kopien von Datendateien, Steuerdateien und Online Redo Logdateien vorzuhalten. Verlieren Sie eine der archivierten Redo Logdateien, spricht man von einem *Loch* in der Sequenz der Dateien. Ein Loch in den archivierten Logdateien macht ein Backup ungültig, aber die Datenbank kann über ein Roll-forward bis zu dem Punkt aktualisiert werden, an dem das Loch beginnt. Besitzen Sie beispielsweise 25 archivierte Redo Logdateien und die Redo Logdatei 15 fehlt, kann das Roll-forward nur bis Log 14 erfolgen.

Ein robustes Backup-Schema ist eine Methode, mit der man sicherstellt, dass man gültige Backups erhält. Wichtig für ein robustes Backup-Schema ist das Wissen um den physischen Standort der Datendateien, die Reihenfolge der Ereignisse beim Backup-Prozess und die Fähigkeit zur Behandlung bestimmter Fehler, die bei einem Backup auftreten können. Ein robustes Backup-Schema zeigt bei Medien-, Programm- und Bedienungsfehlern eine gewisse Fehlertoleranz.

DBAs arbeiten üblicherweise mit physischen und logischen Backups. Bei einem physischen Backup werden die Datenbankdateien an ein Sicherungsziel kopiert. Ein logisches Backup nutzt zum Lesen der Daten in der Datenbank ein Oracle-Werkzeug (**export**), und speichert die Daten und Definitionen auf Systemebene in einer Binärdatei. Die Oracle-Datenbank und das Betriebssystem, auf dem Oracle läuft, bieten zahlreiche Leistungsmerkmale, mit denen sich robuste Backup-Schemata realisieren lassen. Dieses Kapitel bietet eine Übersicht über die physischen Sicherungsprozeduren, die von Oracle angeboten werden, und die Backup-Befehle, die Sie auf den verschiedenen Betriebssystemen einsetzen können. Zusätzlich fließen bei der Planung von Backup-Prozeduren Designbetrachtungen für Decision-Support (DSS)- und OLTP-Systeme ein,

die mit sehr großen Datenbanken (VLDB) arbeiten. Das nächste Kapitel beschäftigt sich mit den logischen Backups von Oracle-Datenbanken.

3.1 Datenbank-Design und grundlegende Backup-Regeln

Bevor wir uns mit den Online- und Offline-Backup-Prozeduren beschäftigen, sollte man die Regeln zur Hinterlegung der Datenbankdateien und andere Designbetrachtungen verstanden haben, die sich auf das Backup-Schema auswirken können. Nachfolgend finden Sie einige einfache Regeln, die das Backup-Schema auf den Ausfall eines Laufwerks oder Bandgeräts vorbereiten, und die Zeit für Recoverys erheblich verkürzen können:

1. Allgemein wird empfohlen, die Logdateien auf einem Laufwerk zu sichern (d.h., Sie setzen das Archivziel so, dass die archivierten Logdateien auf einem Laufwerk angelegt werden), und sie erst später auf ein Sicherungsband zu kopieren. Das Ziel für diese Dateien sollte allerdings nicht auf ein physisches Laufwerk zeigen, auf dem auch Datenbankdateien oder Online Redo Logdateien liegen. Geht eine Datenbankdatei oder die aktuelle Redo Logdatei verloren, benötigt man für ein Recovery die archivierten Logdateien. Geht die archivierte Redo Logdatei oder eine Online Redo Logdatei verloren, die aktuell nicht aktiv ist, sollte die aktuelle Datenbank mithilfe einer Online- oder Offline-Backup-Prozedur gesichert werden, die alle Datenbankdateien auf ein Sicherungsmedium kopiert (Laufwerk oder Band). Danach können die Operationen fortgesetzt werden. Wenn Sie beim Anlegen der Datenbank mit dem **create database**-Befehl den Wert für den Parameter MAXLOGFILES auf einen Wert größer 2 setzen, erleichtert das ein Recovery nach dem Verlust einer inaktiven Online Redo Logdatei. Eine ausführliche Diskussion der Recovery-Prozeduren finden Sie in Kapitel 6.
2. Werden Datenbankdateien auf ein physisches Laufwerk gesichert, und eine Datenbankdatei befindet sich auf demselben Gerät wie ihre Backup-Kopie, kann man nicht von einer adäquaten Sicherung sprechen. Zur Verwaltung der Sicherungen sollten Sie ein oder mehrere separate Laufwerke vorhalten. Die Sicherung von Dateien auf Laufwerken kann das Recovery beschleunigen, da die Dateien nicht vom Band gelesen werden müssen.
3. Die Steuerdatei sollte mehrfach hinterlegt werden; eine Kopie der Steuerdatei sollte man auf unterschiedliche Laufwerke verteilen, die jeweils an separate Controller angeschlossen sind. Um der Datenbank eine neue Steuerdatei hinzuzufügen, fahren Sie die Datenbank herunter, ändern den INIT.ORA-Parameter CONTROL_FILES und starten die Datenbank neu. Für Einzelheiten zu diesem Thema siehe Kapitel 2.

4. Die Online-Logs sollte man multiplexen und pro Gruppe sollte man mindestens zwei Logelemente vorhalten. Die beiden Elemente einer Loggruppe sollten nicht auf dem gleichen physischen Gerät liegen, da dies dem Sinn und Zweck des Multiplexing widerspricht.
5. Manchmal kann es vorteilhaft sein, *Ersatzlaufwerke* vorzuhalten. Dabei handelt es sich um zusätzliche leere Laufwerke, die sich beim Ausfall einer Platte sofort online setzen lassen.
6. Das Multiplexen von archivierten Redo Logdateien ermöglicht in vielen Fällen die Wiederherstellung bei mehreren gleichzeitigen Medienausfällen. Werden die Logdateien beispielsweise auf Laufwerken archiviert, auf Bänder kopiert und danach von den Platten gelöscht, kann es zu Datenverlusten kommen, wenn sowohl das Band als auch Datendateien verloren gehen. Deshalb wird empfohlen, die Kopien der Sicherungen sowohl auf Band als auch auf einer Platte vorzuhalten. Oracle8*i* ermöglicht über den INIT.ORA-Parameter LOG_ARCHIVE_DEST_*n* das automatisierte Multiplexen der archivierten Logdateien, wobei *n* ein Integer von 1 bis 5 ist. Wir empfehlen, die Logdateien zumindest an zwei Zielen zu hinterlegen.
7. Ein Roll-forward einer Datenbank oder Datenbankdatei aus einer Sicherung lässt sich in vielen Fällen vereinfachen und beschleunigen, wenn man alle benötigten Redo Logdateien auf Laufwerken vorhält. Vielen Systeme benötigen für die Rücksicherung der archivierten Redo Logdateien vom Band relativ viel Zeit.
8. Unmittelbar nach einer Änderung der Datenbankstruktur durch Hinzufügen, Umbenennen oder Löschen einer Datendatei (eine Datendatei lässt sich nur löschen, wenn man den dazugehörigen Tablespace löscht), sollte man die Steuerdatei sichern, da sie das Schema der Datenbank enthält. Zusätzlich sollte man auch die neuen Datendateien sichern. Die Steuerdatei lässt sich bei geöffneter Datenbank mit folgendem Befehl sichern:

   ```
   SQL> alter database backup controlfile to 'filespec';
   ```

9. Falls Sie in Ihrem Unternehmen mehrere Oracle-Datenbanken einsetzen, sollten Sie mit dem Oracle Recovery Manager und einem Recovery-Katalog arbeiten. Damit reduzieren Sie die möglichen Benutzerfehler bei Backups und Recoverys. Für Einzelheiten zu diesem Programm siehe Kapitel 7.
10. Nicht vergessen sollte man die Identifizierung der Bänder, auf denen die Sicherungen gespeichert sind. Die Wahl der richtigen Namenskonvention und die Kennzeichnung der Bänder ist äußerst wichtig. Mehr zu diesem Thema finden Sie im weiteren Verlauf dieses Kapitels.

Nachfolgend ein Beispiel für eine typische Sicherungsstrategie, bei der die oben vorgestellten Regeln angewendet wurden:

1. Lassen Sie die Datenbank im ARCHIVELOG-Modus laufen.
2. Führen Sie mindestens einmal pro Woche eine Offline-Sicherung aus, wenn Ihre Datenbank nicht ständig verfügbar sein muss. Falls die Datenbank 24x7 Stunden pro Woche zur Verfügung stehen muss, sollten Sie täglich Online-Sicherungen ziehen.
3. Sichern Sie mindestens alle vier Stunden die archivierten Redo Logdateien. Die Anzahl der zu sichernden Dateien hängt von der Größe der Logdateien und der Anzahl der generierten Redos ab.
4. Exportieren Sie einmal wöchentlich die gesamte Datenbank (oder führen Sie einen inkrementellen, kumulativen bzw. bei großen Datenbanken einen Export auf Tabellenebene durch) im RESTRICT-Modus. Bei Datenbanken im Dauerbetrieb führen Sie diesen Export außerhalb der Kernzeiten aus, wenn keine oder zumindest wenige Zugriffe erfolgen.

3.2 Physische Backups

Ein physisches Backup ist eine Sicherung, bei der man die tatsächlichen physischen Blöcke von einem Standort zum anderen kopiert. Sie können die Datenbankdateien, je nach Backup-Prozedur von Laufwerk zu Laufwerk oder von Laufwerk auf Band kopieren. Zur Sicherung der Datenbank über physische Backups bietet Oracle zwei Möglichkeiten an.

Die erste Option ist die Sicherung der Datenbankdateien nach dem Herunterfahren der Datenbank mit dem Befehl **shutdown normal**. Diesen Backup-Typ bezeichnet man als *offline* oder *kalt*, da die Datenbank bei Ausführung der Sicherung nicht aktiv ist. Manche DBAs binden den Offline-Backup in die Systemsicherung ein, was bedeutet, dass man das Betriebssystem samt Oracle-Dateien sichert. Der DBA muss nur darauf achten, dass die Datenbank vor dem Sichern heruntergefahren wird.

Die zweite Option ist die Ausführung einer physischen Sicherung bei geöffneter Datenbank. Diese Variante empfiehlt sich in allen Fällen, in denen die Datenbank permanent läuft. Dieser Backup-Typ heißt *online* oder *heiß*, da die Datenbank zum Zeitpunkt der Sicherung online und in Betrieb ist. Bei Online-Backups sind allerdings noch einige spezielle Schritte auszuführen.

3.2.1 Offline (kalte)-Backups

Der erste Schritt bei einem Offline-Backup ist das Herunterfahren der Datenbank mit der NORMAL-Option. Falls Sie die Datenbank mit ABORT oder IMMEDIATE beenden, sollten Sie die Datenbank im RESTRICT-Modus neu starten und mit NORMAL herunterfahren, bevor Sie die Datenbankdateien kopieren. Dann benutzen Sie ein Dienstprogramm des Betriebssystems, um alle Datenbank- und Steuerdateien zu kopieren. In die Sicherung sind auch alle ungesicherten archivierten Redo Logdateien einzubeziehen. Die Online Redo Logs sollten nicht gesichert werden, da sie im Zusammenhang mit der Wiederherstellung der Datenbank keine Rolle spielen.

Eine allgemeine Regel empfiehlt, die Datenbank mindestens einmal pro Woche mit einem Offline-Backup zu sichern. Die Anzahl der auszuführenden Sicherungen hängt allerdings von den jeweiligen Anforderungen ab.

Manche DBAs führen die Sicherungen manuell aus und verzichten auf die Automatisierung der Backup-Prozedur. Bei diesem Ansatz können allerdings Probleme auftreten. So wissen Sie nach dem Herunterfahren der Datenbank unter Umständen nicht mehr, wie viele Dateien in der Datenbank vorhanden sind und wo sie liegen. Oder Sie vergessen vielleicht, eine kurz zuvor eingefügte neue Datendatei zu sichern. In Situationen wie diesen ziehen einige DBAs einen DUMP der Steuerdatei und erhalten so die gewünschten Informationen über die Daten- und die Logdateien (d.h., $ strings *controlfile* für UNIX; $dump *controlfile* für VMS). Einfacher ist es allerdings, die gewünschten Informationen direkt aus der Datenbank abzuholen. Die Data Dictionary Views DBA_DATA_FILES oder V$DATAFILE, V$LOGFILE und V$CONTROLFILE zeigen alle Daten-, Redo Log- und Steuerdateien an, die in der Datenbank liegen. (Zum Abholen dieser Informationen aus den $Views brauchen Sie die Datenbank nicht zu öffnen – sie muss lediglich gemountet sein). Automatisiert man die Backup-Prozeduren, erleichtert man sich die Arbeit und reduziert die menschlichen Fehlermöglichkeiten. Für die Automatisierung der Operationen sind allerdings einige Skripte zu erstellen. Beispiele dazu finden Sie in Kapitel 5.

Da die Datenblöcke bei den Sicherungen physisch von der Quelle zum Ziel kopiert werden, fallen fehlerhafte Blöcke nicht unbedingt auf. Anders formuliert werden die Fehler von der Datendatei an die Sicherung weitergereicht. Solche Fehler erkennt man erst beim Rücksichern der Daten. Deshalb sollten Sie Ihre Datenbanksicherungen unbedingt testen – dazu gehört die Rücksicherung der Daten und die Ausführung eines Roll-forward. Vor einem Recovery lässt sich auch eine Fehlersituation simulieren. Die Sicherungen sollten Sie mindestens alle drei Monate testen. Sollte es der Geschäftsbetrieb zulassen, kann das natürlich auch häufiger geschehen.

3.2.2 Offline-Backups durchführen

Um für eine Datenbank eine Offline-Sicherung zu erstellen, sind folgende Schritte auszuführen:

1. Vorbereitung der Sicherung.
   a Richten Sie auf Systemebene eine Textdatei ein, die den Start der Sicherung markiert.
   b Deaktivieren Sie die Logon-Möglichkeit für die Applikation.
   c Verschicken Sie Meldungen, dass die Datenbank in Kürze nicht mehr zur Verfügung steht.
   d Fahren Sie die Datenbank entweder mit **shutdown normal** oder **shutdown immediate** herunter.
2. Führen Sie die Sicherung aus.
   a Löschen Sie die archivierten Redo Logs des letzten Tages aus dem Sicherungsbereich. (Wir setzen voraus, dass Sie Ihre Datenbank täglich sichern.)
   b Verschieben Sie die aktuellen archivierten Logdateien in den Sicherungsbereich.
   c Kopieren Sie die Datendateien, Steuerdateien und Online Redo Logs in den entsprechenden Sicherungsbereich auf den Laufwerken.
3. Beenden Sie die Prozedur.
   a Starten Sie die Datenbank.
   b Aktivieren Sie das Login für die Anwendungen.
   c Kopieren Sie das Datenbank-Image (Daten-, Steuer-, Online Log- und archivierte Logdateien) auf Band.
   d Zuletzt löschen Sie die Datei, die den Beginn der Sicherung markierte.

Mit dem Schritt 1A stellen Sie sicher, dass die Sicherungsprozedur für eine Datenbank nicht zufällig zweimal läuft. Die Schritte 2A und 2B halten die Redo Logdateien so lange auf dem Laufwerk, bis sie für das Online-Recovery nicht mehr benötigt werden, sorgen aber nicht dafür, dass mehrere identische Kopien auf Platte vorgehalten werden. Allerdings kann man die Sicherungen von mehreren Tagen bereithalten: Dazu kopieren Sie zuerst die Sicherung des letzten Tages in den Backup-Bereich des vorletzten Tages, danach kopieren Sie in Schritt 3C den zwei Tage alten Backup-Bereich auf Band.

In einer OFA-konformen Datenbank lassen sich alle Datenbankdateien mit einem einzigen Befehl manipulieren. Die systemseitigen Backup-Prozeduren und die dazugehö-

rigen Befehle finden Sie in Tabelle 3-1. Ausgewählte Befehle werden im weiteren Verlauf des Kapitels diskutiert.

Tabelle 3-1: Backup-Prozeduren und Befehle für verschiedene Betriebssysteme.

Betriebssystem	Systemprozedur	Befehle
UNIX	Cron-Job	obackup, cpio, tar, dd, fbackup, ...
VMS	Batch-Job	Backup
Windows NT	Interaktiv	Backup Manager (nur bei Oracle8 and frühren Versionen) oder das OCOPY-Werkzeug
MAC	Interaktiv	GUI Finder zum Kopieren auf Platte, Software von Fremdherstellern
OS/2	Interaktiv	Die DOS/OS2 copy-Befehle
Novell NetWare	Interaktiv	NBACKUP-Werkzeug, Software von Fremdherstellern
MVS	JCL-Submit	DFDSS oder IDCAMS mit einem EXPORT (nicht REPRO)

3.2.3 Online (heiße)-Backups

Wenn Ihr Unternehmen den Dauerbetrieb der Datenbank an sieben Tagen in der Woche voraussetzt, sollten Sie Online- oder heiße Backups ausführen. Voraussetzung für diese Sicherungsvariante ist, dass die Datenbank im ARCHIVELOG-Modus läuft. Andernfalls liefert Oracle eine Fehlermeldung und verweigert die Ausführung der Online-Sicherungsprozedur. Die Online-Backups ähneln den Offline-Sicherungen, mit dem Unterschied, dass zwei zusätzliche Schritte auszuführen sind. Vor dem Start der Sicherung sollten Sie den Befehl **begin backup,** und am Ende noch ein **end backup** eingeben. Diese beiden Befehle erübrigen sich allerdings, wenn Sie mit dem Recovery Manager arbeiten.

Ein Beispiel:

```
SQL> alter tablespace users begin backup;
SQL> alter tablespace users end backup;
```

Diese beiden Befehle werden vor und nach der Sicherung des Tablespace USERS eingegeben.

Im Gegensatz zu den Offline-Backups, mit denen man die gesamte Datenbank sichert, ist die Einheit eines Online-Backups ein Tablespace: Man sichert je nach Bedarf bestimmte oder alle Tablespaces. Das Online-Backup umfasst die Sicherung der Datendateien (für einen oder mehrere Tablespaces), die aktuelle Steuerdatei und alle archivier-

ten Redo Logdateien, die in den Intervallen zwischen den Sicherungen angelegt wurden. Alle archivierten Redo Logdateien, die nach dem Backup generiert wurden, werden im Falle einer vollständigen Wiederherstellung ebenfalls benötigt. Da die Sicherungseinheit eines Online-Backups ein Tablespace ist, sind gegebenenfalls alle Datendateien zu sichern. Dass dieser Umstand sehr wichtig ist, soll folgendes Beispiel zeigen.

Nehmen wir an, Sie haben in Ihrer Datenbank drei Tablespaces – T1, T2 und T3 – und Sie sichern jede Nacht Teile der Datenbank mit Online-Backups. Das heißt, montags sichern Sie T1, dienstags T2, mittwochs T3 und am nächsten Tag beginnt die Sequenz wieder von vorne. Das bedeutet, dass Sie alle drei Tage über eine vollständige Sicherung der Datenbank verfügen, obwohl Sie im Gegensatz zu einem Offline-Backup nicht alle Tablespaces gleichzeitig sichern. Sollte am Freitag nach Ausführung der Online-Sicherung ein Medienfehler auftreten, bei dem alle Datenbankdateien verloren gehen, müssen Sie die gesamte Datenbank aus den bestehenden Sicherungen wiederherstellen. Da nur die Sicherungen für zwei Tablespaces (T1 und T2 von Donnerstag und Freitag) aktuell sind, müssen Sie die Daten für T3 aus der Mittwoch-Sicherung zurückschreiben. Das bedeutet, dass man zusätzlich auch alle archivierten Redo Logdateien ab der Mittwoch-Sicherung wiederherstellen muss.

Dieses Beispiel zeigt, dass ein Recovery mit Teilsicherungen (Datendateien, die zu unterschiedlichen Zeiten gesichert wurden) bei der ältesten Datenbankdatei beginnt. Deshalb sind alle archivierten Redo Logdateien aufzuheben, die bis zur ältesten Sicherung zurückreichen. Zudem ist sicherzustellen, dass regelmäßig die gesamte Datenbank gesichert wird, damit die Sicherungen aller Datenbankdateien zur Verfügung stehen.

Die Vorteile der Online-Backups im Vergleich zu den Offline-Backups sind wie folgt:

- Den Benutzern steht während der Online-Backups die gesamte Datenbank zur Verfügung, inklusive der zu sichernden Tablespaces.
- Nicht alle Datendateien müssen zum gleichen Zeitpunkt gesichert werden, es sind auch Teilsicherungen möglich. Zur vollständigen Wiederherstellung der Datenbank führt man auf die teilgesicherten Tablespaces die Redo Logs aus.

3.2.4 Beispiel für eine Online-Backup-Prozedur

Bei einem Online-Backup sind folgende Schritte auszuführen:

1. Die Datenbank sollte sich im ARCHIVELOG-Modus befinden. Falls nicht, mounten Sie die Datenbank und geben folgende Befehle ein:

```
SQL> alter database archivelog;
SQL> archive log start
SQL> alter database open;
```

Der erste Befehl setzt die Datenbank in den ARCHIVELOG-Modus. Der zweite Befehl aktiviert die automatische Archivierung (startet den ARCH-Prozess), und der dritte Befehl öffnet die Datenbank.

2. Im nächsten Schritt suchen wir nach der ältesten Online Logsequenz. Dazu geben wir folgenden Befehl ein:

```
SQL> archive log list
   Database log mode              ARCHIVELOG
   Automatic archival             ENABLED
   Archive destination            /oracle/oradata/test/archive
   Oldest online log sequence     59
   Next log sequence to archive   61
   Current log sequence           61
```

Sie müssen alle archivierten Logdateien ab der Sequenznummer 59 als Teil des Online-Backups aufbewahren. Obwohl das Recovery bei der SCN (System Change Number) beginnt, wo der Backup startete (bei der Logsequenznummer 61), sollten Sie aus Vorsicht alle archivierten Logdateien ab der *ältesten Online Logsequenznummer* vorhalten. (Die SCN wird in Kapitel 6 besprochen).

3. Setzen Sie den Tablespace, den Sie sichern möchten, in den HOT BACKUP-Modus:

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

4. Sichern Sie mit einem Systembefehl alle Datenbankdateien, die zum Tablespace gehören.

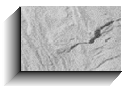
5. Setzen Sie den Tablespace wieder in den NO HOT BACKUP-Modus zurück:

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

Wiederholen Sie für jeden Tablespace, den Sie sichern möchten, die Schritte 3 bis 5.

6. Führen Sie zum Erhalt der aktuellen Logsequenznummer erneut den Befehl **archive log list** aus. Das ist die letzte Redo Logdatei, die in den Online-Backup einzubeziehen ist. Erzwingen Sie einen Logwechsel, damit Oracle eine neue archivierte Logdatei anlegt. Dazu nutzen Sie folgenden Befehl:

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



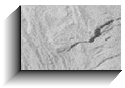
Hinweis

*Beim Recovery der Datenbank mit Online-Backups sind mindestens alle archivierten Online Redo Logs einzusetzen, die zwischen den Befehlen **begin backup** und **end backup** angelegt wurden. Deshalb ist es wichtig, alle archivierten Logdateien zu sichern. Für die Durchführung einer vollständigen Wiederherstellung werden alle Redo Logdateien benötigt.*

7. Sichern Sie alle archivierten Logdateien (die in den Schritten 2 und 6 ermittelt wurden) mit einem Systembefehl. Die Online Redo Logs sollte man niemals sichern, da die Online Logdatei den *End of Backup*-Marker besitzt, was bei einem Recovery zu Fehlermeldungen führen würde.
8. Sichern Sie die Steuerdatei mit folgendem Befehl:

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO 'filespec';
```

Beim Sichern der Steuerdatei auf ein Laufwerk sollten Sie darauf achten, dass auch die Kopie der Steuerdatei auf Band gesichert wird.



Hinweis

Steuerdateien sollten nach jeder Änderung des Datenbankschemas gesichert werden. Zu diesen Änderungen gehören das Hinzufügen, Löschen oder Umbenennen einer Log- oder Datendatei. Neue Datenbankdateien sollte man unmittelbar nach dem Einfügen sichern.

Interne Operationen von heißen Backups

Die internen Mechanismen der heißen Backups sollte man unbedingt kennen. Nach der Eingabe des Befehls **alter tablespace begin backup** werden die Dateien, die zum Tablespace gehören, als *hot-backup-in-progress* markiert. Führt man die Sicherung vor diesem Befehl aus, ist der Backup nutzlos. Dieser Befehl führt für alle Dateien, die sich im HOT BACKUP-Modus befinden, einen Checkpoint aus. Das heißt, dass alle Dirty Buffers, die zu den Dateien im HOT BACKUP-Modus gehören, auf Platte geschrieben werden. Die *Checkpoint-SCN* des Datei-Headers (siehe Kapitel 6) wird auf die festgelegte SCN erhöht, die bei Ausführung des **begin backup**-Befehls abgefangen wurde. Das ist insofern wichtig, als die Checkpoint-SCN in den Sicherungsdateien die gleiche sein muss wie zu Beginn des Backups, und Oracle kann nicht garantieren, dass der Datei-Header der erste Block ist, der von der Systemsicherung kopiert wird. Nach dem ersten Checkpoint werden nun weitere Checkpoints die Sicherung der Datei-Header verhindern, die sich im HOT BACKUP-Modus befinden.

Ist der INIT.ORA-Parameter _LOG_BLOCKS_DURING_BACKUP auf TRUE (den Standardwert) gesetzt, beginnt der Befehl **alter tablespace begin backup** mit dem Logging von ganzen Block-Images bei der ersten Blockänderung. Warum? Weil es bei einem Recovery wichtig sein kann, eine Kopie des gesamten Blocks zu besitzen. Zur Erklärung dieser Operation muss man das Phänomen der *gesplitteten Blöcke* kennen. Ist die Oracle-Blockgröße ein Mehrfaches der Betriebssystemblöcke, ist es in Abhängigkeit davon, wie das Betriebssystem die Blöcke kopiert, möglich, dass ein heißer Backup eine inkonsistente Version eines gegebenen Datenblocks enthält. Wird ein Block beispielsweise zwischen den Leseoperationen auf der Platte aktualisiert, kann die Kopie

in der Sicherungsdatei nutzlos sein, da die vordere und die hintere Hälfte eines Blocks zu verschiedenen Zeitpunkten geschrieben worden sein können. Durch das Logging des Images eines Datenblocks in der Redo Logdatei vor der ersten Änderung lässt sich das Image später bei einem Recovery zur Rekonstruktion eines gesplitteten Blocks einsetzen.

Der Checkpoint, der bei Ausführung des Befehls **alter tablespace begin backup** gesetzt wird, stellt sicher, dass nur die Blöcke an die Redo Logdateien geschrieben werden, die während der heißen Backups geändert werden. Das erklärt, warum beim HOT BACKUP-Modus besonders viele Redos generiert werden. Sollte ein Block längere Zeit im Cache verbleiben, wird er nur einmal geloggt; wird er aber auf Platte geschrieben und erneut in den Cache gelesen (weiterhin im HOT BACKUP-Modus), wird das erste Image des Blocks erneut geloggt.

Zur Prüfung der Blockkonsistenz vor einem Recovery wird die Versionsnummer am Blockanfang mit der Versionsnummer am Ende des Blocks verglichen. Aus diesem Vergleich kann ermittelt werden, ob der Block während eines heißen Backups gesplittet wurde. Sind die beiden Versionsnummern gleich, ist der Block konsistent. Andernfalls wird eine konsistente Version des Blocks benötigt: Dazu wird vor der Ausführung der Redo-Änderungen das erste Image des Blocks aus der Redo auf das Laufwerk kopiert.

Je mehr DMLs (d.h., **insert**-, **update**- oder **delete**-Operationen) man während eines heißen Backups auf die Datendateien ausführt, desto mehr Redos werden in diesem Zeitraum generiert. Deshalb empfiehlt der Oracle Support, die Backups zu Zeiten auszuführen, wenn nur wenig DML-Aktivitäten zu verzeichnen sind. Zudem sollte man den heißen Backup eines Tablespace stets mit dem Befehl **alter tablespace end backup** abschließen, bevor der nächste Tablespace gesichert wird. Sind beispielsweise zwei Tablespaces zu sichern, geben Sie für den ersten Tablespace den Befehl **begin backup** ein, führen die Sicherung auf Systemebene aus und geben vor dem Backup des zweiten Tablespace den Befehl **end backup** ein. Tunlichst zu vermeiden ist, die Sicherung stapelweise auszuführen, wobei auf einen Stapel von **begin backup**-Befehlen ein Stapel von Sicherungen folgt, die mit einem Stapel von **end backup-Befehlen beendet werden**. Grundsätzlich sollten Sie die Tablespaces nur so kurz wie möglich im HOT BACKUP-Modus halten und das zu Zeiten, wenn auf diesen Tablespace nur wenige Zugriffe erfolgen.

Das Kopieren der Dateien während der heißen Backups erfolgt mithilfe von Werkzeugen, die nicht zu Oracle gehören. Wir nehmen an, dass die Hardwarehersteller Sicherungswerkzeuge anbieten, die sehr viel besser sind als die portablen Funktionalitäten, die Oracle entwickeln könnte. Der DBA ist dafür verantwortlich, dass die Tablespaces nur zwischen den Befehlen **begin backup** und **end backup** kopiert werden.

Der Befehl **alter tablespace end backup** legt einen Redo-Datensatz mit der Checkpoint-SCN an, die am Anfang der Sicherung gesetzt wird. Die SCN findet sich auch im

Header der Datendateien wieder, die mit einem heißen Backup gesichert wurden. Dadurch erkennt Oracle, ob sämtliche Redos eingesetzt wurden, die im Zuge der Sicherungen generiert wurden. Anders formuliert muss der DBA bei einem Recovery mithilfe von heißen Backups *zumindest* die Redo einsetzen, die zwischen den Befehlen **begin backup** und **end backup** generiert wurde. Andernfalls wäre die Konsistenz der gesicherten Datendateien nicht sichergestellt. Unterbrechen Sie das Recovery vor Ausführung der Redos und versuchen, die Datenbank zu öffnen, erhalten Sie eine Fehlermeldung. Zudem wird nach Eingabe des Befehls **end backup** das Logging der Block-Images gestoppt, und die Checkpoints der Datendateien werden an den Checkpoint der Datenbank weitergeleitet.

Während eines heißen Backups können Sie die Datenbank weder mit der NORMAL- noch mit der IMMEDIATE-Option herunterfahren. Ein Tablespace, der sich im HOT BACKUP-Modus befindet, lässt sich auch nicht mit den Optionen NORMAL oder TEMPORARY offline setzen. Eine Meldung weist den DBA darauf hin, dass sich die Dateien im HOT BACKUP-Modus befinden, daher kann der DBA die Datenbank nicht herunterfahren oder den Tablespace offline setzen. Damit wird sichergestellt, dass ein *End Backup*-Marker generiert wird und der DBA tatsächlich den Befehl **end backup** eingibt. Kapitel 10 beschreibt anhand einer Fallstudie, was zu tun ist, wenn die Datenbank während der Ausführung eines heißen Backups zusammenbricht.

3.2.5 Kennzeichnung und Namenskonventionen

Obwohl der Recovery Manager die Informationen des Backup-Sets in seinen Metadaten verwalten kann, reicht dies bei einem Disaster-Recovery zur Unterstützung des DBAs nicht aus. Zusätzlich sind definierte Identifizierungsmethoden und Namenskonventionen zu befolgen und entsprechend einzusetzen.

Die Sicherungsbänder sollten durchgängig so etikettiert werden, dass sich bei einem Recovery aus der Kennzeichnung der Punkt der Wiederherstellbarkeit deutlich erkennen lässt. Die Bänder selbst sollte man mit eindeutigen Farben kennzeichnen, und zur leichteren Verwaltung mit vorgegebenen Namenskonventionen arbeiten. Beispielsweise sollte der erste Buchstabe des Namens ein *M*, *W* oder *T* sein, so lassen sich problemlos die monatlichen, wöchentlichen oder täglichen Sicherungen auseinander halten. Der zweite Buchstabe sollte den Backup-Typ identifizieren: V für vollständig, K für kumulativ, I für inkrementell usw. Die folgenden vier Buchstaben geben den Namen der Datenbank wieder. Die nächsten sechs Stellen sind für das Datum im Format *TT/MM/JJ* reserviert. Die letzten drei Buchstaben sind die Initialen desjenigen, der die Sicherung gezogen hat. Wenn beispielsweise der DBA „Johnny“ am 5. Juli 2000 die Datenbank **SALES.AQUILA** vollständig sicherte, sollte die Kennzeichnung wie folgt aussehen: TV-SAAQ-050700-JOH.

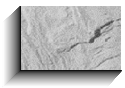
Entscheidet man sich bei der Kennzeichnung der Bänder für ein Farbsystem, sind die vorgegebenen Farbcodes für die täglichen, wöchentlichen und monatlichen Sicherungen und die unterschiedlichen Backup-Typen anzuwenden. Diese Praxis verhindert, dass es im Recovery-Fall zu Irritationen kommt, und hilft, die Zeit für die Wiederherstellung der Datenbank wesentlich zu verkürzen.

3.3 Backup-Befehle in verschiedenen Betriebssystemen

Dieser Abschnitt bietet eine Übersicht über die betriebssystemspezifischen Befehle, die zur Sicherung und Wiederherstellung der Datenbank im Zusammenhang mit Offline- oder Online-Backups benutzt werden. Bei manchen Betriebssystemen, wie IBM MVS, weichen grundlegende Konzepte (wie die Archivierung der Redo Logdateien) im Vergleich zu anderen Systemen ab. Solche Mechanismen werden in den folgenden Ausführungen beschrieben.

3.3.1 Backup/Restore in VMS-Umgebungen

Die Dateien in der VMS-Umgebung haben spezifische Erweiterungen: Bei den Datendateien ist es **dbs**, bei den Steuerdateien **con** und bei den archivierten und Online Redo Logs **rdo**. **backup** und **copy** sind die beiden Befehle, mit denen in VMS die Datenbankdateien gesichert und wiederhergestellt werden. Bei diesen Befehlen werden auch Wildcards unterstützt. Falls das Datenbanklayout OFA-konform ist, lassen sich alle Datendateien mit einem einzigen Befehl und der Angabe einer Wildcard sichern (d.h., Wildcards lassen sich für das Laufwerk, Teile des Pfadnamens und für den Dateinamen ohne die Erweiterung einsetzen). Für weitere Informationen zum OFA-konformen Datenbanklayout siehe Kapitel 2.



Hinweis
*Der **backup**-Befehl führt bei den Dateien diverse Prüfungen aus, was beim **copy**-Befehl nicht der Fall ist.*

Die folgenden Beispiele zeigen die Sicherung der Datendateien, Logdateien und Steuerdateien von Platte zu Platte:

```
$ backup/log/ignore=(interlock, nobackup) -
DiskA:[dir_path]*.dbs DiskB:[backup_dir_path]*.dbs

$ backup/log/ignore=interlock DiskA:[dir_path]*.rdo -
DiskB:[backup_dir_path]*.rdo

$ backup/log/ignore=interlock DiskA:[dir_path]*.con -
DiskB:[backup_dir_path]*.con
```

Zur Sicherung von Platte auf ein Bandgerät sind folgende Befehle einzusetzen:

```
$ mount/foreign tape_device:
$ backup/log/ignore=interlock DiskA:[dir-path]*.dbs -
tape_device:db_test.bck/sav
```

Der **copy**-Befehl führt keine Prüfungen durch oder ignoriert Sperren beim Kopieren von Platte auf ein Bandgerät. Der nachfolgende **copy**-Befehl dient zum Kopieren von Platte auf ein Bandmedium:

```
$ mount/over=id tape_device:
$ copy  DiskA:[backup_dir_path]*.dbs  tape_device:
```

Mit dem **backup**- oder **copy**-Befehl lassen sich die Datenbankdateien auch von Band auf Platte zurücksichern. Auch in diesem Fall kann man mit Wildcards arbeiten. Wurde das Bandlaufwerk mit **foreign** gemountet, ist der **backup**-Befehl zwingend einzusetzen. Den **copy**-Befehl setzt man ein, wenn das Bandlaufwerk hochgefahren ist. Das folgende Beispiel zeigt, wie man mit dem **backup**-Befehl Dateien von Band auf Platte zurückschreibt:

```
$ mount/foreign tape_device:
$ backup/rewind/list tape_device:*/sav
```

Um Dateien von einem Band abzurufen, geben Sie folgenden Befehl ein:

```
$ backup/rewind tape_device:db_test.bck/sav/select=(test.dbs,..) -
disk:[dir_path]file/new_version/owner=parent/log
```

Der folgende Befehl zeigt, wie man mit dem **copy**-Befehl Dateien vom Band auf ein Laufwerk schreibt:

```
$ mount  tape_device:  volume_label   ! ODER verwenden Sie den folgenden
                                        Befehl:
$ mount/over=id tape_device:
$ dir tape_device:                  ! Für eine Auflistung aller Dateien
$ copy tape_device:file disk:[dir_path]file  ! Um eine Datei vom Band
                                               abzurufen
```

3.4 Backup/Restore in UNIX-Umgebungen

In diesem Abschnitt zeigen wir einige Befehle, mit denen man in UNIX physische Backups der Datenbankdateien erstellt.

cpio

cpio ist ein Standard-Dienstprogramm auf UNIX System V-Plattformen. Bei UNIX BSD-Plattformen gehört es nicht zu den Standard-Funktionalitäten, es wird aber von vielen BSD-Anbietern angeboten. Mit diesem Befehl kopieren Sie Dateien und Verzeichnisstrukturen in und aus Archivdateien oder Verzeichnisstrukturen von einem Standort zu einem anderen. Mit den Befehlen **cat**, **ls** oder **find** erzeugen Sie die benötigten einspaltigen Listings mit den Pfadnamen. Mit **cpio** können Sie sowohl Dateien sichern, die Geräte beschreiben (spezielle Dateien), als auch Datendateien sichern.

cpio bietet drei Modi:

1. Den Copy-out-Modus, der eine Archivdatei erstellt:

   cpio -o [aABcLvV] [-C size] [-H hdr] [-O file] [-M msg]

2. Den Copy-in-Modus, mit dessen Hilfe sich eine zuvor gesicherte Datei abrufen lässt:

   cpio -i [6bBcdfkmrsStuvV] [-C size] [-E file] [-H hdr] [-O file]
   [-I file] [-M msg] [-R ID] [patterns]

3. Den Pass-Modus, der eine Kopie der Verzeichnisstruktur von der Quelle an ein neues Ziel übergibt. Dieser Modus arbeitet wie der Copy-out-Modus, kopiert jedoch die Dateien in einen neuen *Verzeichnisbaum*.

   cpio -p [adlmruvV] [-R ID] ***directory***

Eine Erklärung zu den verschiedenen **cpio**-Optionen finden Sie in den entsprechenden **man**-Pages.

Die folgenden drei Beispiele zeigen, wie man einen Verzeichnisbaum von einer Quelle an einen neuen Standort auf der Festplatte kopiert. Der erste Befehl sichert alle Dateien im gleichen Verzeichnispfad, der zweite Befehl alle Daten- und Steuerdateien und der dritte die Redo Logdateien.

```
$ ls /dsk*/ORACLE/prod/*.*  |  cpio -pdk new-dir

$ ls /dsk*/ORACLE/prod/*.dbf  |  cpio -pdk new-dir

$ ls /dsk*/ORACLE/prod/*.rdo  |  cpio -pdk new-dir
```

Um Dateien aus einer Archivdatei (**arch010194**) auf ein Laufwerk zurückzuschreiben, arbeiten Sie mit folgendem Befehl:

```
$ cpio -ic < /bck/arch010194
```

oder

```
$ cat /bck/arch010194 | cpio -ic
```

Bei *einigen* Plattformen kann man mit der Option **-r** (ohne **-p** verwendet) Dateien interaktiv umbenennen. Um Dateien aus einer Archivdatei auf Platte (**arch010194**) zurückzuschreiben und umzubenennen, verwenden Sie folgenden Befehl:

```
$ cpio -icr < /bck/arch010194
```

Beim Kopieren von Platte auf Band werden Sie von **cpio** beim Erreichen des Bandendes zur Eingabe des Gerätenamens aufgefordert. Danach können Sie entweder ein neues Band einlegen oder die Sicherungsroutine über EINGABE verlassen. Um Dateien aus einem Verzeichnis auf Band zu kopieren, verwenden Sie folgenden Befehl:

```
$ ls /dsk*/ORACLE/*.* | cpio -ocBv  >  tape_device
```

Um das aktuelle Verzeichnis und alle Unterverzeichnisse (Verzeichnisbaum) auf Band zu kopieren, arbeiten Sie mit folgendem Befehl:

```
$ find . -depth -print | cpio -ocBv  > tape_device
```

Bei der Wiederherstellung von Band identifizieren Sie zuerst die benötigten Dateien. Dazu geben Sie diesen Befehl ein:

```
$ cpio itBv < tape_device
```

Danach kopieren Sie mit folgendem Befehl die Datei oder den Verzeichnisbaum vom Band auf das Laufwerk:

```
$ cpio -icBv file  < tape_device
$ cpio -icdBv file < tape_device
```

tar

Der **tar**-Befehl ist bei System V und BSD UNIX ein Standard-Dienstprogramm. Mit diesem Befehl werden Archivdateien vom Laufwerk auf Band kopiert oder archivierte Dateien vom Band abgeholt. Zudem lassen sich mit diesem Befehl Verzeichnisstrukturen von einem Verzeichnis in ein neues Verzeichnis kopieren. Einige BSD-Systeme unterstützen vor dem **tar**-Befehl keinen Bindestrich. Ist das Gerät ein Bindestrich (-), schreibt **tar** an den Standard-Output oder liest vom Standard-Input. Eine ausführliche Beschreibung aller Optionen des **tar**-Befehls finden Sie in den **man**-Pages in UNIX.

Nachfolgend wird mit dem **tar**-Befehl eine Verzeichnisstruktur in ein neues Verzeichnis gesichert:

```
$ tar cf - . | ( cd to_dir; tar xf - )
```

In diesem Beispiel wird mit dem **tar**-Befehl am Standard-Output eine **tar**-Datei anlegt. Die Ausgabe wird über eine Pipe an eine Subshell weitergeleitet, die das Verzeichnis (**cd**) wechselt und die Dateien ins gewünschte Verzeichnis kopiert. Das zweite **tar** extrahiert die Dateien in eine hierarchische Struktur.

Das folgende Beispiel zeigt, wie man Dateien aus einem Verzeichnis auf Band sichert:

```
$ tar -cvf  /dev/rmt0h  /dsk*/ORACLE/*.*
```

wobei **rmt0h** das Bandgerät ist.

Um beispielsweise die Inhalte eines Archivbands anzeigen zu lassen und Dateien vom Band zurückzuschreiben, arbeiten Sie mit folgenden Befehlen:

```
$ tar -tvf tape_device

$ tar -xvf tape_device
```

tar und cpio – ein Vergleich

Die Vorteile des **tar**-Befehls sind wie folgt:

- Die Syntax ist relativ einfach.
- Archivdateien lassen sich durch verschiedene Versionen ersetzen und neue Dateien können am Ende eines Archivs angehängt werden, ohne dass das Band komplett neu beschrieben werden muss.

Die Vorteile des **cpio**-Befehls sind wie folgt:

- Er kann Dateien sowohl auf Geräte (Sonderdateien) als auch in Datendateien sichern.
- Er schreibt die Daten in ein Stream-Format, was beim Anlegen einer Bandsicherung Zeit und Platz spart. **cpio** ist üblicherweise schneller als **tar** und sichert die Daten effizienter.
- Treten Probleme auf, versucht **cpio**, das Band mehrmals zu lesen.
- **cpio** kann einen fehlerhaften Bandbereich überspringen.

cp

cp ist ein Befehl unter System V und BSD UNIX, mit dem sich Dateien oder Verzeichnisstrukturen von einem Standort zum anderen (auf Platte) kopieren lassen. Die Syntax sieht wie folgt aus:

cp [-ip] *source_file destination_file*
cp [-ipr] *source_file_list destination_directory*
cp -r [-ip] *source_directory destination_directory*

wobei die Option -i eine interaktive Bestätigung anfordert. Soll die Kopie eine bestehende Datei überschreiben, erhalten Sie eine Eingabeaufforderung. Bestätigen Sie diese mit *yes*, wird die Kopie ausgeführt. Die Option -p wird eingesetzt, um die Charakteristika der Quelldatei zu erhalten. Die Inhalte, Änderungszeiten und Berechtigungen werden zur Zieldatei kopiert. Die Option -r kopiert alle Quellverzeichnisse rekursiv. Wird ein Verzeichnis als Quelldatei angegeben, werden alle dazugehörigen Dateien und Unterverzeichnisse kopiert. Das Ziel muss ein Verzeichnis sein.

Die folgenden Beispiele zeigen, wie man eine Datei und eine Verzeichnisstruktur an einen anderen Standort (auf Platte) kopiert:

```
$ cp datafile  /bck/datafile

$ cp -r data_file_dir    bck_dir
```

volcopy

Das ist ein System V-Befehl, der eine literale Kopie des Dateisystems anlegt, wobei die Blockgrößen des Geräts verwendet werden. Die Syntax ist

$ **volcopy** *[option] fsname srcdevice volname1 destdevice volname2*

wobei *option* entweder -a oder -s sein kann. Sollten Sie mit der Option -a arbeiten, wird eine Verifizierungssequenz aktiviert, die eine positive Operator-Antwort erfordert und die 10-sekündige Verzögerung vor Ausführung der Kopie ersetzt. -s ist die Standardoption, die beim Auftreten einer falschen Verifizierungsssequenz die Operation abbricht. Sollten Länge oder Schreibdichte weder im Befehl angegeben noch auf dem Input-Tape-Label protokolliert sein, werden diese Werte vom Programm angefordert. **fsname** repräsentiert den Namen (beispielsweise root), mit dem das zu kopierende Dateisystem gemountet wurde. **srcdevice** und **volname 1** repräsentieren den Gerätenamen und den Namen des physischen Laufwerks, aus der die Kopie des Dateisystems extrahiert wird. **destdevice** und **volname2** repräsentieren die Namen des Zielgeräts und des Laufwerks.

dump and restor

dump und **restor** sind standardmäßige BSD UNIX-Befehle. Der **dump**-Befehl kopiert alle Dateien, die nach einem bestimmten Datum geändert wurden, von einem angegebenen Dateisystem an eine Datei, eine Pipe, ein Band oder Laufwerke. Dieses Dienstprogramm unterstützt das EOF-Handling, das den Einsatz mehrerer Medien erlaubt. Ist ein Laufwerk gefüllt, werden Sie nach dem Namen des folgenden gefragt. Die Syntax ist

$ **/etc/dump** [*key* [*argument ...*] *file_system*]

wobei *key* das Datum und andere Optionen für Dump angibt; einige Schlüssel benötigen ein *argument*. Die verschiedenen Optionen für *key* sind wie folgt:

[0123456789aBdFfnsSuWw]

wobei 0 bis 9 das Dump-Level angibt. Level 0 bedeutet die Sicherung des gesamten Systems. Level 1 sichert nur die Dateien, die seit dem letzten Level 0-Dump geändert wurden. Level 2 sichert nur die Dateien, die seit dem letzten Level 0- oder Level 1-Dump geändert wurden, usw. Eine ausführliche Beschreibung aller Optionen finden Sie in den **man**-Pages von UNIX (beispielsweise **man** dump).

Das folgende Beispiel sichert das gesamte Dateisystem (**/bck/db_files**) an das Gerät (**/dev/rra2a**) mit einer Blockgröße von 400 Blöcken, wobei jeder Block 1.024 Bytes groß ist.

```
$ dump 0Bf 400 /dev/rra2a  /bck/db_files
```

Das folgende Beispiel sichert das gesamte Dateisystem (**/bck/db_files**) auf ein 6.250-bpi-Band auf einem TU78-Bandlaufwerk:

```
$ dump 0undf /dev/rmt0h /bck/db_files
```

Der **restor**-Befehl führt unter BSD UNIX eine inkrementelle Rücksicherung eines Dateisystems durch. Dieser Befehl erhält Dateien aus einer Datei, einem Band oder einem Laufwerk, die bei einem *früheren* Dump gesichert wurden. Sie können entweder alle Elemente oder Teile eines zerstörten Dateisystems oder einzelne Dateien zurücksichern, die von einem Benutzer überschrieben wurden. Ein Dateisystem mit Spezialdateien sollte nur von einem *Superuser* zurückgesichert werden. Zur Rücksicherung des Root-Dateisystems muss sich das System im Stand-alone-Modus befinden, wobei der **restor**-Befehl keinerlei Argumente akzeptiert; das Argument -r ist implizit. Die Syntax ist wie folgt:

$ **restor** *key [argument] [file-system]*

Der folgende Befehl legt auf dem Laufwerk ein leeres Dateisystem an, wobei das vorhandene zerstört wird, und sichert danach auf das gleiche Laufwerk einen vollständigen Dump. Das Laufwerk kann nicht das Root-Device enthalten, da nach dem **mkfs**-Befehl das Root-Dateisystem nicht mehr vorhanden wäre.

```
$ restor r device      (Hier wird das Standard-Gerät device angenommen)
$ restor r /dev/da0    (Das Dateisystem wird auf Laufwerk da0
                        zurückgeschrieben)
```

Um eine Datei aus einem zuvor gezogenen Dump zurückzusichern, verwenden Sie folgenden Befehl:

```
$ restor x file
```

wobei *file* die Inode-Nummer der Datei ist, die aus dem Dump extrahiert werden soll.

backup und restore

backup und **restore** sind standardmäßige Befehle unter UNIX System V. Das Dienstprogramm **backup** ist eine Schnittstelle für **cpio**. Sicherungen, die mit **backup** gezogen wurden, sollten mit dem **restore**-Bfehl zurückgesichert werden. Die Syntax ist

```
$ backup  [-t] [-p|-c|-f files |-u "user1 [user2]" ]  -d device
$ backup -h
```

wobei -h eine Backup-History anlegt und den Benutzer darüber informiert, wann die letzten inkrementellen/partiellen Backups ausgeführt wurden. Die Option -c generiert einen vollständigen Backup unter Einbeziehung aller Dateien, die seit der Installation des Systems geändert wurden. Die Option -p führt einen inkrementellen/partiellen Backup aus und berücksichtigt nur Dateien, die seit der letzten Sicherung geändert wurden. Die Option -f sichert die angegebenen Dateien. Die Dateinamen können Zeichen enthalten, die erweitert werden – z. B. den Stern (*) oder den Punkt (.). Beachten Sie, dass das Argument in Klammern zu setzen ist. Die Option -u sichert alle Dateien im Home-Verzeichnis des Benutzers. Es ist zumindest ein Benutzer anzugeben.

Wird mehr als ein Benutzer angegeben, ist das Argument in Klammern zu setzen. Das Argument *all* sichert die Home-Verzeichnisse aller Benutzer. Die Option -t ist bei Angabe eines Bandgerätes zusammen mit der Option -d anzugeben.

Der **restore**-Befehl führt die inkrementelle Rücksicherung eines Dateisystems aus, das zuvor mit dem System V-Dienstprogramm **backup** gesichert wurde. Dieses Dienstprogramm dient als Front-End zu **cpio**. Die Syntax ist

```
$ restore [-c] [-i] [-o] [-t] [-d device] [pattern [pattern] ...]
```

wobei die Option -c eine vollständige Wiederherstellung ausführt. Alle Dateien auf dem Band werden wiederhergestellt. Die Option -i holt die Indexdatei des Mediums ab. Die Option -o überschreibt vorhandene Dateien. Ist die wiederherzustellende Datei bereits vorhanden, wird sie erst nach Angabe dieser Option zurückgesichert. Die Option -t weist darauf hin, dass das Bandgerät zu benutzen ist. Die Option muss bei einer Wiederherstellung vom Band zusammen mit der Option -d eingesetzt werden. Die Option **-d** device definiert das einzusetzende Gerät.

fbackup und frestore

Die Befehle **fbackup** und **frestore** werden auf HP-UX System V eingesetzt. Mit dem Befehl **fbackup** werden Dateien selektiv auf ein Ausgabegerät transferiert. **fbackup** kombiniert die Leistungsmerkmale von **dump** und **ftio** und bietet damit einen sehr schnellen und flexiblen Mechanismus zur Sicherung von Dateisystemen. Die Syntax ist wie folgt:

$ **/etc/fbackup** -f *device* [-f *device..*] [-0-9] [-uvyAH] [-i *path*]
[-e *path*][-g *graph_file*] [-I *path*] [-V *path*] [-c *config*]
$ **/etc/fbackup** -f *device* [-f *device..*] [-R *restart_file*] [-uvyAH][-I *path*] [-V *path*]
[-c *config*]

Der Rückgabewerte ist 0 bei normaler Beendigung, 1, wenn eine Unterbrechung auftritt, der Status aber für mögliche Neustarts gesichert werden darf, und 2, wenn eine Fehlerbedingung die Beendigung der Sitzung verhindert. Das Ausgabegerät kann eine Datei, der Standard-Output, ein Raw-Bandgerät, ein Band im DDS-Format oder eine wiederbeschreibbare optische Disk sein.

Die Auswahl der zu sichernden Dateien erfolgt über die explizite Angabe von Dateibäumen, die entweder in die **fbackup**-Sitzung eingebunden oder ausgeschlossen werden. Der Benutzer kann sich mit den Optionen -i (include) oder -e (exclude) einen beliebigen Graph aus Dateien konstruieren. Mit der Option -g bindet man eine oder mehrere Graph-Dateien ein. Bei regelmäßigen Backups bietet die Option -g eine einfachere Schnittstelle zur Steuerung des Backup-Graphs. **fbackup** selektiert die Dateien in diesem Graph und versucht, sie an das Ausgabegeräte zu transferieren. Die Selektivität hängt von dem Modus ab, in dem **fbackup** eingesetzt wird (beispielsweise vollständige oder inkrementelle Backups).

Bei der Ausführung von vollständigen Backups werden alle Dateien im Graph selektiert. Bei inkrementellen Sicherungen werden im Graph nur die Dateien ausgewählt, die seit dem letzten Backup des Graphs verändert wurden. Wird **fbackup** für inkrementelle Backups eingesetzt, ist eine Datenbank der vergangenen Sicherungen vorzuhalten. Standardmäßig hält **fbackup** die Daten in der Textdatei **/usr/adm/fbackupfiles/dates** vor. Das Verzeichnis **/usr/adm/fbackupfiles** ist vor Ausführung des ersten inkrementellen Backups anzulegen. Mit der Option -d lässt sich eine andere Datenbank angeben. Die Einträge für jede Sitzung werden jeweils in zwei Zeilen dokumentiert. Die erste Zeile enthält jeweils die Dateinamen des Graphs, das Backup-Level, die Start- und die Beendigungszeit. Die zweite Zeile enthält die gleiche Information, aber im strftime (3C)-Format nutzt **fbackup** diese Zeile nicht, sondern sie dient nur der Lesbarkeit. Die Dateinamen für die Graphen werden zeichenweise verglichen um zu ermitteln, wann die letzte Sitzung für diesen Graph stattfand.

Das folgende Beispiel zeigt, wie man eine Datei auf Band sichert, die Verzeichnisse mit Dateien enthält. Danach führen wir alle Dateien auf, die in den Backup einbezogen oder ausgeschlossen werden.

```
$ /etc/fbackup -f tape_device - g graph_file -u -0
$ /etc/fbackup -0I /usr -e /usr/lib -f /dev/rmt0h
$ cd /usr/adm/fbackupfiles
$ /etc/fbackup -0uc config -g graphs -I indices -f /dev/rmt0h
# graphs file
i         /dsk1
i         /dsk2
```

```
i          /dsk3/oracle
i          /dsk4/usr
e          /dsk1/usr/class
e          /dsk2/usr/test
```

Der **frestore**-Befehl liest Medien, die von **fbackup** beschrieben wurden. Die Syntax ist wie folgt:

```
$ /etc/frecover -r [-hmosvyAFNOX] [-c config] [-f device] [-S skip]
$ /etc/frecover -R path [-f device]
$ /etc/frecover -x [-hmosvyAFNOX] [-c config] [-e path] [-f device]
[-g graph] [-i path] [-S skip]
```

In den **man**-Pages finden Sie die Beschreibung für die verschiedenen Optionen.

dd

Dieser Befehl kopiert die angegebene Eingabedatei, möglicherweise mit Konversionen, an die angegebene Ausgabedatei. Er kann die Eingabe von einer Datei oder vom Standard-Input lesen. Der **dd**-Befehl ist besonders hilfreich, weil er die Sicherung mit Raw-Device erlaubt, was mit den Befehlen **tar** und **cpio** nicht möglich ist. Mit **dd** können Sie bei Ausführung eines physischen I/Os die Blockgrößen für Input und Output angeben.

Mit dem **dd**-Befehl lassen sich Daten zwischen Geräten mit unterschiedlichen Blockgrößen kopieren, was mit **cpio** und **tar** nicht möglich ist; deshalb lässt sich **dd** als Front-End-Befehl einsetzen, der die Daten vom Band extrahiert und die Blockgrößen so setzt, dass **tar** und **cpio** damit arbeiten können. Die Syntax ist wie folgt:

$ **dd** [*option* = *value* ...]

Nachfolgend sind einige der wichtigsten Optionen beschrieben:

- bs=*n* Setzt die Input- und Output-Blockgrößen auf *n* Bytes.
- count=*n* Erlaubt nur das Kopieren eines Inputs von *n* Blöcken.
- ibs=*n* Die Input-Blockgröße ist auf *n* Bytes gesetzt. Wird die Option ibs nicht genutzt, ist die Blockgröße 512 Bytes. Bei einigen Systemen kann es beim Kopieren von Band auf Platte zu Datenfehlern kommen, wenn das ibs größer als 1024 ist. Als Workaround sollten Sie die Option bs einsetzen.
- if=*file* Definiert die Eingabedatei *file*. Wird die Option nicht verwendet, kommt der Standard-Input zum Einsatz.
- obs=*n* Die Blockgröße des Outputs ist auf *n* Bytes gesetzt. Wird obs nicht verwendet, ist die Blockgröße 512 Bytes.
- of=*file* Definiert die Ausgabedatei *file*. Wird die Option nicht verwendet, kommt der Standard-Output zum Einsatz.

- seek=*n* Überspringt die ersten *n* Blöcke der Ausgabedatei, bevor mit dem Schreiben der Daten begonnen wird.
- skip=*n* Überspringt die ersten *n* Blöcke, bevor mit dem Kopieren der Eingabedatei begonnen wird.

Für die Sicherung von einem Raw-Device auf ein Raw-Bandgerät stehen Ihnen zwei Optionen zur Verfügung:

1. Sie können die Daten des Raw-Device mit dem Befehl Data Dictionary in eine reguläre UNIX-Datei kopieren und dann mit den normalen UNIX-Befehlen wie **cpio** oder **tar** arbeiten.
2. Kopieren Sie die Raw-Device-Daten direkt an das Raw-Bandgerät. Der **dd**-Befehl unterstützt nicht mehrere Geräte, sollte also die Partition größer als ein Band sein, ist der Befehl mehrfach einzugeben. Prüfen Sie, ob bei **dd** besondere Blockgrößen zu berücksichtigen sind. Die folgenden Schritte verdeutlichen die Prozedur:
   - ❏ Mounten Sie das erste Bandgerät und sichern Sie das Raw-Device mit folgendem Befehl auf Band:

     ```
     # dd if=raw_device of=tape_device bs=block_size count=number
     ```

   - ❏ Sollten Sie weitere Bänder benötigen, geben Sie nach Aufforderung das Folgende ein. *skip* wird für jedes weitere Band durch den *count*-Wert erhöht.

     ```
     # dd if=raw_device of=tape_device skip=number bs=block_size
     count=number
     ```

Um Daten von einem Raw-Device auf eine Raw-Partition zurückzuschreiben, sind folgende Schritte auszuführen:

1. Mounten Sie das Bandgerät und geben Sie Folgendes ein:

   ```
   # dd if=tape_device of=filename bs=block_size count=number
   ```

2. Mounten Sie, falls benötigt, weitere Bandgeräte und erhöhen Sie **seek** oder **oseek** um **count**:

   ```
   # dd if=tape_device of=raw_device [seek|oseek]=number bs=block_size
   count=number
   ```

crontab – Befehl für den automatisierten UNIX-Scheduler

Physische Backups oder Exports lassen sich mit **crontab** einplanen, einem Befehl für einen automatisierten Scheduler. Kapitel 5 zeigt ein Beispiel für ein UNIX-Skript, mit dem sich heiße und kalte Backups ausführen lassen. Dieses Skript wird über **crontab** gestartet. In diesem Abschnitt beschreiben wir die Funktionsweise des Schedulers.

Der **crontab**-Befehl beschreibt eine Datei, die aus Befehlen besteht, die in regelmäßigen Intervallen ausgeführt werden. Das **cron**-Programm liest, interpretiert und führt

die **crontab**-Datei aus. Die Befehle werden üblicherweise von der Bourne Shell (sh) ausgeführt. Die Syntax ist wie folgt:

crontab [*file*]
crontab -e [*username*]
crontab -l [*username*]
crontab -r [*username*]

wobei *file* die **crontab**-Datei ist. Über die Option -e kann man die **crontab**-Datei mit dem Editor bearbeiten, die über die EDITOR-Variable definiert wurde. Die Option -r löscht die aktuelle **crontab**-Datei. Ist *username* angegeben, löscht man damit die **crontab**-Datei des entsprechenden Benutzers. Nur der *Superuser* kann die **crontab**-Dateien anderer Benutzer löschen. Die Option -l listet den Inhalt der aktuellen **crontab**-Datei auf. Das Argument *file* ist der Name der Datei, die Ihre **crontab**-Datei sein soll. Die Datei wird an eine Datei namens *username* kopiert, die sich im **crontab**-Verzeichnis des Systems befindet.

Der **crontab**-Befehl liest eine Datei oder den Standard-Input in ein Verzeichnis, das die **crontab**-Dateien aller Benutzer enthält: **/usr/spool/cron/crontabs/username**. Mit **crontab** können Sie Ihre **crontab**-Datei anzeigen oder löschen lassen. Im **crontab**-Verzeichnis selbst können Sie nicht auf die **crontab**-Dateien anderer Benutzer zugreifen. Falls Sie den Standard-Output und Standard-Error eines Befehls, den Sie aus der **crontab**-Datei ausführen lassen, nicht umlenken, erhalten Sie die Ausgabe per Mail. Die **crontab**-Datei besteht aus Zeilen mit jeweils sechs Feldern, die durch Leerstellen (Leerzeichen oder Tabs) getrennt sind. Die ersten fünf Felder sind Integer, die den Zeitpunkt festlegen, zu dem der Befehl gestartet wird. Im sechsten Feld befindet sich der Befehl, der von **cron** ausgeführt wird. Tabelle 3-2 zeigt die ersten fünf Felder einer Zeile in der **crontab**-Datei.

Tabelle 3-2: Beschreibung der ersten fünf Felder einer Zeile in der crontab-Datei.

Feld	Bereich	Bedeutung
1	0 bis 59	Minuten
2	0 bis 23	Stunden (Mitternacht ist 0)
3	1 bis 31	Tag im Monat
4	1 bis 12	Monat im Jahr
5	0 bis 6	Wochentag (Sonntag ist 0; Samstag ist 6)

Jedes Feld kann eines der folgenden Elemente enthalten:

- Einen Integer
- Einen Bereich

- Eine Liste (mit Integern oder Bereichen)
- Einen Stern (*) (zeigt alle gültigen Werte, d.h. alle gültigen Zeiten an)

Die Felder „Wochentag“ und „Tag im Monat“ werden, falls beide angegeben sind, jeweils separat interpretiert. Soll nur eine der Angaben gemacht werden, ist im anderen Feld ein Stern (*) einzutragen. Einige Beispiele dazu siehe in Tabelle 3-3.

Tabelle 3-3: Beispiele für crontab-Einträge.

10 0 * * 3	Führt den Befehl nur am Mittwoch um 12:10 aus.
0 6 1,9 * 1	Führt den Befehl um 6 Uhr am ersten und neunten jedes Monats und jeden Montag aus.
0,30 7-20 * * *	Führt den Befehl täglich alle 30 Minuten von 7 bis 20 Uhr aus.

Das sechste Feld enthält den Befehl, der von **cron** zur angegebenen Zeit auszuführen ist. Der Befehls-String wird von einem Newline oder einem Prozentzeichen (%) begrenzt. Der gesamte Text, der dem Prozentzeichen folgt, wird dem Befehl als Standard-Input übergeben. Dem Prozentzeichen selbst wird ein Backslash vorangestellt. Zeilen, die mit einem #-Zeichen beginnen, sind Kommentare. Um mit dem **crontab**-Befehl arbeiten zu können, benötigen Sie die entsprechenden Berechtigungen. Der Systemverwalter kann den **crontab**-Befehl entweder allen, nur bestimmten oder für keine Benutzer verfügbar machen. Die Zugriffsberechtigungen werden über die Dateien **/usr/sbin/cron.d/cron.allow** und **/usr/sbin/cron.d/cron.deny** kontrolliert. Sollte die Datei **cron.allow** vorhanden, aber leer sein, können alle Benutzer mit dem **crontab**-Befehl arbeiten. Ist keine der beiden Dateien vorhanden, kann nur der *Superuser* auf diesen Befehl zugreifen.

Andere dazugehörige Dateien sind:

/usr/sbin/cron.d	Das Hauptverzeichnis für den cron-Prozess.
/usr/sbin/cron.d/log	Abrechnungsinformationen für die cron-Verarbeitung.
/usr/sbin/cron.d/crontab.allow	Eine Datei mit einer Liste von Benutzern, die mit crontab arbeiten dürfen.
/usr/sbin/cron.d/crontab.deny	Eine Datei mit einer Liste von Benutzern, die nicht mit crontab arbeiten dürfen.
/usr/spool/cron/crontabs	Der Standort des crontab-Textes, der auszuführen ist.

3.4.1 Backup/Restore in einer IBM MVS-Umgebung

Backup- und Restore-Prozeduren für Oracle sind unter MVS und anderen Systemen identisch. Extern gibt es allerdings einige systembezogene Unterschiede. Im Wesentlichen sind die Oracle-Dateien unter MVS nichts anderes als VSAM-Dateien, und der ARCH-Prozess übergibt zur Archivierung der Online Redo Logdateien einen Batch-Job.

Sicherung von Datendateien

Oracle-Dateien werden als physische Images mit Dienstprogrammen wie IBMs DF/DSS oder FDR gesichert. Sie können die Datenbankdateien mit IDCAMS EXPORT und dem Parameter CIMODE sichern. IDCAMS REPRO funktioniert nicht, da Oracle auf dem CI-Level arbeitet und keine VSAM-Datensätze benutzt. Dateien, die zu einem Tablespace gehören, sollten über einen Batch-Job als Einheit gesichert werden. Nachfolgend finden Sie ein Beispiel für eine Sicherung in Form eines physischen Images:

```
//BACKUP   JOB (0000,O7),'ORACLE IMAGE COPY',CLASS=A
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD  SYSOUT=*
//DUMPOUT  DD  DSN=ORACLE.ORA1V.IMAGE.COPY,DISP=(NEW,CATLG,DELETE),
//             UNIT=TAPE,VOL=(,99,SER=(BKUP01,BKUP02,BKUP03,BKUP04)),
//             LABEL=(1,SL,EXPDT=98000)
//SYSIN    DD  *
  DUMP DATASET(INCLUDE(ORACLE.ORA1V.**))
       OUTDD(DUMPOUT)
/*
//
```

Archivierung der Redo Logdateien

Die Archivierung der Redo Logdateien verläuft im Unterschied zu UNIX und anderen Betriebssystemen etwas anders. Wenn Sie die Datenbank im ARCHIVELOG-Modus laufen lassen, setzt Oracle für MVS einen Timer mit dem Intervall MAXWAIT und übergibt zur Archivierung der Online Redo Logdateien einen Batch-Job. Sollte der Archivierungs-Job nach Ablauf von MAXWAIT noch nicht beendet sein, übergibt Oracle einen weiteren Batch-Job zur Sicherung des gleichen Logs. Gelegentlich kann es vorkommen, dass sich die automatische Anforderung zum Mounten des Bandgeräts verzögert und ein zweiter Job gestartet wird, bevor das Band gemountet ist. Der zweite Job erkennt, dass das Redo Log bereits verarbeitet wurde und beendet sich. Oracle für MVS nutzt zur Steuerung der Archivierung den INIT.ORA-Parameter ACS (Archive Control String). Nachfolgend ein Beispiel:

```
ACS="TYPE=SUBMIT,INCJCL=/DD/ARCH,ODSN1=/DD/O1,ODSN2=/DD/O2,MAXWAIT=30"
```

Damit erhält der Archiver folgende Instruktionen:

1. Setze einen Timer für 30 Minuten.
2. Lies die JCL aus der ARCH Data Dictionary-Anweisung.
3. Ersetze die Schlüsselwörter durch die passenden Werte.
4. Übergib einen Batch-Job zur Archivierung des zuletzt gefüllten Online Redo Logs an die Anweisungen O1 DD und O2 DD. (Jawohl, Oracle für MVS besitzt seit Jahren eine duale Archivierung!)
5. Nach Ablauf des Timers prüft Oracle, ob das Online Redo Log archiviert wurde. Falls nicht, wird der Prozess wiederholt, bis die Archivierung abgeschlossen ist.

Nachfolgend ein Beispiel für die Archivierung mit JCL:

```
//ARCHIVE  JOB (0000,O7),'ORACLE ARCHIVE',CLASS=A
//STEP1    EXEC PGM=ARCHIVE,PARM='++/DD/SYSPARM'
//STEPLIB  DD  DISP=SHR,DSN=ORACLE.ORA1.AUTHLOAD
//SYSERR   DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//SYSIN    DD  DUMMY
//O1       DD  DISP=(NEW,CATLG,DELETE),
//             DSN=ORACLE.AL%LOGSEQ%.LOG,
//UNIT=SYSDA,DCB=(RECFM=FB,LRECL=4096,BLKSIZE=24576),
//             SPACE=(4096,(200,350),RLSE),VOL=SER=ORA001
//ORA$MMIO DD  DUMMY
//SYSPARM  DD  *
%LOGSEQ%
%LOGNAME%
%ODSN1%
%ODSN2%
/*
//
```

In dem oben aufgeführten JCL-Code ist

- %LOGNAME% das zu archivierende Online Redo Log.
- %LOGSEQ% die Redo Logsequenznummer.
- %ODSN1% der ODSN1-String aus dem ACS-PArameter.
- %ODSN2% der ODSN2-String aus dem ACS-Parameter.

Sicherung von Steuerdateien

Steuerdateien sollten im Rahmen der normalen kalten Backups genau wie der Rest der Datenbank und die Online Redo Logs gesichert werden. Daneben können Sie die Steuerdatei auch mit dem Befehl **alter database backup controlfile to '*file*'** sichern. Sollten Sie sich für die Sicherung mit diesem Befehl entscheiden, ist die Datei mit der Steuerdatei für den Backup eine VSAM-Datei, die mit dem Dienstprogramm IDCAMS vor der Ausführung des **alter database**-Befehls anzulegen ist. Falls Sie zum Anlegen der

VSAM-Datei einen Batch-Job übergeben möchten, sollten Sie die Steuerdatei im zweiten Schritt des gleichen Jobs sichern. Der folgende JCL-Code legt eine VSAM-Datei für die Aufnahme einer Steuerdatei an und gibt danach zur Sicherung der Steuerdatei den Befehl **alter database** aus.

```
//ORACLE1  JOB (0000,ORA),'ORACLE',CLASS=A,
//         MSGCLASS=X,PRTY=15,MSGLEVEL=(1,1),
//         REGION=4096K
//*-------------------------------------------------------*
//*        CREATE A VSAM FILE                             *
//*        TO HOLD A CONTROL FILE BACK                    *
//*-------------------------------------------------------*
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
   DELETE (ORACLE.ORA1V.C020295)                    -
          CLUSTER PURGE
   DEFINE CLUSTER                                   -
     (                                              -
      NAME(ORACLE.ORA1V.C020295)                    -
          VOLUMES(ORA001)                           -
          CONTROLINTERVALSIZE(4096)                 -
          RECORDS(400)                              -
          RECORDSIZE(4089 4089)                     -
          NONSPANNED                                -
          UNIQUE                                    -
          NONINDEXED                                -
          SPEED                                     -
          SHR(3 3)                                  -
     )                                              -
     DATA                                           -
     (                                              -
     NAME(ORACLE.ORA1V.C020295.DATA)                -
     )
//*-------------------------------------------------------*
//*        RUN SVRMGR TO BACKUP THE CONTROL FILE          *

//*        TO A VSAM FILE AND TO TRACE                    *
//*-------------------------------------------------------*
//STEP2    EXEC PGM=SVRMGR
//STEPLIB  DD DSN=ORACLE.ORA1.CMDLOAD,DISP=SHR
//SYSMDUMP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*,DCB=(LRECL=132,BLKSIZE=1320,RECFM=VB)
//SYSERR   DD SYSOUT=*,DCB=(LRECL=132,BLKSIZE=1320,RECFM=VB)
//ORAPRINT DD SYSOUT=*
//DBAINIT  DD DUMMY
//ORA@ORA1 DD DUMMY
//SYSIN    DD *
```

```
CONNECT SYSTEM/MANAGER
ALTER DATABASE BACKUP CONTROLFILE TO '/DSN/ORACLE.ORA1V.C020295';
ALTER DATABASE BACKUP CONTROLFILE TO TRACE;
/*
//
```

Beachten Sie, dass der letzte **alter database**-Befehl eine Trace-Datei mit dem SQL-Skript zur Einrichtung einer neuen Steuerdatei anlegt. Unter MVS werden die Trace-Dateien mit dem Parameter MPM TRACEDS als Modell angelegt. Die INIT.ORA-Parameter USER_DUMP_DEST und BACKGROUND_DUMP_DEST werden nicht verwendet. Nachfolgend finden Sie ein Beispiel für den Parameter TRACEDS:

```
TRACEDS="ORA1.PROD1.TRACE** UNIT=SYSDA"
```

3.4.2 Backup/Restore unter Windows NT

Windows NT 4.x bietet ein Werkzeug namens **backup**, mit dem Benutzer Dateien von einem Laufwerk auf Disketten oder Bänder sichern können. Der **backup**-Befehl steht in der Eingabeaufforderung oder über eine grafische Schnittstelle zur Verfügung. Zur Nutzung der letztgenannten Variante wählen Sie Start | Programme | Administrative Tools | Backup. Um das Programm aus dem System heraus aufzurufen, geben Sie in der Eingabeaufforderung **backup** ein. Bei einem kalten Backup ist darauf zu achten, dass die Datenbank ordentlich heruntergefahren wurde, oder sicherzustellen, dass der Tablespace bei einem heißen Backup offline ist. Erst dann können Sie die Dateien mit dem Dienstprogramm **backup** sichern.

Mit dem **at**-Befehl lassen sich unter Windows NT Tasks für einen bestimmten Zeitpunkt einplanen. Sie können eine Batch-Datei mit allen Befehlen für einen Backup einrichten und die Ausführung dieser Datei mit dem **at**-Befehl vorgeben. Um eine Batch-Datei namens **orabak.bat** um 2:00 Uhr einzuplanen, geben Sie Folgendes ein:

C:> **at 02:00 c:\oracle\orabak.bat**

Ein Datenbankmanagement-Werkzeug, das man unter Windows NT ab Oracle7, Release 7.1 bis Oracle8 (allerdings NICHT in Oracle8*i* für NT) einsetzen kann, ist der Oracle Backup Manager für Windows NT. Zum Aufruf dieses Programms klicken Sie auf Start | Programme | Oracle für Windows NT. Um mit dem Werkzeug arbeiten zu können, benötigen Sie das Passwort für den Benutzer *internal* (das Standard-Passwort ist *oracle*). Der Backup Manager läuft in zwei Modi, je nachdem, ob Sie die Datenbank im ARCHIVELOG- oder NOARCHIVELOG-Modus betreiben. Abbildung 3-1 und Tabelle 3-4 bieten einige Fakten und Ansichten zum Backup Manager, wenn die Datenbank im ARCHIVELOG-Modus läuft.

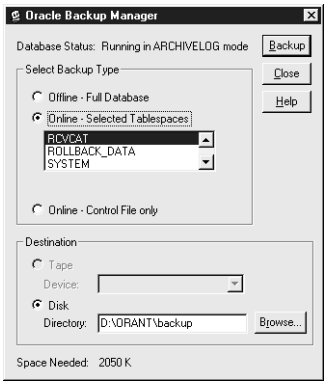


Abbildung 3-1: Das Dialogfeld des Backup Managers, wenn die Datenbank im ARCHIVELOG-Modus läuft.

Tabelle 3-4: Die Elemente des Dialogfelds im Backup Manager, wenn die Datenbank im ARCHIVELOG-Modus läuft.

Dialogelement	Beschreibung
Database Status	Zeigt den Status der Datenbank.
Offline - Full Database	Falls ausgewählt, führt diese Option einen Offline-Backup aus. Ist die Datenbank nicht geöffnet, werden nur die Daten-, Log- und Steuerdateien gesichert. Ist die Datenbank geöffnet, fährt sie die Datenbank herunter, führt den Backup aus und startet die Datenbank neu.
Online - Selected Tablespace	Falls ausgewählt, können Sie online eine partielle Datenbanksicherung ausführen. Die Option sichert den ausgewählten Tablespace.
Online - Control File only	Falls ausgewählt, wird eine Kopie der Steuerdatei gesichert.
Tape	Falls ausgewählt, erfolgt die Sicherung auf Band.
Device	Verweist auf das Bandgerät, das die Sicherung der Datenbankdateien aufnehmen wird.
Disk, Directory and Browse	Falls ausgewählt, wird die Datenbankdatei in das angegebene Verzeichnis gesichert. Oracle empfiehlt, den gesamten Pfadnamen anzugeben. Mit der Schaltfläche Browse wählen Sie den Standort, an dem die gesicherte Datendatei abgelegt werden soll.
Backup	Initiiert die Sicherungsprozedur.

Wird die Datenbank im NOARCHIVELOG-Modus betrieben oder sollte sie heruntergefahren sein, erscheint ein anderes Dialogfeld (siehe Abbildung 3-2), das in Tabelle 3-5 beschrieben ist.

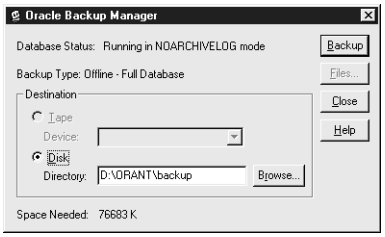


Abbildung 3-2: Das Dialogfeld des Backup Managers, wenn die Datenbank im NOARCHIVELOG-Modus läuft.

Tabelle 3-5: Die Elemente des Dialogfelds im Backup Manager, wenn die Datenbank im NOARCHIVELOG-Modus läuft.

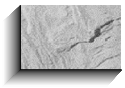
Dialogelement	Erklärung
Database Status	Zeigt den Status der Datenbank: ob sie im NOARCHIVELOG-Modus läuft oder nicht.
Backup Type	Zeigt, dass in diesem Modus nur ein Offline-Backup möglich ist.
Tape	Falls ausgewählt, wird die Sicherung auf Band ausgeführt.
Device	Zeigt das Bandgerät, das die Datenbankdateien sichert.
Disk, Directory and Browse	Falls ausgewählt, wird die Datenbankdatei in das angegebene Verzeichnis gesichert. Oracle empfiehlt, den gesamten Pfad anzugeben. Über die Schaltfläche Browse können sie den Standort auswählen, an dem die Datei gesichert werden soll.
Backup	Initiiert die Sicherungsprozedur.
Files	Diese Schaltfläche ist nur aktiviert, wenn die Datenbank inaktiv ist. Über diese Schaltfläche können Sie die Liste der ausgewählten Datenbankdateien vor Ausführung des Backups ansehen und ggf. ändern.

Sollten Sie die Schaltfläche Files zur Prüfung oder Änderung der Datenbankdateien wählen, erscheint ein anderes Dialogfeld. Hier sehen Sie alle selektierten Datenbankdateien und können Dateinamen hinzufügen, ändern oder löschen.

Die folgende Prozedur zeigt den Einsatz des Backup Managers:

1. Öffnen Sie den Backup Manager.
2. Nach Aufforderung geben Sie das Passwort für den Benutzer *internal* ein und wählen OK. Das Standard-Passwort ist *oracle*.
3. Falls Sie im NOARCHIVELOG-Modus arbeiten oder die Datenbank nicht aktiv ist, überspringen Sie diesen Punkt. Andernfalls wählen Sie Offline Full Database, Online Selected Tablespace oder Online Control File Only – je nachdem, was Sie sichern möchten.
4. Wählen als Sicherungsziel entweder Bandgerät (Tape) oder Laufwerk (Disk).
5. Sollten Sie sich im letzten Schritt für Disk entschieden haben, geben Sie entweder im Feld Directory den vollständigen Pfadnamen an oder selektieren diesen über die Schaltfläche Browse.
6. Ist die Datenbank geöffnet, überspringen Sie diesen Schritt. Andernfalls überprüfen Sie anhand der Schaltfläche Files, dass Sie die richtigen Daten-, Log- und Steuerdateien sichern.
7. Über Choose Backup starten Sie die Sicherungsprozedur. Sollte die Datenbank geöffnet sein und ein Offline-Backup angefordert werden, wird die Datenbank automatisch heruntergefahren und nach Ausführung der Sicherung neu gestartet.

Falls Sie die Daten bei einem heißen Backup manuell kopieren möchten, steht Ihnen dafür der Befehl **OCOPY*nn*** zur Verfügung (wobei ***nn*** die Version von **OCOPY** ist). Mit diesem Befehl lassen sich Sicherungen auf Laufwerke oder Disketten ausführen.



Hinweis
***OCOPYnn** wird nur in Oracle8 und älteren Versionen eingesetzt.*
*In Oracle8i heißt dieser Befehl **OCOPY**.*

Beim Kopieren auf Diskette lassen sich große Dateien mit der Option /B splitten. Beim Zurückschreiben von mehreren Disketten sollten Sie die Option /R nutzen. Das folgende Beispiel zeigt, wie man auf ein Laufwerk sichert, die Sicherung auf mehrere Disketten überträgt und diese Sicherung wieder auf ein Laufwerk von Oracle8*i* zurückschreibt.

```
C:\> OCOPY current_file backup_file
C:\> OCOPY /B current_file a:
C:\> OCOPY /R a: restore_dir
```