EDITION ORACLE

Johannes Ahrends, Dierk Lenz, Patrick Schwanke, Günter Unbescheid

Oracle 11g Release 2 für den DBA

Produktive Umgebungen effizient konfigurieren, optimieren und verwalten

ADDISON-WESLEY

3 Hardware und Betriebssysteme

Bei der Planung eines neuen Systems steht man häufig vor einer Vielzahl an Auswahlmöglichkeiten im Hinblick auf Hardware und Betriebssystem. In der Tat gibt es keine allgemeingültigen Empfehlungen für diese Thematik, sondern lediglich einige Erfahrungswerte sowie Empfehlungen, welche Fragen im Einzelfall beantwortet werden müssen, um hier zu einer guten Lösung zu gelangen. Ein paar grundsätzliche Überlegungen vorneweg:

- Gerade bei kritischen Anwendungen und Datenbanken empfiehlt es sich, lieber die zweitneueste als die neueste Technologie zu nutzen, um eventuellen Kinderkrankheiten, die es ja nun einmal auch in der IT gibt, aus dem Weg zu gehen. Andererseits sollten auch keine Auslaufmodelle zum Zuge kommen, sonst steht nach kurzer Zeit schon wieder eine aufwendige Migration auf die nachfolgende Technologie an. Hier kommt es also auf den richtigen Kompromiss an.

- Bei allen konkreten Überlegungen zum Thema Hardware- und Plattformauswahl sollte sehr frühzeitig auf eine Oracle-Zertifizierung geachtet werden. Insofern können auch alle Aussagen und Empfehlungen dieses Kapitels nur einen Gesamteindruck wiedergeben. Aufgrund der Dynamik der Materie und der vielen Detailregelungen sollte vor jeder Entscheidung daher immer die aktuelle Zertifizierungsmatrix von Oracle zurate gezogen werden.

  Eine zentrale Anlaufstelle hierfür ist der „Certify"-Bereich auf der My Oracle Support-Seite (siehe Abbildung 3.1). Ergänzend dazu gibt es die sogenannten *Oracle Validated Configurations*, die eine umfangreiche Liste bereits vorgetesteter und validierter Konfigurationen (Hardware, Betriebssystem, Oracle-Version, ggf. mit RAC oder ASM) darstellen.

- Bei allen Überlegungen und Entscheidungen sind natürlich auch die „Total Cost of Ownership" (TCO) zu berücksichtigen, also neben den Hardware-, Lizenz- und sonstigen Anschaffungskosten auch die Betriebskosten. Die Anschaffungskosten sind aufgrund der Vielfalt der Hersteller und Lizenzmodelle in den folgenden Abschnitten nicht weiter berücksichtigt, sollten aber im konkreten Fall frühzeitig betrachtet werden, um sie mit dem verfügbaren Budgetrahmen vergleichen zu können.

  Die Betriebskosten wurden in der Vergangenheit selten angemessen berücksichtigt und meistens auch einer anderen Kostenstelle zugeschlagen. Tatsächlich belaufen sich die Betriebskosten nicht selten auf ein Mehrfaches der Anschaffungskosten, wenn man beispielsweise die Kosten für Strom und Klimaanlage mit berücksichtigt. Diese Überlegungen führten in den letzten Jahren zur Virtualisierung, von der weiter unten ebenfalls die Rede sein wird.

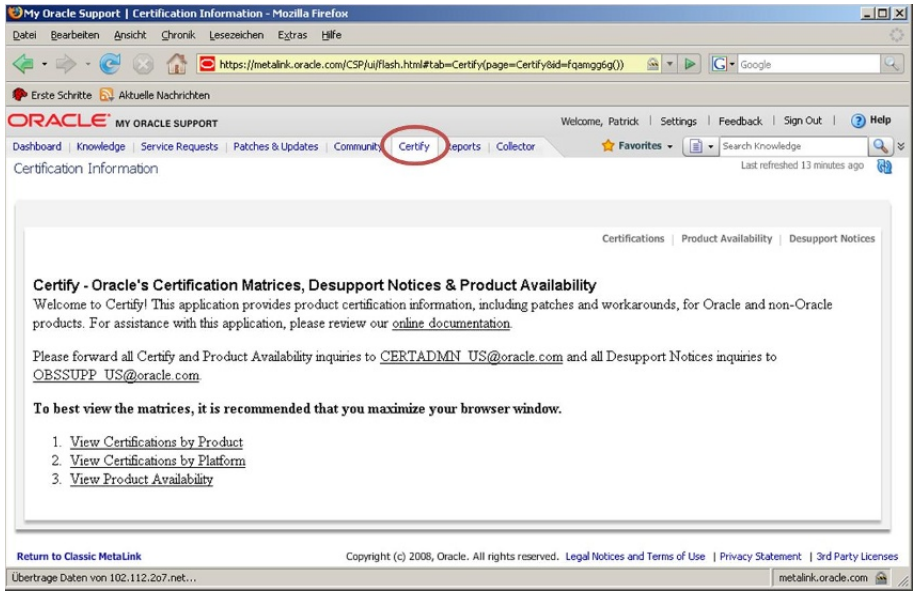


Abbildung 3.1: Certify-Bereich auf der My Oracle Support-Seite

3.1 Plattformen

Der Oracle-Datenbank ist für eine Vielzahl von Plattformen verfügbar. Darunter sind neben MS-Windows auch alle gängigen Unix- und Linux-Derivate. Ohne Anspruch auf Vollständigkeit oder Korrektheit seien hier genannt:

- Suse Linux Enterprise Server in den Versionen 9 und 10
- Redhat Enterprise Linux in den Versionen 4 und 5 sowie der Redhat-Klon „Oracle Enterprise Linux“ in den Versionen 4 und 5
- HP-UX 11i, also 11.11, 11.23 oder 11.31, auf Itanium oder PA-RISC. HP-UX 11.0 ist dagegen nicht mehr unterstützt.
- AIX 5L sowie AIX 6.1
- Sun Solaris 8, 9 und 10 auf der SPARC-Plattform. Solaris auf x86-Plattform ist nur noch bis einschließlich Oracle 10gR2 unterstützt.
- HP Tru64 ist nur noch in der Version 5.1b und nur bis einschließlich Oracle 10gR2 unterstützt.
- MS-Windows 2008, 2003, Vista und XP

Auf der Certify-Seite ist jeweils im Detail ausgeführt, welche Datenbankversion auf welcher OS-Version unterstützt ist und ob bzw. welche Patches oder zusätzlichen Packages des Betriebssystems benötigt werden.

Gerade die älteren Versionen eines Betriebssystems sind häufig nicht mehr für das jeweils neueste Oracle-Release zertifiziert und sollten daher für neu aufzusetzende Datenbanken nicht verwendet werden. Auch gibt es Unterschiede zwischen verschie-

denen Varianten eines Betriebssystems, z. B. zwischen Linux x86, Linux Itanium oder zSeries-basiertem Linux.

Auch wenn eine Vielzahl von Betriebssystemen formell unterstützt ist, gibt es doch zum Teil deutliche Unterschiede bei den Verfügbarkeiten von Datenbank-Releases, Patch-Releases oder einzelnen Patches für die verschiedenen Plattformen. So stand Oracle 11g beispielweise zuerst für Linux 64-Bit und Linux 32-Bit zur Verfügung, bald darauf auch für MS-Windows, und erst später kamen sukzessive die verschiedenen Unix-Plattformen hinzu. Dieselbe oder zumindest eine sehr ähnliche Erscheinungs-Reihenfolge lässt sich auch bei Patches und Patchsets beobachten.

Anders als noch vor einigen Jahren lassen sich – zumindest außerhalb des Cluster-Betriebs – keine eindeutigen Argumente pro oder kontra MS-Windows bzw. Unix/Linux mehr formulieren. Windows bietet eine stabile Plattform für den Oracle-Datenbankbetrieb, ist 64-Bit-fähig und steht relativ weit vorne in den Release-Zyklen. Dennoch laufen nach Erfahrung der Autoren ca. 75 % aller Oracle-Datenbanken unter Unix/-Linux. Bei Datenbanken im Cluster- bzw. RAC-Betrieb beträgt der Unix-/Linux-Anteil schätzungsweise 95 %.

Bei der Entscheidungsfindung für ein Betriebssystem sollten insbesondere folgende Aspekte genau beachtet werden:

- Gibt es Vorgaben durch die IT-Strategie des eigenen Unternehmens oder Anbieters? Manchmal ist eine bestimmte Plattform als strategische Plattform gesetzt, so dass man hier vor vollendete Tatsachen gesetzt wird.
- Wo gibt es das meiste Know-how im Unternehmen? Wenn es beispielsweise keinerlei Linux-Erfahrung gibt, sollte auch die Oracle-Datenbank ohne triftigen Grund nicht auf Linux laufen.
- Was wird von der Hardware unterstützt bzw. was ist für die Hardware zertifiziert? So ist z. B. ein Solaris x86-Betriebssystem nicht auf jedem x86-basierten Server unterstützt.

3.1.1 Unix und Linux

Da die meisten Oracle-Datenbanken auf Unix- oder Linux-Rechnern laufen, sollen an dieser Stelle einige Details zu Installation und Betrieb in dieser Umgebung gegeben werden. Auf Unix- und Linux-Systemen sind zunächst einige Vorarbeiten zu leisten, bevor mit der Installation der Datenbank-Software begonnen werden kann. Die einzelnen Schritte sind in dem jeweiligen *Installation Guide* der Dokumentation beschrieben, z. B. *Installation Guide 11g Release 2 for Linux*. Dazu gehören insbesondere

- das Einspielen benötigter Betriebssystem-Patches und Software-Packages (z. B. mittels RPM-Installer auf Linux, pkgadd auf Solaris)
- das Setzen einiger OS-Kernel-Parameter, insbesondere zur Konfiguration benötigter Shared Memory- und Semaphoren-Einstellungen.
- die Definition von Swap-Speicher in passender Menge
- das Erstellen bestimmter Betriebssystembenutzer und -gruppen, insbesondere der Eigentümer der Oracle-Software, z. B. `oracle`

- das Erstellen des Basisverzeichnisses für die Software-Installation (Oracle-Base-Verzeichnis)

Für die Software- bzw. Datenbankinstallation sollten auch die beiden folgenden Teile der Oracle-Dokumentation berücksichtigt werden:

- In den Appendix-Kapiteln des *Administrator's Reference for Linux and UNIX* werden einige Eigenheiten der unterschiedlichen Unix-Derivate aufgelistet.
- In den jeweiligen Release Notes, z. B. *Release Notes for Solaris Operating System (SPARC 64-Bit)* sind insbesondere bekannte Probleme im Umfeld von Installation, Konfiguration oder Upgrade der Software beschrieben.

Patchen des Betriebssystems

Auf Unix- und Linux-Plattformen besteht die Datenbank-Software letztlich aus einer Reihe von Libraries, die bei der Installation auf dem jeweiligen Server erstmalig gebunden werden. Bei der Installation eines Oracle-Patches oder -Patchsets werden aktualisierte Libraries eingespielt und automatisch neu gebunden.

Nach dem Einspielen eines Betriebssystem-Upgrades oder -Patches empfiehlt es sich grundsätzlich, eine Neubindung der Libraries manuell anzustoßen. Dazu gibt es das `relink`-Werkzeug im Verzeichnis `$ORACLE_HOME/bin`. Dieses Werkzeug muss als der Software-Owner des betreffenden Oracle-Homes aufgerufen werden. Außerdem müssen die Umgebungsvariable `$ORACLE_HOME` sowie plattformspezifisch Umgebungsvariablen für Library-Pfade zuvor entsprechend gesetzt sein:[1]

Für Oracle-Homes, die Datenbank-Software oder ASM-Software beinhalten, sieht dies z. B. so aus:

```
[root@ora11gr2 ~]# su - oracle

[oracle@ora11gr2 ~]$ echo $ORACLE_HOME
/u01/app/oracle/product/11.2.0/dbhome_1/

[oracle@ora11gr2 ~]$ $ORACLE_HOME/bin/relink all
```

In einem Client-Home würde dieses Verfahren diverse Fehlermeldungen erzeugen, da nur die Client-Komponenten vorhanden sind. Hier muss stattdessen die Option `relink client` verwendet werden.

Für die Oracle-Clusterware schließlich gibt es kein entsprechendes `relink`-Pendant. Hier können lediglich die sogenannten „Client Shared Libraries" neu gebunden werden:

```
[root@ora11gr2 ~]# su - grid

[grid@ora11gr2 ~]$ echo $ORACLE_HOME
/u01/app/grid

[grid@ora11gr2 ~]$ $ORACLE_HOME/bin/genclntsh
```

1 Für plattformspezifische Details siehe auch auf der My Oracle Support-Seite Doc-ID 131321.1.

Shared-Memory-Verwaltung

Jeder Server- oder Hintergrundprozess einer Oracle-Instanz hat seinen eigenen privaten Speicherbereich, die sogenannte PGA (Process Global Area). Außerdem teilen sich alle diese Prozesse einen gemeinsamen Shared-Memory-Bereich, die SGA (Shared Global Area). Nun gibt es auf Unix- und Linux-Systemen zwei Varianten der Shared-Memory-Verwaltung:

- Die System V[2]- oder IPC-Variante benutzt zur Allokation eines Shared-Memory-Segments den Systemaufruf `shmget()`. Dies ist die am weitesten verbreitete Form, Shared-Memory zu benutzen. Dabei wird ein monolithisches Shared-Memory-Segment erzeugt.
- Die POSIX[3]-Variante benutzt stattdessen den Systemaufruf `shm_open()`. Diese Möglichkeit wird bisher eher selten genutzt. Das Shared-Memory-Segment kann dabei aus kleinen einzelnen *Granulen* zusammengesetzt sein.

Bis zur Datenbankversion 10gR2 benutzt Oracle ausschließlich die System V-Technik. Einen einfachen Weg, die SGA auf Betriebssystemebene zu sehen, bietet das Kommandozeilen-Werkzeug `sysresv` im Verzeichnis `$ORACLE_HOME/bin` in Verbindung mit dem Betriebssystemkommando `ipcs`.

Mit `sysresv` kann unter Angabe der Oracle-SID die ID des Shared-Memory-Segments bestimmt werden, `ipcs` liefert die Größe des Segments (Spalte BYTES) und die Anzahl der darauf zugreifenden Prozesse (Spalte NATTCH).

```
[oracle@ora11gr2 ~]$ sysresv -l PS112

IPC Resources for ORACLE_SID "PS112" :
Shared-Memory:
ID              KEY
1179649         0xf01aba2c
Semaphores:
ID              KEY
1015808         0xc525cc68
Oracle Instance alive for sid "PS112"

[oracle@ora11gr2 ~]$ ipcs -m

------ Shared-Memory Segments --------
key        shmid    owner    perms    bytes      nattch
...
0xf01aba2c 1179649  oracle   660      348127232  30
...
```

Seit der Datenbankversion 11g ist es möglich, die Speicherverwaltung vollständig automatisch von der Oracle-Instanz durchführen zu lassen und mit dem Serverparameter `memory_target` nur noch eine Gesamt-Speichergröße anzugeben. Dieses sogenannte *Automatic Memory Management (AMM)* verschiebt insbesondere automatisch Teile des Gesamt-Speicherpools zwischen SGA und PGA.

---

2 System V, ursprünglich ein Unix-Betriebssystem von AT&T, bezeichnet heute eine ganze Familie von Unix-Derivaten, die einige gemeinsame Schnittstellen aufweisen, u. a. die genannte Schnittstelle zur Shared-Memory-Verwaltung.

3 POSIX steht für Portable Operating System Interface, eine standardisierte Schnittstelle zwischen Anwendung und Betriebssystem. Viele Unix-Betriebssysteme, insbesondere AIX, HP-UX und Solaris, sowie einige Linux-Distributionen sind POSIX-konform. Damit bieten Sie unter anderem auch eine Shared-Memory-Verwaltung gemäß POSIX.

Aktiviert man AMM nun in einer Oracle 11g-Datenbank, indem man `memory_target` auf irgendeinen Wert setzt, z. B. 500M, ergibt sich ein völlig anderes Bild: An der Stelle der SGA taucht nur noch ein „Dummy"-Segment mit einer Größe von 4 KB und ohne angehängte Prozesse auf.

Der Grund hierfür ist, dass es mit der System V-Methode nicht einfach möglich ist, Speicher zwischen dem monolithischen Shared-Memory-Segment und den privaten Speicherbereichen der Prozesse (PGA) zu verschieben. Genau das ist aber ja die Anforderung. Daher wechselt die Oracle-Instanz bei Benutzung von AMM – und nur dann – zu der alternativen Methode, dem POSIX-Shared-Memory-Management.

Dieses erlaubt es, Shared-Memory in kleine Portionen, sogenannte „Granulen", aufzuteilen. Diese Granulen können dann bei Bedarf einzeln dealloziert und in den PGA-Bereich alloziert werden, um Speicher zwischen SGA und PGA zu verschieben. Leider ist das auf diese Weise verwaltete Shared-Memory nicht mittels der üblichen Werkzeuge wie z. B. `ipcs` sichtbar. Stattdessen wird es über ein virtuelles Dateisystem verwaltet, z. B. `/dev/shm` unter Linux. Der Shared-Memory-Verbrauch via POSIX entspricht nun einfach der Belegung dieses virtuellen Dateisystems.

```
[oracle@ora11gr2 ~]$ df -k /dev/shm
Filesystem    1K-blocks      Used Available Use% Mounted on
tmpfs            614400    335688    278712  55% /dev/shm
```

Ein Blick in das Verzeichnis `/dev/shm` offenbart auch die einzelnen Granulen, die je nach Gesamtgröße des Speichers entweder 4 MB oder 16 MB groß sind. Manche haben auch die Größe 0, dieser Speicherbereich ist aktuell gerade der PGA zugeordnet, kann aber jederzeit in die SGA zurückgeholt werden.

```
[oracle@ora11gr2 ~]$ df ls -lah /dev/shm
...
-rw-r-----  1 oracle oinstall 4,0M Jan  5 19:51 ora_PS112_285474827_68
-rw-r-----  1 oracle oinstall 4,0M Jan  5 19:51 ora_PS112_285474827_69
-rw-r-----  1 oracle oinstall    0 Jan  5 19:51 ora_PS112_285474827_7
-rw-r-----  1 oracle oinstall 4,0M Jan  5 19:51 ora_PS112_285474827_70
...
```

Nützliche Werkzeuge für das Troubleshooting sind in diesem Zusammenhang die OS-Kommandos `fuser` sowie `pmap`.

`fuser` listet zu einer Shared-Memory-Granule die darauf zugreifenden Prozesse auf:

```
$ /sbin/fuser -v /dev/shm/ora_PS112_285474827_68

                     USER        PID ACCESS COMMAND
ora_PS112_285474827_68:
                     oracle    22848 ....m oracle
                     oracle    22850 ....m oracle
                     oracle    22854 ....m oracle
...
```

Umgekehrt listet `pmap` die zu einem Prozess gehörenden Shared-Memory-Granulen auf.

Für den Datenbankadministrator ist es auch wichtig zu wissen, dass vor dem Starten der Oracle-Instanz das zugrunde liegende virtuelle Dateisystem erst einmal passend dimensioniert werden muss. Ausschlaggebend ist hierbei der Wert für `memory_max_target`, der – wenn er nicht explizit festgelegt wird – standardmäßig den Wert von `memory_target` übernimmt.

Beim ersten Anlauf bekommt man häufig folgenden Fehler beim Startup der Instanz:

```
SQL> startup
ORA-00845: MEMORY_TARGET not supported on this system
```

Entgegen dem Wortlaut der Fehlermeldung ist die Benutzung von `memory_target` in den meisten Fällen sehr wohl unterstützt.[4] Ein Blick in das Alert-Log gibt genauere Auskunft:

```
[oracle@ora11gr2 trace]$ tail -3 alert_PS112.log
Starting ORACLE instance (normal)
WARNING: You are trying to use the MEMORY_TARGET feature.
This feature requires the /dev/shm file system to be mounted
for at least 2097152000 bytes. /dev/shm is either not
mounted or is mounted with available space less than this
size. Please fix this so that MEMORY_TARGET can work as
expected. Current available is 629145600 and used is 0 bytes.
Ensure that the mount point is /dev/shm for this directory.
memory_target needs larger /dev/shm
```

Die Lösung besteht also darin, das Dateisystem `/dev/shm` zu unmounten und neu zu mounten, wobei über den Mount-Parameter `size` eine höhere Größe spezifiziert wird.

```
[root@ora11gr2 ~]# umount /dev/shm
# Anpassen des Eintrags in /etc/fstab, z. B.:
tmpfs    /dev/shm    tmpfs    size=2048m    0 0
[root@ora11gr2 ~]# mount /dev/shm
```

Enterprise Linux und Unbreakable Linux

Oracle bietet mit *Oracle Enterprise Linux (OEL)* eine eigene Linux-Distribution an, die eng an Redhat Enterprise Linux (RHEL) angelehnt ist. Vom Umfang her entspricht sie der RHEL AS-Version, welche die umfangreichste Version der RHEL-Familie darstellt. Versionssprünge und Updates werden 1:1 nachvollzogen. Die Distribution kann frei von der Oracle-Seite heruntergeladen und eingesetzt werden, der Support ist jedoch kostenpflichtig. Beim Abschluss des Support-Vertrags erhält der Kunde eine CSI-Nummer (Customer Support Identifier). Mit dieser Nummer wird nach der Installation die Registrierung beim *Unbreakable Linux Network (ULN)* durchgeführt und der Update-Service aktiviert.

Die klare Trennung zwischen Distribution (OEL) und Support-Vertrag (ULN) einerseits sowie die Kongruenz von RHEL und OEL andererseits sind entscheidend wichtig, um zu verstehen, dass für einen Umstieg von RHEL auf OEL keine Neuinstallation des Betriebssystems nötig ist. Nach dem Abschluss des Support-Vertrags muss lediglich das Update-Werkzeug `up2date` vom ULN heruntergeladen und auf dem Redhat-System ausgeführt werden. Aus technischer Sicht ändert sich also lediglich der Mechanismus für Software-Updates, die nun nicht mehr vom Redhat Network, sondern vom ULN heruntergeladen werden. Die Installation des Update-Werkzeugs besteht aus folgenden Schritten:

- Installation zweier RPM-Pakete
- Importieren von Oracles GPG-Key. Dies geschieht ebenfalls über ein rpm-Kommando.

4 AMM ist auf Linux, Solaris, MS-Windows, HP-UX und AIX unterstützt.

- Aufruf von `up2date --register` und Durchlaufen des Registrierungs-Wizards

Beginnend mit OEL 5 Update 2 enthält die Distribution außerdem ein RPM-Paket namens `oracle-validated`. In früheren Versionen war dieses Paket ausschließlich über ULN verfügbar. Mit der Installation dieses Pakets werden diverse Voraussetzungen für eine Oracle Server-Installation überprüft und – wenn nötig – automatisch entsprechende Anpassungen vorgenommen. Dabei handelt es sich um:

- Installation weiterer RPM-Pakete
- Überprüfung und Anpassung von Kernel-Parametern gemäß Oracle-Empfehlungen
- Erstellung der einschlägigen Betriebssystembenutzer und -gruppen
- Setzen der OS Benutzer-Limits für den Betriebssystembenutzer gemäß Oracle-Empfehlungen
- Überprüfung und Anpassung der Bootloader-Parameter für den Grub-Bootloader
- Überprüfung und Anpassung von Modulparametern

Geänderte Dateien wie z. B. `/etc/sysctl.conf` werden zuvor gesichert, indem die Endung `.orabackup` angehängt wird. Ein Protokoll aller Aktionen steht in `/etc/sysconfig/oracle-validated/results/orakernel.log` zur Verfügung. Insgesamt stellt die Installation dieses Pakets eine deutliche Zeitersparnis dar, da nahezu alle Voraussetzungen abgedeckt werden. Bei einer nachfolgenden Installation der Oracle Server-Software sollten alle „Pre-Installation Requirements“ erfüllt sein.

3.1.2 MS-Windows

Oracle-Installationen unter MS-Windows erfreuen sich sowohl in Entwicklungs- und Test- als auch zunehmend in produktiven Umgebungen großer Beliebtheit. Gerade im Windows-Umfeld finden sich auch viele reine Client-Installationen, die lediglich die Oracle Net-Produkte zur Verbindung auf einen Datenbank-Server, diverse APIs sowie Entwicklungskomponenten umfassen.

Im Folgenden ist aber grundsätzlich von der Server-Variante zur Installation der Datenbank-Software die Rede. Die Software-Installation setzt übrigens nicht voraus, dass eine Server-Version vom MS-Windows installiert ist; auch auf Windows XP oder Windows Vista läuft die Datenbank-Software problemlos.[5] Dies ist insbesondere für Entwicklungs- und Übungsumgebungen auf Desktop-PCs oder Laptops interessant.

Viele spezielle Merkmale der Oracle-Software unter MS-Windows sorgen mittlerweile für eine sehr gute Integration in die Windows-Welt. Hierunter zählen u. a. die Integration der Alert-Informationen in die Windows-Ereignisanzeige, diverse Möglichkeiten zur Nutzung von Windows-Anmeldungen bei der Oracle-Autorisierung, die Verfügbarkeit zahlreicher Werkzeuge und Programmierschnittstellen sowie die schnelle Verfügbarkeit der kompletten Oracle-Software-Palette unter MS-Windows.

Der letzte Punkt ist ausschlaggebend dafür, dass mit MS-Windows wohl die vollständigste Entwicklungsumgebung in a box für Oracle-basierte Systeme gegeben ist.

---

5 Für Oracle 10g auf Windows Vista gibt es allerdings eigene Software-Pakete mit den Release-Ständen 10.2.0.3 und höher. Nur diese dürfen für Windows Vista verwendet werden!

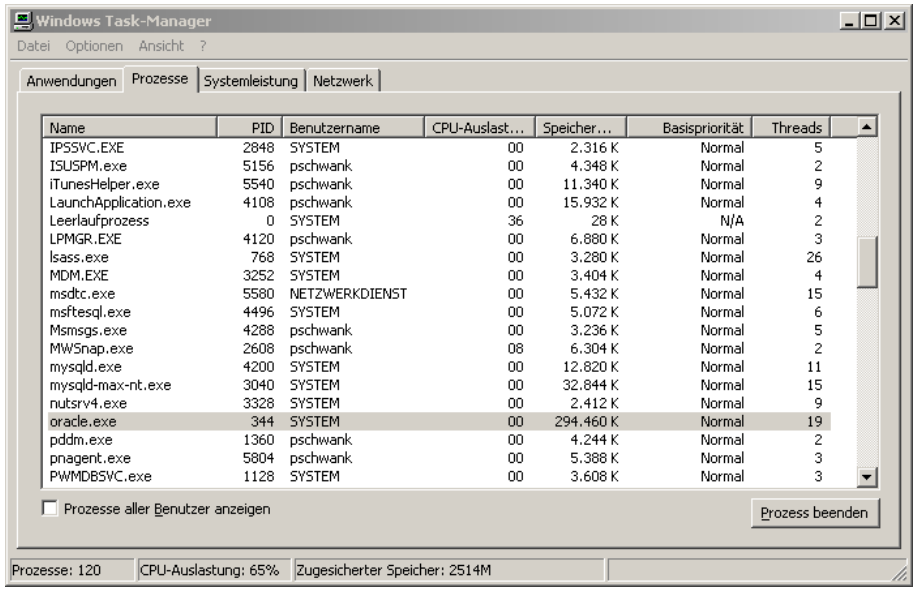


Abbildung 3.2: Oracle-Prozess mit Threads unter MS-Windows

Ein grundlegender Unterschied zwischen der Oracle-Implementierung unter MS-Windows und praktisch allen anderen Implementierungen ist die Prozessarchitektur: Anstelle der sonst üblichen Multiprozessvariante wurde die Oracle-Instanz unter MS-Windows mithilfe der Multithreaded-Architektur des Betriebssystems implementiert.

Mit dieser Architektur ist eine Betriebssystemressource wie Shared-Memory für die SGA nicht notwendig, da Speicheranforderungen am Prozess durchgeführt werden und somit automatisch allen Oracle-Threads zur Verfügung stehen. Diese Architektur kann man mit dem Windows Task-Manager nachvollziehen: Die Spalte „Threads“ in Abbildung 3.2 ist standardmäßig nicht eingestellt, sie muss erst über den Menüpunkt „Ansicht, Spalten auswählen“ aktiviert werden. Neben drei Threads, die grundsätzlich für den Oracle-Prozess gestartet sind, gibt es für jeden Hintergrund- und Server-“Prozess“ der Instanz genau einen Thread, im Beispiel sind es insgesamt 19.

Der Oracle-Prozess wird nicht als Anwendung auf der Oberfläche, sondern als Dienst gestartet. Nur so ist die Lauffähigkeit des Oracle-Systems ohne Abhängigkeit von einer Benutzeranmeldung am System möglich.

3.2 Prozessorarchitekturen

Auch die Entscheidung für einen Prozessortyp sollte vorher überdacht werden.

Während transaktionsorientierte Datenbanken (OLTP-Betrieb) durch die höhere Parallelisierung deutlich von Mehrkernprozessoren profitieren können, schrumpft dieser Vorteil bei Data-Warehouse-Systemen (OLAP-Betrieb). Außerdem sind bei Mehrkernprozessoren nicht zwangsläufig auch die I/O-Kanäle der CPU entsprechend mehrfach vorhanden. Da Datenbanken meistens I/O-lastig sind, sind in jedem Fall Prozessoren mit hoher I/O-Performance zu bevorzugen.

Eine Entscheidung 32-Bit versus 64-Bit wird wohl immer häufiger zugunsten der 64-Bit-Variante ausfallen, da 32-Bit-Prozessoren irgendwann der Vergangenheit angehören werden. Trotzdem sollen hier einige Argumente dargelegt werden, die auch verdeutlichen, welche Datenbanken tatsächlich von einer 64-Bit-Architektur profitieren können.

- Datenintensive Anwendungen aus dem OLTP-Bereich können von dem größeren Adressraum profitieren, der insbesondere für das Caching der Daten verwendet wird, um das I/O-Volumen zu reduzieren. Ob dies für eine bestehende Datenbank der Fall ist, kann z. B. mit der Performance-View `V$DB_CACHE_ADVICE` überprüft werden. OLAP-Anwendungen ziehen hieraus meistens weniger Nutzen, da der Schwerpunkt des Datenzugriffs auf Ladeoperationen sowie sogenannten „Full Table Scans“ und Sortier-Operationen liegt, die von einem größeren Daten-Cache nicht profitieren können.
- Rechenintensive Anwendungen schöpfen Mehrwert aus der höheren Registerbreite,[6] da insbesondere komplexe Berechnungen mit hoher Präzision schneller durchgeführt werden können. Indirekt wird dieser Effekt noch dadurch verstärkt, dass viele 64-Bit-Prozessoren einfach aufgrund ihrer Aktualität höhere Taktraten als 32-Bit-Prozessoren aufweisen. Auf Datenbanken bezogen bedeutet dies, dass insbesondere Anwendungen, die komplexe PL/SQL- oder Java-Routinen in der Datenbank benutzen, hiervon profitieren können.
- Betrachtet man eine gesamte Umgebung aus vielen Datenbanken, ist es häufig einfach kostengünstiger, anstelle vieler 32-Bit-Server wenige 64-Bit-Server zu betreiben. Letztere können – mit deutlich mehr Hauptspeicher ausgebaut – jeweils mehrere Datenbankinstanzen schultern. Zusammen mit Virtualisierungslösungen (siehe Abschnitt 3.8) stellt dies eine auch aus wirtschaftlichen Aspekten sehr attraktive Architektur dar.

3.3 Storage für Datenbanken

Heutzutage ist es kein Problem mehr, für kleine dreistellige Beträge Storage-Kapazitäten im Terabyte-Bereich zu bekommen. Stellte die reine Speicherkapazität also das einzige Auswahlkriterium dar, wäre nicht zu verstehen, warum viele Unternehmen für geschäftskritische Datenbanken fünf-, sechs- oder noch höherstellige Beträge in eine Storage-Lösung investieren.

In den folgenden Abschnitten werden Anforderungen an und passende Technologien für eine Storage-Lösung im Datenbankumfeld diskutiert. Es wird sich insbesondere zeigen, dass die reine Speicherkapazität heutzutage eher nebensächlich ist.

3.3.1 Storage-Kriterien

Die Storage-Anforderungen im Datenbankumfeld werden von drei Hauptfaktoren getrieben, die sich aus den geschäftlichen Anforderungen im Rahmen einer Applikation ergeben.

Performance Die angefragten Daten müssen innerhalb bestimmter, definierter Antwortzeiten zur Verfügung stehen. Häufig ist dies in Form von Service-Level-Verein-

6 In einer 64-Bit-CPU sind ja auch alle anderen CPU-Register auf 64 Bit Breite vergrößert.

barungen geregelt, z. B. „95 % aller Anforderungen eines bestimmten Typs müssen in weniger als zwei Sekunden beantwortet werden." Da produktive Datenbanken in den meisten Fällen intensives Random-I/O durchführen,[7] kommt es hier primär auf die I/O-Kapazität an, gemessen z. B. in I/O-Operationen pro Sekunde (IOPS).

„Desaster Recovery" Dabei steht die Frage nach Verfügbarkeit und Ausfallsicherheit im Vordergrund. Geben einzelne Komponenten den Geist auf, z. B. eine oder mehrere Festplatten oder Kabel, darf dies weder die Verfügbarkeit der Daten noch die Performance des gesamten Storage-Systems negativ beeinträchtigen. Es sei darauf hingewiesen, dass Verfügbarkeit sowohl eine Zeit- als auch eine Datendimension hat. In der Praxis muss daher immer ein sinnvoller Kompromiss zwischen tolerierbarem Datenverlust, tolerierbarer Ausfallzeit und Kosten einer geeigneten Storage-Lösung gefunden werden.

„Business Continuity" Damit ist – im Gegensatz zum Ausfall einer oder weniger Komponenten – der Ausfall z. B. eines kompletten Rechenzentrums gemeint. Der hierbei häufig genutzte Begriff „K-Fall"[8] umfasst allerdings nicht nur häufig zitierte, spektakuläre Ereignisse wie Flugzeugabsturz oder Naturkatastrophen.

In der Praxis überwiegen vielmehr die „kleinen Katastrophen", vom versehentlichen Betätigen eines Not-Aus-Schalters über den Ausfall der Klimaanlage, Wasser- oder Rauchschäden, Kabeldurchtrennungen durch Baggerarbeiten, Hardwaredefekte durch Spannungsspitzen bis hin zum Umzug eines Rechenzentrums. Auch in solchen Situationen müssen Datenverlust und Ausfallzeiten in definierten Grenzen gehalten werden, damit nicht das Schicksal des gesamten Unternehmens auf dem Spiel steht.

3.3.2 Storage-Technologien

Aus heutiger Sicht kommen vor allem diese vier Speichertypen für den Datenbankbetrieb infrage:

Fibre Channel (FC) Dabei handelt es um SCSI-basierte Platten mit FC-Anschluss. Diese Technologie ist ausgereift und gerade im Rechenzentrumsbetrieb gut etabliert, allerdings auch nicht ganz preisgünstig. Eine FC-Infrastruktur in Form von Glasfasern und FC-Switches wird benötigt. Die I/O-Kapazität liegt bei ca. 200 IOPS pro Festplatte, Speicherkapazitäten im 500-GB-Bereich sind mittlerweile verfügbar. Außerdem sind diese Festplatten für den Serverbetrieb, sprich für den Dauereinsatz, ausgelegt.

Serial Attached SCSI (SAS) Bei dieser, noch relativ neuen, Technologie werden dieselben Festplatten wie bei Fibre Channel verwendet. Lediglich der Anschluss erfolgt stattdessen über einen seriellen SCSI-Controller. Aktuell ist die Marktdurchdringung noch recht gering. In Zukunft könnte dieser Speichertyp aber eine kostengünstige Alternative zu den FC-Platten darstellen, mit denselben hohen I/O-Raten wie im FC-Bereich.

---

7 Vor allem transaktionsorientierte, also OLTP-Systeme führen vorrangig zufällige I/O-Zugriffe aus. In Data-Warehouse- bzw. OLAP-Systemen kann aber auch Sequential I/O, also sequenzielles Lesen und Schreiben, eine bedeutende Rolle spielen.

8 wahlweise für Krisen- oder Katastrophenfall

Serial ATA (S-ATA) Diese Festplatten sind zwar preisgünstiger als die vorgenannten Typen, allerdings nur bedingt für den Datenbankeinsatz geeignet. Zum einen liegt die I/O-Kapazität deutlich niedriger, typischerweise bei etwa 80 IOPS pro Festplatte, zum anderen sind nicht alle Festplatten dieses Typs für den Servereinsatz bzw. Dauerbetrieb ausgelegt. Ein Blick in die technische Dokumentation offenbart häufig erlaubte Betriebszeiten (oder POH = Power-On Hours) von beispielsweise acht Stunden pro Tag.

Bei längeren Nutzungsdauern besteht die Gefahr, dass die Festplatte vorzeitig den Geist aufgibt, sprich die sogenannte MTBF (Mean Time Between Failure) sinkt mit steigender Nutzungsdauer deutlich ab. Aus diesen Gründen sind solche Festplatten häufig zur Speicherung von Archivdaten mit gelegentlichen Zugriffen geeignet, nicht jedoch für den eigentlichen Transaktionsbetrieb. Die Speicherkapazitäten haben bereits die Terabyte-Marke überschritten.

Solid State Drives (SSD) Auch wenn diese Technologie noch relativ wenig Verbreitung gefunden hat, bietet sie doch eine Reihe interessanter Vorteile, insbesondere deutlich höhere Robustheit, Zugriffszeiten unterhalb einer Millisekunde, niedrigen Energieverbrauch und deutlich höhere I/O-Kapazitäten (IOPS).

Dem stehen heutzutage Nachteile bei der Speicherkapazität und – zumindest auf den ersten Blick – beim Preis gegenüber. SSDs sind zwar scheinbar teurer als selbst FC-basierte herkömmliche Festplatten. Diese Betrachtung stimmt aber zumindest bei Flash-basierten SSDs nur für den Preis pro Speicherkapazität, also pro GB. Wie oben erwähnt, ist aber häufig eher die I/O-Kapazität der bestimmende Faktor. Betrachtet man die Anschaffungs- oder auch die Stromkosten pro IOPS, schneiden SSDs meistens besser ab als herkömmliche Festplatten. Dass SSDs dennoch wenig verbreitet sind, liegt auch am Phänomen der begrenzten Wiederbeschreibbarkeit („wear-out“):

Flash-basierte SSDs erlauben heutzutage zwischen einer und fünf Millionen Schreibzyklen pro Block. Dieser Wert hat sich im Laufe der letzten Jahre stetig erhöht. Außerdem sorgen manche SSD-Festplatten automatisch für eine gleichmäßige Verteilung der Schreibzyklen, indem häufig beschriebene Blöcke ggf. intern verschoben werden. Trotzdem ist dies für schreibintensive Systeme wie OLTP-Datenbanken ein wichtiger Aspekt, der bedacht werden sollte, da er die Haltbarkeit der Festplatte einschränkt. Beispielhaft seien hier die sogenannten EFDs (Enterprise Flash Disks) von EMC[2] genannt.

SDRAM-basierte SSDs haben zwar keine Einschränkung bei der Zahl erlaubter Schreibzyklen und außerdem nochmals deutlich bessere Zugriffszeiten, sind aber wiederum um einige Faktoren teurer als Flash-basierte SSDs. Solche Platten werden z. B. von Texas Memory Systems angeboten.

Um den Performance-Anforderungen einer Anwendung Rechnung zu tragen, ist wie gesagt die I/O-Kapazität eine wichtige Kennzahl. Ergibt sich beispielsweise aus Hochrechnungen oder aus Messungen an existierenden Altsystemen ein Bedarf von 5000 IOPS, sollte das Storage-System mindestens 25 FC-Festplatten mit je 200 IOPS umfassen. Ob die daraus resultierende Speicherkapazität, z. B. 10 TB, benötigt wird, ist zweitrangig. Da die Speicherkapazitäten in den letzten 20 Jahren um mehrere Größenordnungen, die I/O-Raten dagegen nur um wenige Faktoren gewachsen sind, ist es sogar ein typisches Phänomen, dass Überkapazitäten im Storage-Bereich entstehen.

Dies ist auch gerade bei Konsolidierungen von Altsystemen zu berücksichtigen, wenn eine Anzahl alter Server auf ein neues System zusammengelegt wird. Die Datenmengen passen anschließend vielleicht auf wenige neue Festplatten. Allzu leicht wird darüber vergessen, dass die I/O-Kapazität einer neuen Festplatte eben nicht diejenige von zehn alten Festplatten ersetzen kann.

Von den Herstellern der Storage-Systeme werden meistens zwei Werte zur I/O-Performance angegeben, nämlich ohne und mit Berücksichtigung des Storage-Caches, letztere typischerweise um einen Faktor 5 bis 10 höher als erstere. Ein solcher Lese- und Schreib-Cache ist in praktisch jedem heutigen Storage-System integriert und bewirkt eine Verbesserung der I/O-Leistungen.

Dennoch sind die herstellerseitigen Angaben kritisch zu betrachten. Die IOPS-Werte für das Lesen aus dem Cache sind meistens Benchmark-Werte, die unter stark optimierten Bedingungen gewonnen sind und mit dem realen Datenbankbetrieb wenig zu tun haben. Realistischer sind die nativen, plattenbasierten IOPS-Werte, die bis zur maximalen Leistung des Storage-Controllers linear mit der Anzahl der Festplatten skalieren. Realistische cache-basierte Werte ergeben sich, indem auf die nativen Werte 10 bis 20 Prozent aufgeschlagen werden.

Ein wichtiger Aspekt beim Storage-Cache ist auch die Möglichkeit einer Cache-Segmentierung. Diese noch relativ neue Technologie trägt der Tatsache Rechnung, dass ein Storage-System im Allgemeinen von mehreren Anwendungen und Datenbanken benutzt wird. Durch Segmentierung des Caches kann eine einzelne Datenbank nicht einfach den kompletten Cache fluten und dadurch die Performance anderer Systeme beeinflussen, die dasselbe Storage-System benutzen.

Allgemein profitieren lesende Zugriffe vom Storage-Cache weniger als schreibende Zugriffe, da die für Cache-Effizienz entscheidende Zugriffslokalität größtenteils bereits vom Buffer-Cache der Oracle-Instanz „abgefangen“ wird. Insofern sollte der größere Teil des Storage-Caches für schreibende Zugriffe konfiguriert werden.

Es versteht sich von selbst, dass der storage-seitige Cache als NVRAM, also nicht-flüchtiger, batteriegestützter Speicher, ausgeführt sein muss, da die Oracle-Instanz davon ausgeht, dass ein vom Storage-System bestätigter Schreibvorgang auch tatsächlich auf den Platten angekommen ist. Bei flüchtigem Cache würde ein Stromausfall zu einem Verlust z. B. von Redolog-Daten führen, so dass eine nachfolgende Wiederherstellung der Datenbank nicht gewährleistet werden könnte. Ein nicht batterie-gestützter Storage-Cache muss daher deaktiviert werden!

Aus demselben Grund muss auch ein Schreib-Cache auf dem lokalen Host-Bus-Adapter deaktiviert werden. Wie dies bewerkstelligt wird, unterscheidet sich allerdings von Hersteller zu Hersteller.

3.3.3 Storage-Anbindung

Praktisch alle Datenbanken, mit denen ein Unternehmen Geld verdient oder die sonstwie für die betrieblichen Prozesse wichtig sind, werden heutzutage an ein zentralisiertes Storage angebunden, also entweder ein SAN oder NAS. Traditionell haben die SAN-Lösungen im Oracle-Umfeld einen besseren Ruf als solche auf NAS-Basis. Dies hat auch einige handfeste Gründe:

- Unabhängig von der konkreten Implementierung, sei es über Glasfaser (FC SAN), IP (iSCSI) oder Ethernet (FCoE), kommt letztlich immer SCSI als Kommunikationsprotokoll zum Einsatz. Das blockorientierte SCSI-Protokoll passt technisch sehr gut zum ebenfalls blockorientierten Zugriff einer Oracle-Instanz auf die Datenbank-Dateien
- Vor allem aber hatten FC SANs über einen langen Zeitraum schlicht höhere Bandbreiten als die jeweils zeitgenössischen Netzwerktechnologien.
- Hinzu kommt die Fülle an Storage-Technologien wie beispielsweise Split-Mirror-, Snapshot- oder Replikationstechniken, die zwar nicht ausschließlich, aber doch überwiegend im SAN-Bereich entstanden sind. Gerade kritische und hochlastige Umgebungen wurden daher häufig an ein SAN angebunden.

In den letzten Jahren hat diese Differenzierung aber an Bedeutung verloren. Zum einen stehen mittlerweile auch in NAS-Umgebungen entsprechende Storage-Technologien zur Verfügung. Zum anderen stehen mit Gigabit- und 10-Gigabit-Ethernet Netzwerkprotokolle zur Verfügung, die den Vergleich mit Glasfaser hinsichtlich Bandbreite nicht zu scheuen brauchen. Dass die Preise im Netzwerkbereich tendenziell schneller fallen als im Glasfaserbereich, leistet weiteren Vorschub für NAS-Lösungen, aber auch für alternative SCSI-basierte Lösungen wie iSCSI und FCoE.

Insgesamt ist die Situation also nicht mehr so klar wie noch vor einigen Jahren. Entsprechend finden sich mittlerweile auch viele Zwitter-Lösungen wie z. B. EMC^2 Celerra, die NetApp FAS-Serie oder Openfiler, die als NAS/SAN-Appliances beide Welten abdecken.

Lokale Festplatten, also DAS (Direct Attached Storage), findet dagegen im Umfeld produktiver Datenbanken nahezu keine Verwendung mehr und soll daher an dieser Stelle auch nicht weiter diskutiert werden.

NAS-basierte Lösungen bestehen in der Praxis meistens aus einem NFS-Server, der ein oder mehrere Dateisysteme exportiert. Wichtig ist, hier zunächst zwischen Single-Instance- und RAC-Datenbanken zu unterscheiden. Bis Anfang 2007 gab es das sogenannte Oracle Storage Compatibility Program (OSCP), in dem NFS-Server und angelehnte Technologien wie Replikation und Snapshotting von Oracle zertifiziert wurden.

Mit der Beendigung des OSCP stellt sich die Situation nun wie folgt dar: Für RAC-Umgebungen gibt es weiterhin eine über die Certify-Seite im My Oracle Support erreichbare Liste zertifizierter NFS-Server. Andere Systeme sind nicht unterstützt (siehe auch Abschnitt 3.6).

Für Single-Instance-Umgebungen reicht ein beliebiger NFS-Server, solange bestimmte Vorgaben für die Mount-Optionen eingehalten werden. Es gibt dabei vor allem plattformbedingt leichte syntaktische Unterschiede. Grundsätzlich ist Folgendes zu beachten:

- Die Optionen `rw`, `hard`, `noac`, `actimeo=0`, `rsize` und `wsize` sind zwingend erforderlich, wobei `rsize` und `wsize` typischerweise auf 32768 eingestellt werden.
- Wenn möglich, sollte NFSv3 über TCP verwendet werden. Dies wird durch die Mount-Optionen `vers=3` und `proto=tcp` (bzw. einfach nur `tcp` unter Linux) erreicht. Prinzipiell sind aber auch NFSv2 sowie UDP als Transportprotokoll erlaubt.
- Windows als NFS-Client ist nicht unterstützt.

- Weitere benötigte Einstellungen sind auf der My Oracle Support-Seite Doc-ID 359515.1 beschrieben.

Direct NFS

Beginnend mit der Datenbankversion 11g kann sich die Oracle-Instanz selbst als direkter NFS-Client verhalten und auf diesem Wege Datenbankdateien auf NFS-Servern lesen und schreiben. Dies bietet einige interessante Vorteile:

- Der integrierte NFS-Client ist nicht an die Funktionsweise eines OS-Kernel-basierten NFS-Clients gebunden, der lediglich in begrenztem Maße, nämlich über die Mount-Optionen, beeinflusst werden kann. Vielmehr können hier einige Oracle-spezifische Optimierungen greifen, insbesondere eine Umgehung des Betriebssystem-Caches.
- Die Oracle-Instanz kann intern ein Multipathing über bis zu vier gleichzeitige Pfade durchführen. So kann auch unabhängig von OS-spezifischen Mechanismen wie NIC-Bonding oder -Teaming und der ansonsten dafür notwendigen Unterstützung durch Switches eine entsprechende Verfügbarkeit und nahezu lineare Durchsatzskalierung erreicht werden.
- Die NFS-Kommunikation wird automatisch von der Oracle-Instanz konfiguriert. Auch spezielle Einstellungen z. B. für RAC-Datenbanken müssen nicht mehr von Hand vorgenommen werden. Insofern ist auch das Risiko hinsichtlich Support und Zertifizierung geringer.
- Auch Oracle-Instanzen auf MS-Windows können auf diese Weise NFS-basiertes Storage nutzen.

Direct NFS wird auf Ebene eines Oracle-Homes eingeschaltet und konfiguriert. Werden mehrere Instanzen aus einem Oracle-Home heraus gestartet, sind also jeweils alle Instanzen betroffen.

Um den Direct NFS-Client zu nutzen, muss die sogenannte *Oracle Disk Manager (ODM)*-Library ausgetauscht werden.[9] Die ODM-Library, die standardmäßig genutzt wird, beherrscht kein Direct NFS. Die Dateinamen und Pfade für die Standard- und die NFS-fähige Library sind betriebssystemspezifisch und in der Oracle-Dokumentation im *Oracle Clusterware Installation Guide* für die jeweilige Plattform beschrieben, z. B.:

- auf Linux: libodm11.so und libnfsodm11.so im Verzeichnis `<Oracle-Home>/lib`.
- auf MS-Windows: oraodm11.dll und oranfsodm11.dll im Verzeichnis `<Oracle-Home>\bin`.

Auf MS-Windows muss die Oracle-Instanz zunächst heruntergefahren werden, dann werden die Libraries getauscht und die Instanz wieder gestartet:

```
C:\> cd %ORACLE_HOME%\bin
C:\Oracle\Ora11g\BIN> copy oraodm11.dll oraodm11.dll.stub
C:\Oracle\Ora11g\BIN> copy /Y oranfsodm11.dll oraodm11.dll
```

Auf Linux und Unix können zwar alternativ bei noch laufender Instanz die Libraries getauscht werden. Trotzdem ist ein Neustart der Instanz nötig, damit die neue ODM-

9 Wer sich mit RMAN-Sicherungen der Oracle-Datenbanken auf Band beschäftigt, kennt wahrscheinlich ein ähnliches Verfahren. Auch dort wird eine Oracle-Library, nämlich die Media-Management-Library (MML), ausgetauscht.

Library auch tatsächlich genutzt wird. Das Tauschen der Libraries geht z. B. für Linux wie folgt:

```
$ cd $ORACLE_HOME/lib
$ cp libodm11.so libodm11.so_stub
$ ln -s libnfsodm11.so libodm11.so
```

Vor einer Direct NFS-Konfiguration sollten die betreffenden NFS-Dateisysteme zunächst ganz normal über das Betriebssystem gemountet werden. Die konkreten Mount-Optionen spielen hierbei keine Rolle, da diese vom Direct NFS ohnehin nicht verwendet werden. Es geht nur darum sicherzustellen, dass das Mounting grundsätzlich funktioniert.

Im Zuge der Konfiguration muss nun vor allem angegeben werden, welche NFS-Dateisysteme von welchen NFS-Servern überhaupt exportiert werden und wo diese gemountet werden sollen. Beim normalen Betriebssystem-NFS wird dies über Einträge in der Datei `/etc/fstab` definiert. Die tatsächlich erfolgten NFS-Mounts sind anschließend in der Datei `/etc/mtab` (bei AIX: `/etc/filesystems`) eingetragen.

Die Oracle-Instanz kann nun wahlweise einfach die Einträge aus der `/etc/mtab` übernehmen oder eine eigene Konfiguration nutzen. Insgesamt wird in folgender Reihenfolge nach Mount-Einträgen gesucht:

1. In der Datei `<Oracle-Home>/dbs/oranfstab`, bei MS-Windows entsprechend mit Backslashes. Unter Windows ist dies auch die einzig mögliche Konfigurationsdatei für Direct NFS.
2. In der Datei `/etc/oranfstab` (für Unix und Linux).
3. In der jeweiligen vom Betriebssystem gepflegten Datei für gemountete Dateisysteme. Bei AIX ist dies die Datei `/etc/filesystems`, auf anderen Unix-Derivaten und Linux die `/etc/mtab`. Hierfür ist keine gesonderte Konfiguration erforderlich, da diese Datei vom Betriebssystem gepflegt wird.

Die Oracle-eigenen Konfigurationsdateien haben ein deutlich anderes Format als die `/etc/fstab`. Sie enthalten pro NFS-Server je einen Block der Form:

```
server: NfsServerName
path: NfsServer-IP-Adresse
[local: OracleServer-IP-Adresse]
... (maximal 4 path-/local-Paare, local jeweils optional)
export: /path/on/nfs/server mount: /path/to/mountpoint
... (beliebig viele export-/mount-Zeilen)
[uid: User-ID]
[gid: Group-ID]
```

Also z. B.:

```
server: openfiler11g
path: 10.10.8.150
path: 10.10.8.151
export: /mnt/vg_nfs/nfsshare mount: /u03/oradata
```

In diesem Fall exportiert der NFS-Server namens `openfiler11g` das Verzeichnis `/mnt/vg_nfs/nfsshare`, das lokal – also auf dem Oracle-Server – unter dem Verzeichnis `/u03/oradata` eingehängt wird. Die beiden Pfadeinträge geben IP-Adressen und damit Netzwerkkarten des NFS-Servers an. Optional kann mit einem `local`-Eintrag auch eine lokale IP-Adresse des Oracle-Servers festgeschrieben werden, von der aus der NFS-Server

erreichbar ist. Hierüber wird das Multipathing realisiert. Im obigen Beispiel gibt es also zwei Pfade zum NFS-Server, zwischen denen Lastverteilung und Failover stattfinden.

Um ein „Teaming" zweier Netzwerkkarten des Oracle-Servers zu realisieren, wäre z. B. folgende Konfiguration geeignet (in diesem Beispiel findet keine Lastverteilung statt, da beide Pfade an derselben IP-Adresse des NFS-Server zusammenlaufen):

```
server: openfiler11g
path: 10.10.8.150
local: 10.10.11.47
path: 10.10.8.150
local: 10.10.14.32
export: /mnt/vg_nfs/nfsshare mount: D:\testnfsmount
uid: 101
gid: 101
```

Hier sind auch zwei weitere, optionale und nur unter Windows verfügbare Einträge zu sehen, die festlegen, mit welcher User- und Group-ID sich Oracle beim NFS-Server anmeldet (bei Unix- und Linux-basierten Oracle-Servern werden einfach die User- und Group-ID des Software-Owners genutzt). Lässt man diese Einträge weg, wird für beide der Wert 65534 verwendet, was auf den meisten Systemen dem Benutzer „nobody" und der gleichnamigen Gruppe entspricht. Was der NFS-Server mit diesen Anmeldedaten macht, ist eine andere Frage und wird auf dem NFS-Server über die Exportoptionen `root_squash`, `all_squash`, `anonuid` und `anongid` geregelt.

Bei Oracle auf Windows steht – wie oben gezeigt – als Mount-Pfad ein Windows-Pfad, z. B. `D:\oradata`. Dieses Verzeichnis muss auf dem Datenbank-Server nicht existieren. Wenn es existiert, wird es nicht benutzt. Beim Anlegen z. B. eines Tablespace mit einer Datendatei in diesem Verzeichnis erkennt die Oracle-Instanz automatisch, dass es sich um einen Direct NFS-Pfad handelt, und legt die Datei auf dem NFS-Server an.

Bei mehreren NFS-Servern enthält die Konfigurationsdatei eine entsprechende Anzahl dieser Blöcke, jeweils durch eine Leerzeile getrennt. Auf Oracle-Servern unter MS-Windows ist der Mount-Eintrag entsprechend ein Windows-Pfad.

Bei einer RAC-Datenbank müssen alle Cluster-Knoten mit derselben Konfigurationsdatei versehen sein.

Aufseiten des NFS-Servers müssen natürlich auch einige Voraussetzungen erfüllt sein, damit das alles funktioniert:

- Der NFS-Server muss die NFS-Version 3 unterstützen.
- Bei Linux-basierten NFS-Servern, also z. B. auch Openfiler, gibt es eine Spezialität: Der NFS-Server erwartet standardmäßig, dass der Client sich von einem reservierten Port, also einer Portnummer unterhalb von 1024, aus anmeldet. Ansonsten wird die Verbindung abgewiesen. Für den normalen NFS-Client des Betriebssystems ist dies kein Problem, da dieser als root-Benutzer läuft. Die Oracle-Instanz läuft aber nicht als root, kann daher keinen reservierten Port benutzen und wird somit vom NFS-Server abgewiesen. Dies äußert sich dann in Einträgen wie dem folgenden im Alert-Log der Oracle-Instanz:

  ```
  Direct NFS: NFS3ERR 1 Not owner. path 10.10.8.150 mntport 754 nfsport 2049
  ```

  Leider ist diese Fehlermeldung nicht gerade sprechend. Die Lösung ist eine zusätzliche, nur für Linux-basierte NFS-Server benötigte Exportoption namens `insecure`.

Der Eintrag in der Datei `/etc/exports` auf dem NFS-Server muss also z. B. so aussehen:

```
/mnt/vg_nfs/nfsshare ora11gr2(rw,sync,all_squash,anonuid=101,anongid=101,insecure)
```

Als einfache Kontrolle, ob alles funktioniert, bietet sich eine Abfrage der View `V$DNFS_FILES` an:

```
SQL> SELECT * FROM v$dnfs_files;

FILENAME                         FILESIZE       PNUM     SVR_ID
-------------------------------- ---------- ---------- ----------
/u03/oradata/testnfs01.dbf       52436992           9          1
/u03/oradata/testnfs02.dbf       31465472           9          1
```

Liefert diese Abfrage keine Zeilen zurück, funktioniert der Direct NFS-Zugriff noch nicht. Im Alert-Log sollten dann entsprechende Fehlermeldungen zu finden sein. In diesem Fall sind dennoch ganz normal Lese- und Schreibzugriffe auf die entsprechenden Tablespaces und Datendateien möglich. Die Oracle-Instanz verwendet einfach weiterhin den NFS-Client des Betriebssystems. Für die Fehlersuche und Analyse muss deshalb auch stets das Oracle Alert-Log zurate gezogen werden.

3.3.4 Storage-Hochverfügbarkeit

Beim Thema Verfügbarkeit und Ausfallsicherheit muss vor allem auf zwei entscheidende Kriterien abgestellt werden:

Recovery Time Objective (RTO) Welche Ausfallzeit ist akzeptabel? Mit der RTO ist die Zeitspanne vom Schadensfall bis zur vollständigen Wiederherstellung gemeint. In der Praxis sind zwischen 0 und mehreren Tagen alle Anforderungen vertreten.

Recovery Point Objective (RPO) Welches Ausmaß an Datenverlust tolerierbar? Auch hier handelt es sich um eine Zeitspanne. Bei einem Schadensfall dürfen maximal Daten aus der RPO-Zeitspanne vor dem Schadensfall verloren gehen. Ein RPO von 0 bedeutet also, dass kein Datenverlust hinnehmbar ist.

Klar ist: Je strenger die Anforderungen an RTO bzw. RPO, desto höher ist der technologische Aufwand zur Realisierung und desto teurer ist eine entsprechende Lösung. Die Grenzfälle RTO = 0 und RPO = 0 sind in der Praxis nicht garantierbar.

An dieser Stelle sollte auch von vornherein an die Auswahl eines geeigneten Dateisystems gedacht werden.[10]. Journaling-Dateisysteme sind mittlerweile Standard und haben den großen Vorteil, dass nach einem Systemabsturz oder Stromausfall keine zeitraubende Überprüfung des gesamten Dateisystems mit oft erfolglosen Reparaturversuchen anfällt.[11] Vielmehr kann jederzeit innerhalb von Sekunden ein konsistenter Zustand wiederhergestellt werden. Beispiele für Journaling-Dateisysteme sind XFS, JFS, ext3/ext4 oder das Veritas File System (VxFS).

Eine der größeren Herausforderungen im Storage-Bereich ist die Vermeidung von Datenkorruptionen auf den Festplatten. Diese können ihren Ursprung in Festplattenfehlern, Controller-Fehlern, Bugs in der Storage-Software oder in der Datenbank-Software haben.

10 falls überhaupt ein Dateisystem verwendet wird. Eine gute Alternative zu einem Dateisystem ist die Verwendung von Oracle Automatic Storage Management (ASM), das weiter unten beschrieben wird.

11 Der Zeitaufwand für solche Überprüfungen kann durchaus im Stundenbereich liegen.

Standardmäßig überprüft die Instanz bei jeder Änderung sowie bei jedem Lesen, Sichern oder Wiederherstellen eines Oracle-Blocks eine im Block hinterlegte Prüfsumme. Dadurch können Datenkorruptionen seitens der Oracle-Software größtenteils vermieden werden. Korruptionen, die innerhalb des Storage-Systems entstehen, werden jedoch erst nachträglich, z. B. beim nächsten Lesen des Blockes, erkannt. Eine einfache Behebung ist dann meistens nicht mehr möglich. Durch die zunehmende Komplexität der Storage-Systeme steigt auch die Wahrscheinlichkeit solcher Probleme.

Oracle arbeitet daher im Rahmen der sogenannten HARD[12]-Initiative mit verschiedenen Storage-Herstellern zusammen. Durch eine gemeinsame Festlegung des Prüfsummen-Algorithmus können diese eine Datenvalidierung innerhalb des Storage-Systems durchführen. Somit werden korrupte Daten gar nicht erst auf die Festplatten geschrieben, sondern führen sofort zu einer Fehlermeldung, so dass das Problem zeitnah behoben werden kann.

Diese Technologie findet sich z. B. als NetApp Snap Validator for Oracle oder EMC[2] Double Checksum, aber auch für bestimmte Storage-Systeme von HP, Hitachi, Sun und weiteren Anbietern. Eine genaue Referenzliste findet sich auf den Oracle-Webseiten, erreichbar über jede Suchmaschine z. B. mit den Stichwörtern „Oracle“ und „HARD“.

Grundsätzlich sollten für produktive und geschäftskritische Datenbanken auch die Storage-Komponenten hochverfügbar ausgelegt sein.

Dazu gehört zunächst einmal, dass fehleranfällige Komponenten wie Festplatten, Lüfter oder Netzteile redundant und „hot-swappable“ ausgelegt sind, also bei einem Ausfall ein Ersatzteil automatisch einspringt und das defekte Teil im laufenden Betrieb ausgewechselt werden kann.

Durch Multipathing-Technologien kann ein Ausfall eines Host Bus Adapters (FC- oder iSCSI-HBA), Ports, Kabels oder Switches abgefedert werden. Jeder Datenbank-Server und jedes Storage-Array sollte insofern zumindest mit zwei HBAs (oder einen Dual-Port HBA) ausgestattet sein, wobei die beiden Ports mit unterschiedlichen Switches verbunden werden.

SAME (Stripe and Mirror Everything)

Bei den Festplatten selbst hat sich seit vielen Jahren die RAID[13]-Technologie bewährt. Oracle favorisiert in diesem Zusammenhang für die Datenbankdateien eine Konfiguration namens *SAME (Stripe and Mirror Everything)*:

- Aus Performance-Gründen sollten alle Dateien über viele Festplatten gestripet werden (RAID 0), und zwar mit einer Stripe-Größe von 1 MB. Neben der erhöhten besseren Performance für wahlfreien Zugriff (Random-I/O) wird dadurch auch eine möglichst gleichmäßige Auslastung der Platten-Queues erreicht.

  Gleichzeitig führt die relativ hohe Stripe-Größe von 1 MB dazu, dass sequenzielle Zugriffe, z. B. Full Table Scans, auch die hohen Transferraten der Festplatten ausnutzen und nicht bloß zu vielen einzelnen Random-I/Os im Storage-Array führen. Bei Festplatten mit besonders hohen Transferraten ist auch eine Erhöhung der Stripe-Größe auf wenige MB sinnvoll, um diese besser auszunutzen.

12 Hardware Assisted Resilient Data

13 Redundant Array of Independent Disks

Deutlich kleinere Werte sind dagegen i. A. nicht sinnvoll, da das Storage-System dann tendenziell zu viel Zeit mit dem Positionieren der Schreib-/Leseköpfe verbringt, also die Gesamteffizienz wieder schlechter wird. Hat man ein dediziertes Storage-Array für eine einzelne Datenbank zur Verfügung, sollte tatsächlich über alle Platten gestripet werden. In der Praxis nutzt eine Datenbank häufig einige bis wenige Dutzend Platten, über die dann gestripet wird.

- Aus Gründen der Hochverfügbarkeit sollten alle Dateien gespiegelt werden (RAID 1). Dadurch erhöht sich natürlich die Anzahl benötigter Festplatten, bei Einfachspiegelung auf das Doppelte. In den allermeisten Fällen genügt dies, lediglich in Ausnahmefällen extrem kritischer Anwendungen lohnt sich eine Mehrfachspiegelung.

  Es sollte auch bedacht werden, dass eine einfache Plattenspiegelung in einem Storage-Array nicht alle denkbaren Fehlerfälle abdeckt. So werden z. B. Benutzer- oder Anwenderfehler, die zu Datenverlusten führen, einfach mitgespiegelt. Hier sind andere Lösungen, z. B. Standby-Datenbanken (siehe Kapitel 12.5) erforderlich. Auch ein Ausfall des kompletten Storage-Arrays oder andere K-Fall-Szenarien wären fatal. Entsprechende Lösungen mit Desastertoleranz werden weiter unten diskutiert.

- Die Gesamtstrategie lautet: RAID 1+0, nicht RAID 0+1. Also: zuerst spiegeln, dann stripen. Es werden zunächst Spiegelpaare gebildet, anschließend wird aus diesen Spiegelpaaren ein Stripe-Set mit einer Stripe-Größe von 1 MB gebaut. Der entscheidende Vorteil liegt darin, dass ein RAID 1+0-Array auch den gleichzeitigen Ausfall zweier oder mehrerer Festplatten verkraftet, solange diese nicht zufällig zu demselben Spiegelpaar gehören. Die Wahrscheinlichkeit dafür ist relativ gering (invers zur Größe der Stripe-Sets). Bei einem RAID 0+1-Array dagegen ist ein Mehrfachausfall fatal, sobald die betreffenden Festplatten zu demselben Stripe-Set gehören. Bei Einfachspiegelung beträgt die Wahrscheinlichkeit dafür schon 50 %.

Ein solches RAID-Array kann wahlweise mit den Mitteln des Storage-Arrays oder aber mit Oracle-eigenen Mitteln erzeugt werden. Im letzteren Fall präsentiert das Storage-System einfach eine flache Ansammlung von Festplatten, die mithilfe von Oracles *ASM (Automatic Storage Management)* in ein RAID-Array konfiguriert werden. Dies wird in Abschnitt 3.4 genauer beschrieben.

Eine weiteres Quentchen Hochverfügbarkeit kann durch den prophylaktischen Einbau von Reserveplatten („Spare Disk Drives“) erreicht werden, die als Hot Standby automatisch einspringen, wenn eine Platte ausfällt, d. h., es findet ein automatisches Remirroring statt, das angeschlagene Spiegelpaar wird also wieder vervollständigt. Der Ausfall eines kompletten Spiegelpaars wird dadurch noch ein wenig unwahrscheinlicher.

Split Mirror-Technologien

Einige Storage-Hersteller bieten sogenannte Split Mirror-Technologien an. Im Datenbankumfeld bietet dies vor allem bei Hochlast-Datenbanken Vorteile, um ohne eine zusätzliche Belastung des Datenbank-Servers

- Sicherungen der Datenbank durchzuführen
- Kopien der Datenbank abzuziehen, aus denen dann beispielsweise Data-Warehouse-, Test- oder Entwicklungsdatenbanken befüllt werden.

Das Verfahren besteht darin, dass entweder eine weitere Spiegelplatte benutzt oder die im Rahmen von RAID 1 ohnehin vorhandene Spiegelplatte hierfür „missbraucht“ wird. Im normalen Betrieb läuft die Spiegelplatte mit synchronem Datenbestand. Anstatt nun wie bei einer herkömmlichen Datenbanksicherung Dateien über das Betriebssystem des Datenbank-Servers zu kopieren, wird die Synchronisierung der Spiegelplatte unterbrochen, der Spiegel wird – wie man sagt – „aufgebrochen“ (Split Mirror).

Die abgetrennten Spiegelplatten werden in das Dateisystem eines anderen Servers gemountet und von dort z. B. auf Band gesichert. Oder es wird ein Recovery durchgeführt, um in einen konsistenten Zustand zu kommen, so dass die Datenbank auf dem zweiten Server geöffnet werden kann, um damit ein Data Warehouse zu pflegen.

Anschließend können die Spiegelplatten wieder an den Originalserver zurückgehängt werden, wo sie vom Storage-System automatisch wieder resynchronisiert werden, sie werden sozusagen wieder neu verspiegelt („Resilvering“).

In diese Kategorie fallen zum Beispiel die sogenannten BCVs (Business Continuance Volumes[14]) von EMC², Hitachi ShadowImage oder auch StorageWorks BusinessCopy.

Desastertoleranz

Um sich gegen den Ausfall eines kompletten Storage-Arrays oder gar Rechenzentrums abzusichern, wird häufig eine Storage-basierende Replikation eingesetzt. Dabei werden das gesamte Storage-Array oder einzelne Volumes in einen mehr oder weniger weit entfernten Standby-Standort repliziert. Um den Server nicht zu belasten, läuft diese Replikation rein storage-seitig ab. Entsprechende Technologien finden sich bei den einschlägigen Herstellern z. B. als EMC² SRDF, NetApp SyncMirror und MetroCluster, HP StorageWorks XP Continuous Access oder auch HDS TrueCopy.

Grundsätzlich gibt es zwei Replikationsmodi:

Synchrone Replikation bietet die beste Absicherung gegen Datenverlust, außerdem sind die Daten auf der Standby-Seite sofort verfügbar. Damit werden also sehr gute Werte für RPO und RTO[15] erreicht. Allerdings führt dies gerade bei größeren Entfernungen zwischen den beiden Standorten zu höheren Latenzzeiten, hat also einen negativen Einfluss auf die Storage-Performance, da ja jeder Schreibvorgang erst auch von der Standby-Seite bestätigt werden muss, bevor die Bestätigung an das Betriebssystem zurückgegeben werden kann. Aus demselben Grund schlagen Storage-Probleme auf der Standby-Seite auch häufig auf die Primärseite durch.

Asynchrone Replikation liefert leicht schlechtere Werte bei RPO und RTO, dafür ist der Performance-Einfluss auf die Applikation deutlich geringer, in den meisten Fällen ist praktisch keine Beeinflussung des Servers spürbar.

3.4 Oracle Automatic Storage Management

Mit Oracle 10g hat Oracle ein eigenes Dateisystem namens *Automatic Storage Management (ASM)* eingeführt, das unabhängig vom Betriebssystem (Unix, Linux und MS-Win-

14 Im Gegensatz zur Definition von „Business Continuity“ in Abschnitt 3.3.1 verweist der Begriff hier nicht auf K-Fall-Szenarien.

15 Recovery Point Objective und Recovery Time Objective (siehe Abschnitt 3.3.3)

dows) eine optimale Implementierung für den Betrieb von Oracle-Datenbanken darstellen soll. Neben der Möglichkeit der Mehrfachspiegelung werden die Dateisysteme über alle verfügbaren Platten gestripet und entsprechen damit dem von Oracle seit Längerem propagierten SAME-Prinzip (Stripe and Mirror Everything).

Zunächst ein paar Worte zur Historie: Bis zur Datenbankversion 9i wurden für RAC-Datenbanken ausschließlich die Benutzung von Raw Devices oder die Verwendung bestimmter Cluster-Dateisysteme unterstützt, die ihrerseits mit deutlichen Lizenzkosten verbunden waren.

Mit Version 9i hat Oracle dann das Oracle Cluster File System (OCFS) eingeführt (siehe Abschnitt 3.6.2). Dieses ist betriebssystemunabhängig und kann als separates Produkt eingesetzt werden, die Verzeichnisse sind aber z. B. unter MS-Windows als normale Dateisysteme zu erkennen, und somit besteht die Gefahr, dass diese Verzeichnisse durch Unachtsamkeit zerstört und damit die Datenbank korrumpiert wird. Außerdem steht OCFS nur für den Betrieb von Real Application Clusters zur Verfügung.

Mit Version 10g wurde daher ASM als eine unabhängige Storage-Technik geschaffen, die für alle Unix-, Linux- und Windows-basierten Oracle-Datenbanken ab der Datenbankversion 10g zur Verfügung steht und für das Betriebssystem unsichtbar ist.

Wichtig zu wissen ist, dass ASM nicht nur für RAC-Datenbanken, sondern für alle Datenbanken benutzbar ist. Und es gibt einen deutlichen Trend, ASM einfach für alle Datenbanken zu benutzen. Dafür spricht eine Reihe von Gründen:

- ASM bietet eine hervorragende Performance, da es jeden Dateisystem-Cache umgeht und sich somit ausschließlich die Datenbankinstanz um das Caching kümmert.
- Das automatische Rebalancing von ASM sorgt dafür, dass das Hinzufügen oder Entfernen von Platten extrem einfach und ohne weitere Eingriffe vonstatten geht.
- Selbst Migrationen wie beispielsweise ein Umzug auf ein anderes Storage oder einen komplett neuen Storage-Turm lassen sich per ASM-Rebalancing sehr einfach und vor allem ohne Auszeit durchführen, indem einfach sukzessive alte Platten entfernt und neue Platten eingehängt und jedes Mal das Rebelancing abgewartet wird (diese Technik ist in Kapitel 14.3.1 genau beschrieben).
- Da ASM implizit Oracle Managed Files (OMF) benutzt, sind das Erstellen und Verwalten von Tablespaces (Lokation, Größe, Autoextend-Eigenschaften) extrem einfach. Die entsprechenden Syntaxklauseln können wegfallen. Mit sogenannten Bigfile-Tablespace lässt sich sogar eine 1:1-Zuordnung zwischen Tablespaces und Datendateien einrichten.
- Die Kommunikations-Schnittstelle zwischen Storage- und Datenbankadministration kann ebenfalls sehr einfach und klar gehalten werden. Das Storage-Team braucht ausschließlich entsprechende Platten zur Verfügung zu stellen. Je nach Aufgabenverteilung hängt jemand aus dem Storage- oder jemand aus dem Datenbankteam diese Platten in ASM ein. Mehr ist nicht zu tun.
- Seit der Datenbankversion 11.2 können nicht nur Datenbankdateien, sondern auch beliebige weitere Dateien in ASM gespeichert werden, indem man das sogenannte ACFS (ASM Cluster File System) nutzt.

Eine Einschränkung sei hier genannt: Es ist meistens nicht zu empfehlen, hinsichtlich Spiegelung (RAID 1) komplett auf ASM zu setzen. Der Grund dafür ist, dass ein per Plat-

tenturm implementiertes RAID meistens über bessere Alarmierungsfunktionen verfügt, um automatisch den Hersteller zu informieren, der dann gegebenenfalls einen Plattentausch veranlasst, eine neue Spare-Platte einsetzt etc. Verfügt die Hardware über solche Überwachungs- und Alarmierungsfunktionen, sollte ASM-Spiegelung also entweder nicht oder zusätzlich zur Hardware-Spiegelung eingerichtet werden.

Da ASM eine Schnittstelle zum Storage-Bereich darstellt, ist die konkrete Konfiguration teilweise spezifisch für das verwendete Betriebssystem. Diese spezifischen Eigenschaften beziehen sich auf das Bereitstellen der Storage-Devices sowie die Zuordnung dieser Devices zu ASM-Diskgruppen, wofür unter MS-Windows und Linux sogar eigene Werkzeuge existieren. Alle anderen Einstellungen sind plattform-unabhängig. In den folgenden Abschnitten werden die einschlägigen Schritte genauer beschrieben.

3.4.1 ASM-Storage-Devices konfigurieren

Die zu verwendenden Platten sollten zuvor partitioniert werden, wobei pro physischer Platte nicht mehr als eine Partition von der ASM-Instanz genutzt werden sollte. Es wird empfohlen, eine einzige Partition über die gesamte Platte zu erstellen. Diese Partition wird später als ASM-Disk benutzt.

Entscheidende Voraussetzungen für ASM sind:

- die persistente Zuordnung der benutzten physischen Platten zu Device-Namen (auch über einen Reboot des Servers hinweg),
- persistente Zugriffsrechte auf diese Devices (ebenfalls über einen Reboot des Servers hinweg)
- sowie identische Device-Namen über alle Cluster-Knoten.

Auf Unix-Systemen sind diese Anforderungen weniger problematisch zu erfüllen als auf Linux- und MS-Windows-Servern. Für die letzteren beiden gibt es daher eigene Werkzeuge, um dies zu erreichen. In jedem Falle (auch für Unix-Plattformen) empfiehlt sich ein Blick in die jeweilige Dokumentation, in diesem Falle den *Grid Infrastructure Installation Guide* für die jeweilige Plattform.

Für alle Plattformen außer Windows müssen die Zugriffsrechte auf den Devices so gesetzt sein, dass als Eigentümer der Software Owner der Grid Infrastructure (standardmäßig der Benutzer `grid`) gesetzt ist. Die Gruppe muss der OSASM-Gruppe entsprechen (standardmäßig `asmadmin`). Für beide müssen Lese- und Schreibrechte gesetzt sein, ansonsten sollten keinerlei Rechte bestehen, also zum Beispiel:

```
$ chown grid:asmadmin /dev/rhdisk1
$ chmod 660 /dev/rhdisk1
```

Diese Kommandos müssen für Unix-Plattformen auf allen beteiligten Knoten einmalig ausgeführt werden.

Spezialfall Linux

Bei Linux-Systemen ist die Situation etwas schwieriger, da bei jedem Neustart des Servers Eigentümer, Gruppe und Zugriffsrechte wieder zurückgesetzt werden. Hier wird – wie weiter unten beschrieben – entweder mit Standardwerkzeugen wie `udev` oder `devlabel` oder mit dem Oracle-eigenen Werkzeug ASMLib gearbeitet.

Bei ASMLib handelt es sich im Wesentlichen um einen kombinierten Daemon sowie ein Kommandozeilenwerkzeug namens `oracleasm`, das ASM-Platten „stempelt“, also mit persistenten, ASM-spezifischen Plattennamen versieht sowie für das Setzen der Permissions und Ownerships beim Systemstart sorgt.

Wir empfehlen, ASMLib zu nutzen, da es einige spezifische Vorteile bietet:

- Es erlaubt die Vergabe sprechender Device-Namen wie beispielsweise `MYORADB DISK05`.
- Mit der Standardeinstellung der ASM-Instanz werden automatisch genau die via ASMLib konfigurierten Platten gescannt. Weitere Platten werden nicht benötigt, der ASM-Serverparameter `asm_diskstring` muss daher nicht gesetzt oder gepflegt werden.
- Das Kommandozeilenwerkzeug `oracleasm` liefert einige nützliche Funktionen, beispielsweise das Auflisten aller für ASM verwendeten Devices oder deren Zuordnung zu physischen Devices.
- Da ASMLib seine Konfiguration zentral im Platten-Header ablegt, ist keine Synchronisation über alle Knoten notwendig. Wird ein neuer Server dem Cluster hinzugefügt oder muss ein Server neu installiert werden, ist es nicht nötig, irgendwelche Konfigurationsdateien anzupassen oder zu übertragen. Vielmehr genügt auf dem neuen Knoten ein einziger Aufruf der Form:

  ```
  $ service oracleasm scandisks
  ```

Die Installation und Benutzung von ASMlib ist in Kapitel 4.2.1 genauer beschrieben.

Spezialfall Windows

Unter Windows müssen die Partitionen vorhanden, dürfen jedoch nicht mit Betriebssystemmitteln formatiert worden sein. Bevor sie als Diskgruppen benutzt werden können, müssen sie mit einem der Werkzeuge *asmtool* (Kommandozeile) oder *asmtoolg* (grafische Oberfläche) vorformatiert werden. Dies wird als „Stamping“ bezeichnet.

Die so markierten Partitionen stehen jetzt für das Betriebssystem nicht mehr zur Verfügung. Um sie wieder normal nutzbar zu machen, müssen Sie wiederum das ASM-Tool aufrufen und die Markierungen löschen. Abbildung 3.3 zeigt das grafische ASM-Tool mit der Anzeige der möglichen Partitionen und deren Größe.

Sobald die Partitionen für ASM genutzt werden, haben sie zusätzlich einen *ASM Link Name* (siehe Abbildung 3.4), der sich aus `ORCLDISK`, dem eingegebenen Präfix und einer eindeutigen Nummer zusammensetzt.

3.4.2 ASM-Instanzen erstellen und konfigurieren

Seit der Datenbankversion 11gR2 wird eine ASM-Instanz normalerweise bereits im Rahmen der Grid-Infrastructure-Installation erstellt (siehe Kapitel 4.2.1), es sei denn, man wählt dort als Installationsart den Punkt „Software Only“.

In älteren Datenbankversion bietet der Database Configuration Assistant (DBCA) einen Auswahlpunkt zur Erstellung einer ASM-Instanz (siehe Abbildung 3.5).

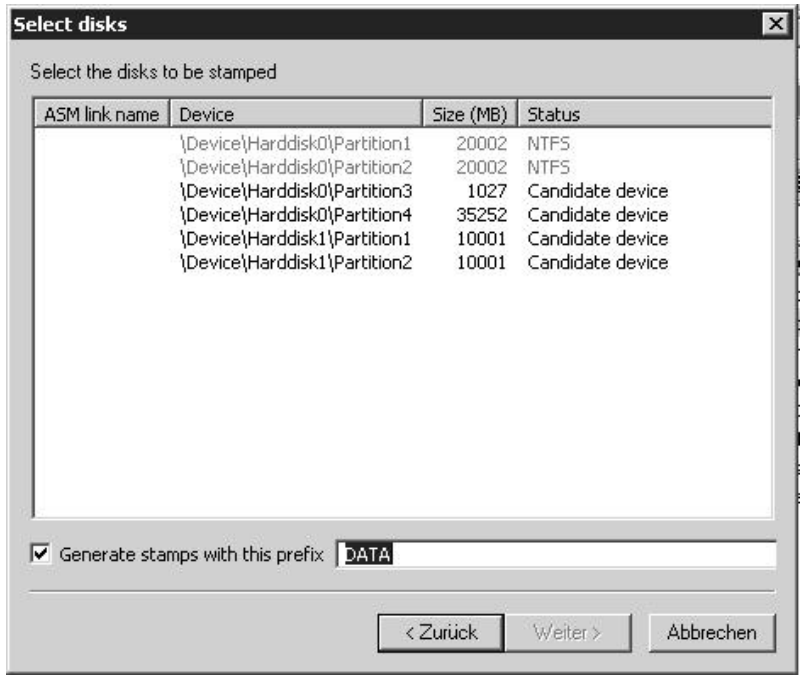


Abbildung 3.3: Stamping der Windows-Partitionen als Vorbereitung für die Nutzung mit ASM

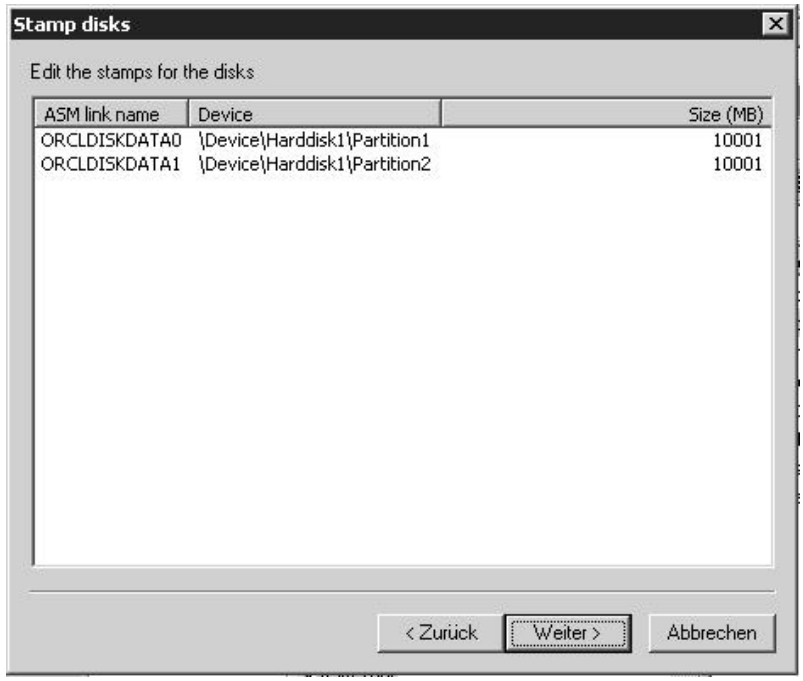


Abbildung 3.4: Für ASM markierte Windows-Partitionen

In den meisten Fällen genügt diese Standardinstallation auch vollauf, da es bei einer ASM-Instanz nichts gibt, was man nicht im Nachhinein einfach umkonfigurieren könnte. Auch gibt es normalerweise keinen Grund, mehrere ASM-Instanzen auf einem Server zu betreiben.

Sollte man dennoch einmal eine ASM-Instanz manuell aufsetzen müssen, seien hier die wichtigsten Serverparameter genannt:

Der einzige unbedingt benötigte Parameter lautet `instance_type`, der auf den Wert `asm` gesetzt sein muss. Dieser Parameter macht den Unterschied zwischen einer normalen Datenbankinstanz und einer ASM-Instanz aus.

Der Standardname für eine ASM-Instanz lautet `+ASM`. Installiert man die Grid Infrastructure 11.2 mit ASM für einen einzelnen Server oder erstellt man mit dem DBCA in älteren Versionen eine ASM-Instanz in einer Umgebung ohne Clusterware, so wird automatisch dieser Instanzname gewählt. In Cluster-Umgebungen mit mehreren Servern werden die ASM-Instanzen dagegen genau wie Datenbankinstanzen in der Art `+ASM1`, `+ASM2` etc. durchnummeriert. In jedem Falle spiegelt dies der Serverparameter `instance_name` der ASM-Instanz wider.

Mit dem Parameter `asm_diskgroups` kann eine Liste von Diskgruppen angegeben werden, die gemountet werden sollen, zum Beispiel:

```
asm_diskgroups='DATA, FRA'
```

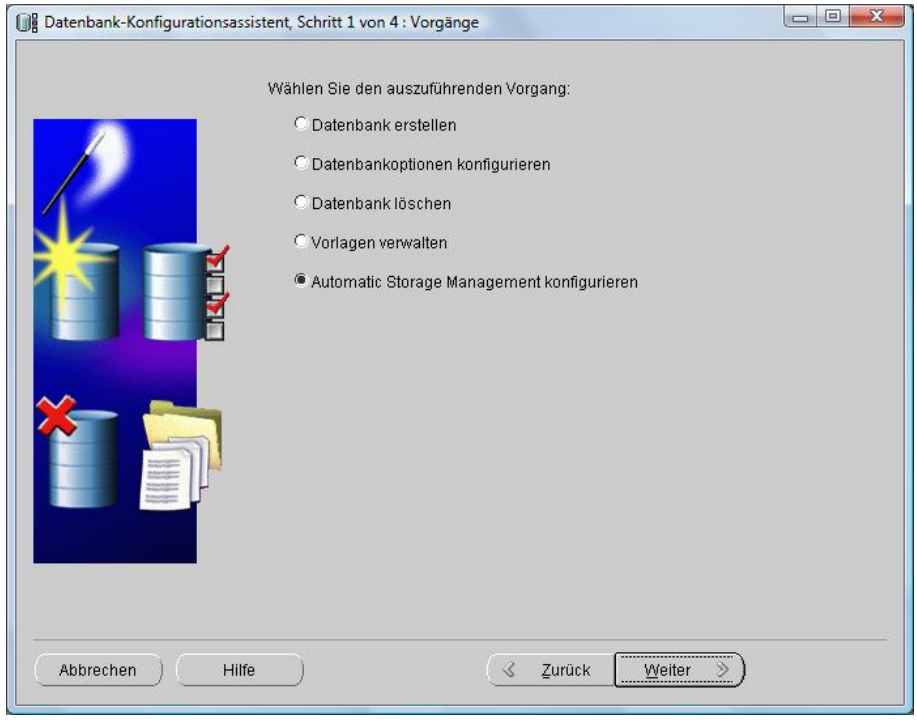


Abbildung 3.5: Erstellung einer ASM-Instanz in Oracle 10gR1, 10gR2 und 11gR1

Normalerweise wird der Parameterwert beim Erstellen bzw. Löschen einer Diskgruppe automatisch angepasst, so dass nicht manuell eingegriffen werden muss.

Gegebenenfalls muss mit dem Parameter `asm_diskstring` eine Liste der zu durchsuchenden Platten-Devices angegeben werden. Dasselbe gilt unter Linux, wenn die ASMLib nicht verwendet wird, zum Beispiel:

```
asm_diskstring='/dev/xvdb*,/dev/xvdc*'
```

Arbeitet man unter Windows oder mit ASMLib unter Linux, kann man den Parameter einfach frei lassen. In diesem Fall werden einfach alle Devices durchsucht, die mit ASMLib bzw. mit dem Windows-Werkzeug `asmtool` gestempelt wurden. Dies liegt daran, dass in diesem Fall betriebssystemabhängige Standardwerte greifen. Folgende Tabelle zeigt die Standardwerte auf den entsprechenden Plattformen:

Betriebssystem	Standardwert
Solaris	`/dev/rdsk/*`
Windows	`\\.\orcldisk*`
Linux	`/dev/raw/*`
Linux mit ASMLib	`/dev/oracleasm/disks/*, ORCL:*`
HPUX	`/dev/rdsk/*`
HP Tru64	`/dev/rdisk/*`
AIX	`/dev/*`

Die SGA einer ASM-Instanz wird seit der Version 11gR1 standardmäßig mithilfe des Serverparameters `memory_target` auf ca. 256 MB dimensioniert, was vollkommen ausreichend ist. Für ältere Versionen gelten folgende Empfehlungen für die ASM-Instanz:

```
shared_pool_size=128M
large_pool_size=12M
db_cache_size=64M
processes=25+15*Anzahl zugreifender Datenbankinstanzen
```

Genau wie bei Datenbankinstanzen auch, muss der Parameter `remote_login_passwordfile` gesetzt sein, um eine Remote-Anmeldung mit SYSDBA-Rolle an die ASM-Instanz zu ermöglichen. Da andere Logins ohnehin nicht möglich sind, ist dies bei ASM-Instanzen besonders relevant und für eine Remote-Administration unverzichtbar:

```
remote_login_passwordfile=EXCLUSIVE
```

Achtung: Bei der Datenbankversion 10g ist eine Remote-Anmeldung standardmäßig dennoch nicht möglich. Dies liegt daran, dass eine ASM-Instanz nie geöffnet ist, sondern sich im NOMOUNT- oder MOUNT-Status befindet. Der Listener blockiert in diesem Falle sämtliche Remote-Anmeldungen und kann – wie in Kapitel 9.1.2 beschrieben – nur mit der Klausel `UR=A` überredet werden. Als Administrator einer ASM-Instanz der Version 10g sollte man also einen TNS-Eintrag wie den folgenden benutzen:

```
ASM10G =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = ora10g)(PORT = 1521))
    (CONNECT_DATA =
      (SERVICE_NAME = +ASM)
      (UR=A)
    )
  )
```

In der Version 11g ist dies nicht mehr notwendig.

Es gibt übrigens keinen Serverparameter `compatible` für eine ASM-Instanz. Stattdessen werden die Versionskompatibilitäten der ASM-Metadaten sowie der zugreifenden Datenbankinstanzen pro Diskgruppe geregelt. Dies ist in Abschnitt 3.4.3 genauer beschrieben.

3.4.3 ASM-Storage konfigurieren

Ist die ASM-Instanz einmal gestartet, müssen als Nächstes sogenannte Diskgruppen definiert werden. Diese bilden, ähnlich wie Tablespaces, eine Möglichkeit der administrativen Trennung verschiedener Teile des ASM-Storage. Auf die Frage, wie viele und welche Diskgruppen es geben sollte, gibt es keine pauschale Antwort. Allerdings sollte man sich an folgende Richtlinien halten:

- Definitiv sollte es mindestens zwei Diskgruppen geben, nämlich eine für den produktiven Teil der Datenbank, also für Daten- und Temporärdateien sowie die Serverparameterdatei, und eine weitere für alle sicherungsrelavanten Dateien, also Backup-Sets, archivierte Redologdateien und gegebenenfalls Flashback-Logs. Es empfiehlt sich, Online-Redologdateien und Kontrolldateien zu spiegeln und auf beiden Diskgruppen zu halten.

  Die erstere Diskgruppe hat häufig „DATA“ als Namensbestandteil, die letztere „FRA“ (Fast Recovery Area).

- Nutzt man das in Abschnitt 3.5 beschriebene ASM-Cluster-Dateisystem, so empfiehlt sich auch dafür eine eigene Diskgruppe.

- Diskgruppen bilden auch eine Abstraktionsebene für unterschiedliche physische Storage-Implementierungen. Möchte man zum Beispiel alte Daten auf kostengünstigerem Storage unterbringen, so lassen sich, wie in Kapitel 13.6 beschrieben, alte Datengenerationen in dedizierte Tablespaces verschieben, die auf entsprechendem Storage liegen. Die ASM-Diskgruppe leistet hier eine sprechende Zuordnung (z. B. eine Diskgruppe DATAOLD) und vermittelt an der Schnittstelle zwischen Datenbank- und Storage-Administrator.

- Die weiter unten beschriebenen Versionskompatibilitäten werden pro ASM-Diskgruppe eingestellt. Hat man also eine gemischte Umgebung von 10g- und 11g-Datenbanken, die auf eine gemeinsame 11g-ASM-Instanz zugreifen sollen, wobei die 11g-Datenbanken auch von neuen ASM-Features der Version 11g profitieren sollen, benötigt man unterschiedliche Diskgruppen mit jeweils passenden Kompatibilitätseinstellungen.

Die ASM-Diskgruppe spielt aus Datenbanksicht ausschließlich für die Platzierung der Datenbankdateien eine Rolle. Bei der Erstellung einer Tabelle entscheidet also die Wahl des Tablespaces indirekt auch über die Wahl der Diskgruppe und damit über die storage-seitige Platzierung. Tablespaces mit Datendateien in unterschiedlichen ASM-Diskgruppen sollte man vermeiden, da die Übersicht leidet.

Das Erstellen und Verwalten von ASM-Diskgruppen kann je nach Präferenz oder Aufgabenteilung über drei verschiedene Werkzeuge durchgeführt werden. So wird ein DBA, der ASM mitbetreut, womöglich den Weg via SQL*Plus favorisieren, während ein Systemadministrator eher mit den anderen Werkzeugen besser zurechtkommt. In allen drei Fällen sollte der Aufruf als Eigentümer der Grid Infrastructure, also z. B. als Benutzer `grid`, geschehen:

- Mithilfe von SQL*Plus, wobei eine Anmeldung an die ASM-Instanz mit SYSASM-Rolle (SYSDBA-Rolle in der Datenbankversion 10g) notwendig ist:

```
$ sqlplus / as sysasm
$ sqlplus sys@asm11g as sysasm
$ sqlplus sys@asm10g as sysdba
```

- Mit dem Kommandozeilenwerkzeug `asmcmd`, das wahlweise interaktiv genutzt oder direkt mit einem entsprechenden Kommando aufgerufen werden kann, beispielsweise:

```
$ asmcmd -p
$ asmcmd ls DATA
$ asmcmd ls DATA > DATA_contents.txt
```

  `asmcmd` führt im Hintergrund eine lokale Anmeldung mit SYSASM- bzw. SYSDBA-Privileg an die ASM-Instanz durch und übersetzt Kommandos, die wie typische Unix-Kommandos aussehen, in SQL-Syntax.

  Eine vollständige Kommandoreferenz findet sich in der Oracle-Dokumentation *Oracle Database Storage Administrator's Guide*. Tabelle 3.1 gibt einen Überblick. `asmcmd` gibt es seit der Version 10gR2, mit der Version 11gR2 wurde sein Funktionsumfang deutlich erweitert, um alle ASM-relevanten Aufgaben auch ohne SQL-Kenntnisse abbilden zu können.

- Mit dem in der Version 11.2 neu eingeführten ASM Configuration Assistant (ASMCA) als grafischem Werkzeug. Die ausführbare Datei `asmca` befindet sich im Unterverzeichnis `bin` des Grid-Infrastructure-Homes. Zur Zeit ist der Funktionsumfang noch etwas eingeschränkt und deckt vor allem Funktionen im Bereich ACFS ab (siehe Abschnitt 3.5).

ASM-Diskgruppen erstellen

Das folgende Kommando erstellt eine ASM-Diskgruppe namens DATA mit Hilfe des einschlägigen SQL-Befehls:

```
$ sqlplus / as sysasm

SQL> CREATE DISKGROUP data
     NORMAL REDUNDANCY
     FAILGROUP data_fg1 DISK 'ORCL:ASMDISK3' NAME data_fg1_disk1
     FAILGROUP data_fg2 DISK 'ORCL:ASMDISK4' NAME data_fg2_disk1
     ATTRIBUTE 'COMPATIBLE.ASM'   = '11.2',
               'COMPATIBLE.RDBMS' = '10.1';
```

Das `asmcmd`-Werkzeug erlaubt dasselbe mit Hilfe des Befehls `mkdg`. Alle notwendigen Angaben müssen in Form einer kleinen XML-Datei mitgegeben werden:

```
create_dg_data.xml:
<dg name="data" redundancy="normal">
  <fg name="data_fg1">
    <dsk string="ORCL:ASMDISK3" name="data_fg1_disk1"/>
  </fg>
  <fg name="data_fg2">
    <dsk string="ORCL:ASMDISK4" name="data_fg2_disk1"/>
  </fg>
  <a name="compatible.asm" value="11.2"/>
```

```
  <a name="compatible.rdbms" value="10.1"/>
</dg>

ASMCMD> mkdg create_dg_data.xml
```

Dateien in ASM	ASM-Instanz	ASM-Diskgruppen	ASM-Templates	Zugriffs-Rechte	ACFS Volumes
cd**	dsget	chdg	chtmpl	chgrp	volcreate
cp*	dsset	chkdg	lstmpl	chmod	voldelete
du**	lsct**	dropdg	mktmpl	chown	voldisable
find**	lsop	iostat	rmtmpl	groups	volenable
ls**	lspwusr	lsattr		grpmod	volinfo
lsof	orapwusr	lsdg**		lsgrp	volresize
mkalias**	shutdown	lsdsk*		lsusr	volset
mkdir**	spbackup	lsod		mkgrp	volstat
pwd**	spcopy	md_backup*		mkusr	
rm**	spget	md_restore*		passwd	
rmalias**	spmove	mkdg		rmgrp	
	spset	mount		rmusr	
	startup	offline			
		online			
		rebal			
		remap*			
		setattr			
		umount			

*: bereits in 11gR1 verfügbar, ** bereits in 10gR2 verfügbar

Tabelle 3.1: Kommandos im asmcmd-Werkzeug

Im ASMCA kann man Diskgruppen nach einem Klick auf den *Create*-Button im Reiter *Disk Groups* erstellen. Abbildung 3.6 zeigt das entsprechende Dialogfenster.

Wenn im ASMCA die Disks nicht angezeigt werden oder man beim Anlegen Fehlermeldungen ähnlich den folgenden erhält:

```
ERROR at line 1:
ORA-15018: diskgroup cannot be created
ORA-15031: disk specification '/dev/xvdc2' matches no disks
ORA-15025: could not open disk '/dev/xvdc2'
ORA-15056: additional error message
```

so kann das unterschiedliche Gründe haben. Häufige Ursachen sind:

- Unter MS-Windows wurden die Devices nicht mit dem Werkzeug `asmtool` oder `asmtoolg` vorformatiert (siehe Abschnitt 3.4.1).
- Unter Linux wurden die Devices nicht via ASMLib erfasst (siehe Abschnitt 3.4.1 sowie Kapitel 4.2.1).
- Unter Unix sind Eigentümer, Gruppe oder Berechtigungen auf den Disk-Devices nicht richtig gesetzt (siehe Abschnitt 3.4.1).
- Der Pfad, unter dem die Devices gesucht werden sollen (Serverparameter `asm_disk_string`), ist gesetzt, zeigt aber auf einen anderen Bereich.

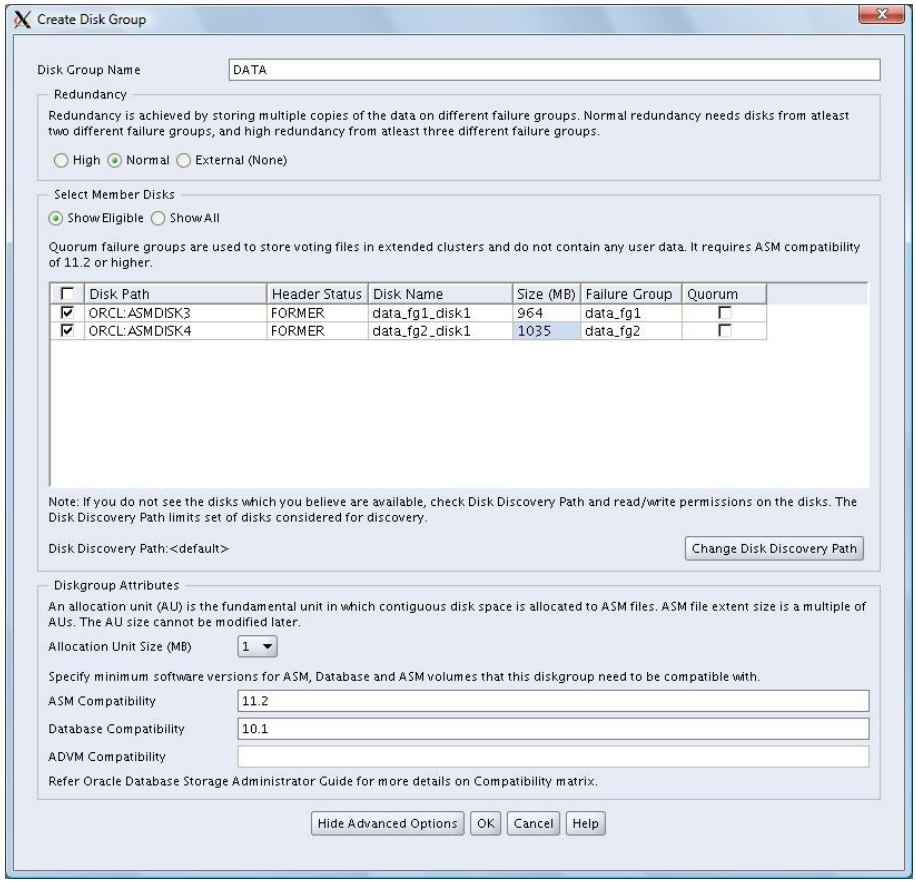


Abbildung 3.6: Erstellen einer ASM-Diskgruppe mit dem ASMCA

In allen drei Beispielen wird dieselbe Diskgruppe DATA erstellt, die aus zwei ASM-Disks besteht. Alle Daten innerhalb dieser Diskgruppe werden einfach gespiegelt (RAID 1). Dies wird durch die Redundanzeinstellung festgelegt:

NORMAL REDUNDANCY ist die Standardeinstellung und bewirkt eine mindestens einfache Spiegelung, d. h., alle Daten werden doppelt gehalten. Insbesondere muss es mindestens zwei sogenannte Failgroups geben (siehe unten). Mit Hilfe von ASM-Templates können aber bestimmte Dateien auch doppelt gespiegelt, also dreifach vorgehalten werden. Dies ist zum Beispiel standardmäßig für die Kontrolldateien der Fall.

HIGH REDUNDANCY erzwingt eine doppelte Spiegelung aller Daten, alle Daten werden also dreifach gespeichert. Es muss daher auch mindestens drei Failgroups geben (siehe unten).

EXTERNAL REDUNDANCY führt keine Spiegelung der Daten durch, jedenfalls nicht durch die ASM-Instanz. In diesem Falle sollten die ASM-Disks also bereits durch das darunterliegende Storage-System gespiegelt sein.

Die Redundanzeinstellung lässt sich nachträglich nicht mehr ändern. Möchte man also beispielsweise nachträglich alle Daten spiegeln, muss man zunächst eine neue Diskgruppe mit der gewünschten Redundanzeinstellung erzeugen und alle ASM-Dateien in diese neue Diskgruppe verschieben (siehe dazu Abschnitt 3.4.6).

Failgroups

Wichtig für eine effektive Spiegelung ist die Information, ob und inwieweit es physikalische Abhängigkeiten zwischen den ASM-Disks gibt. Hängen beispielsweise verschiedene ASM-Disks an demselben SCSI-Controller, sind bei einem Ausfall dieses Controllers alle Disks betroffen. Eine Spiegelung innerhalb dieser Teilgruppe wäre also riskant.

Wegen dieses Umstands kann jeder ASM-Disk eine sogenannte *Failgroup* zugeordnet werden. Disks, deren Ausfall auf die beschriebene Weise korreliert ist, sollten derselben Failgroup angehören. Auf die konkreten Namen der Failgroups (im obigen Beispiel jeweils `data_fg1` und `data_fg2`) kommt es nicht an.

Die ASM-Instanz sorgt nun dafür, dass die im Rahmen der Spiegelung erzeugten Datenkopien stets auf unterschiedlichen Failgroups liegen, damit beim Ausfall einer Platte auch tatsächlich nur eine einzige Kopie betroffen ist. Auf welchen Platten die Daten konkret zu liegen kommen, kann nicht beeinflusst werden. Da die Abbildung von Daten auf Failgroups nicht statisch ist, können ohne Probleme auch mehr als zwei bzw. drei Failgroups definiert sein.

Achtung: Gibt man keine Failgroup an, wird jede ASM-Disk implizit einer eigenen Failgroup zugeordnet, die so heißt wie die Disk selbst. In diesem Fall geht die ASM-Instanz also davon aus, dass es keinerlei Korrelationen zwischen den Disks gibt, also von einem sehr optimistischen Szenario! Man sollte dies genau prüfen und besser entsprechende Failgroup-Angaben machen.

Auch wenn die ASM-Instanz dies nicht erzwingt, so sollten die Failgroups innerhalb einer Diskgruppe doch möglichst gleich groß sein, vor allem wenn man mit der Minimalanzahl von Failgroups arbeitet, da ansonsten die kleinste Failgroup den nutzbaren Speicherplatz determiniert.

Führt die ASM-Instanz keinerlei Spiegelung durch (`EXTERNAL REDUNDANCY`), ist die Angabe einer Failgroup nicht erlaubt und auch nicht sinnvoll.

Jede Failgroup besteht also aus einer oder mehreren ASM-Disks. Diese Disks haben auf jeden Fall einen OS-spezifischen Pfad (im obigen Beispiel ASMLib-Namen, ansonsten Pfade in der Art `/dev/xyz`), der in der Spalte `path` der View `V$ASM_DISK` auftaucht. Ein optional zu vergebender, OS-unabhängiger, sprechender Name (`NAME`-Klausel im SQL-Kommando) kann in derselben View in der Spalte `name` eingesehen werden.

Achtung: Informationen über ASM-Disks und Diskgruppen können über zwei Paare von Views abgefragt werden:

- `V$ASM_DISK` und `V$ASM_DISKGROUP` sind naheliegend und für kleine ASM-Umgebungen (wenige Disks) auch nicht zu beanstanden. Für größere Umgebungen verursachen diese Views aber sehr viel Overhead, da bei jeder Abfrage eine Disk-Discovery durchgeführt wird. Monitoring-Werkzeuge, die regelmäßig ASM-Informationen abgreifen, sollten diese Views daher nicht benutzen. Hat man gerade etwas an einer Diskgruppe geändert, ist ein einmaliger Aufruf dieser Views dagegen sinnvoll, da somit ein „Refresh“ durchgeführt wird.

Die Pendants im `asmcmd`-Werkzeug lauten:

```
ASMCMD> lsdsk --discovery
ASMCMD> lsdg --discovery
```

- `V$ASM_DISK_STAT` und `V$ASM_DISKGROUP_STAT` sind gecachete und in diesem Sinne statische Varianten der beiden erstgenannten Views. Für regelmäßige Abfragen, gerade in großen Umgebungen, ist ihre Benutzung dringend zu empfehlen. Änderungen an der Konfiguration spiegeln sich aber erst nach einem Refresh durch Selektion der Views ohne `_STAT` wieder.

  Das Pendant im `asmcmd`-Werkzeug lautet:

  ```
  ASMCMD> lsdsk
  ASMCMD> lsdg
  ```

Versions-Kompatibilität

Die Kompatibilität zwischen unterschiedlichen Versionen von ASM-Metadaten, ASM-Instanzen und zugreifenden Datenbank-Instanzen wird seit der Version 11g auf Diskgruppen-Ebene geregelt.[16] Dabei gibt es bis zu drei unterschiedliche Diskgruppen-Attribute, die jeweils eine Versionsnummer aufnehmen:

- Das Attribut `COMPATIBLE.ASM` gibt an, von welcher Version eine ASM-Instanz, die auf diese Diskgruppe zugreifen möchte, mindestens sein muss. Dadurch beeinflusst sind insbesondere die Verwaltungsdaten der ASM-Instanz. Beispielsweise versteht erst eine ASM-Instanz der Version 11.2 ein ACFS-Dateisystem. Erstellt man eine Diskgruppe mit SQL*Plus oder `asmcmd`, lautet die Standardeinstellung 10.1. Alle neueren ASM-Fähigkeiten sind also nicht möglich. Bei einer Erstellung mittels ASMCA ist die Standardeinstellung dagegen 11.2. Mögliche Werte liegen zwischen 10.1 und 11.2.
- Mit `COMPATIBLE.RDBMS` wird festgelegt, welche Datenbankinstanzen auf die ASM-Instanz zugreifen können. Maßgeblich ist dabei nicht die Version der Datenbanksoftware, sondern der Wert des Serverparameters `compatible` der Datenbankinstanz. Mögliche Werte liegen zwischen 10.1 und 11.2. Mindestwert und Standardwert ist – auch bei einer Erstellung mittels ASMCA – der Wert 10.1. ASM-Instanzen der Version 11g können also standardmäßig von allen 10g- und 11g-Datenbanken genutzt werden.
- `COMPATIBLE.ADVM` schließlich wird nur im Zusammenhang mit ACFS benötigt (siehe Abschnitt 3.5). Mindestwert und gleichzeitig Standardwert ist 11.2.

Alle drei Versionsnummern bilden eine Einbahnstraße, sprich eine Erhöhung ist möglich, aber niemals ein Zurücksetzen auf einen kleineren Wert.

3.4.4 Rebalancing

Wie oben beschrieben, sorgt die ASM-Instanz automatisch für ein Striping sowie gegebenenfalls eine Spiegelung aller Daten. Beim Hinzufügen, Entfernen, Vergrößern oder

16 Dies ist anders als bei Datenbank-Instanzen selbst, wo der Serverparameter `compatible` dies steuert. In ASM-Instanzen der Version 10g gibt es keine Kompatibilitäts-Einstellungen, was nicht so sehr ins Gewicht fiel, da ASM ja auch mit der Version 10g erst eingeführt wurde.

Verkleinern einer ASM-Disk müssen nun die Daten neu verteilt werden. Dieser Prozess wird als *Rebalancing* bezeichnet.

Das Rebalancing einer Diskgruppe läuft normalerweise automatisch nach einer der oben genannten „Umbaumaßnahmen“ an. Es lässt sich aber auch manuell steuern und insbesondere mit einer Priorität versehen. Außerdem kann man beeinflussen, ob das Rebalancing synchron oder asynchron zur Kommandozeile ablaufen soll. Im ersteren Fall hängt die Kommandozeile, bis das Rebalancing abgeschlossen ist oder mit einer Fehlermeldung abbricht. Im letzteren Fall läuft der Prozess im Hintergrund weiter, wobei der Fortschritt oder eventuelle Abbrüche über eine View kontrolliert werden können.

Selbst wenn man die Session beendet, läuft ein angestoßenes Rebalancing weiter. Dies liegt daran, dass eigene Hintergrundprozesse mit den Namen RBAL bzw. ARBn dafür zuständig ist (siehe Kapitel 1.18.3).

Es ist ein gleichzeitig Rebalancing mehrerer Diskgruppen möglich, aber pro Diskgruppe kann zu jedem Zeitpunkt nur eine Rebalance-Operation stattfinden.

Die folgende Abfrage zeigt, welche Disks bereits in Verwendung sind und welche noch keiner Diskgruppe zugeordnet sind:

```
SQL> SELECT dg.name AS dg_name,
            d.name AS disk_name,
            d.path AS disk_path,
            d.failgroup AS disk_failgroup,
            d.mount_status AS disk_status
     FROM   v$asm_disk d, v$asm_diskgroup dg
     WHERE  d.group_number = dg.group_number(+);

DG_NAME         DISK_NAME       DISK_PATH       DISK_FAILGROUP  DISK_STATUS
--------------- --------------- --------------- --------------- ---------------
                                ORCL:ASMDISK3                   CLOSED
                                ORCL:ASMDISK4                   CLOSED
DATA            DATA_FG1_DISK1  ORCL:ASMDISK1   DATA_FG1        CACHED
DATA            DATA_FG2_DISK1  ORCL:ASMDISK2   DATA_FG2        CACHED
```

Im `asmcmd`-Werkzeug liefert folgendes Kommando ähnliche Informationen:

```
ASMCMD> lsdsk -p -k --discovery
```

Möchte man eine Disk aus der Diskgruppe DATA entfernen und dafür die beiden noch freien Disks einhängen, kann man dies in einem Schritt tun. Dies ist sogar besser als ein mehrschrittiges Vorgehen, da ansonsten nach jedem Einzelschritt ein Rebalancing stattfindet, während das folgende Kommando nur ein Rebalancing auslöst:

```
SQL> ALTER DISKGROUP data
     ADD FAILGROUP data_fg1 DISK 'ORCL:ASMDISK3' NAME data_fg1_disk3,
                            DISK 'ORCL:ASMDISK4' NAME data_fg1_disk4
     DROP DISK data_fg1_disk1
     REBALANCE POWER 5
     WAIT;
```

Das zugehörige `asmcmd`-Kommando benötigt wieder eine XML-Datei. Es wird in jedem Falle asynchron ausgeführt:

```
change_dg_data.xml
<chdg name="data" power="5">
```

```
  <drop>
    <dsk name="data_fg1_disk1"/>
  </drop>
  <add><fg name="data_fg1">
    <dsk string="ORCL:ASMDISK3" name="data_fg1_disk3"/>
      <dsk string="ORCL:ASMDISK4" name="data_fg1_disk4"/>
  </fg></add>
</chdg>

ASMCMD> chdg change_dg_data.xml
```

Wie man sieht, werden zwei ASM-Disks eingehängt und der Failgroup `DATA_FG1` zugeschlagen, gleichzeitig wird eine der alten Disks aus der Diskgruppe entfernt. Im Vergleich mit der obigen Abfrage sieht man auch, dass bereits eingehängte Disks stets mit ihrem Namen angesprochen werden (Spalte `name` in der View `V$ASM_DISK`), während neue Disks, die ja gerade erst mit einem Namen versehen werden, über ihren OS-spezifischen Pfad referenziert werden müssen (Spalte `path` in `V$ASM_DISK`).

Die `REBALANCE POWER`-Klausel ist optional und erlaubt eine Priorisierung der Rebalance-Operation. Der kleinste Wert ist 0 und steht für die niedrigste Priorisierung, sprich die auftretende I/O-Last ist minimal, das Rebalancing dauert aber relativ lange. Der höchstmögliche Wert ist 11 und sorgt für ein schnellstmögliches Rebalancing um den Preis einer deutlichen I/O-Last. Lässt man die Klausel, greift als Standardeinstellung der Wert des Serverparameters `asm_power_limit` der ASM-Instanz, der selber wiederum den Standardwert 1 hat, was also eine niedrige Priorität darstellt.

Die `WAIT`-Klausel bestimmt, dass die SQL*Plus-Konsole solange hängt, bis das Rebalancing abgeschlossen ist oder mit einer Fehlermeldung abbricht. Da das durchaus einige Minuten oder sogar Stunden dauern kann, empfiehlt sich meistens die Alternative `NOWAIT`, was auch dem Standardverhalten entspricht. Dabei erscheint der Kommandoprompt bereits nach wenigen Sekunden wieder, im Hintergrund läuft das Rebalancing aber asynchron weiter.

Hat man das Kommando mit `WAIT` gestartet, genügt ein Abbruch mit der üblichen Tastenkombination Ctrl-C, um in eine asynchrone Bearbeitung zu wechseln. Das Rebalancing läuft auch in diesem Fall einfach im Hintergrund weiter.

Eine Fortschritts- und Erfolgskontrolle ist mit der View `V$ASM_OPERATION` bzw. dem Befehl `lsop` im `asmcmd`-Werkzeug möglich:

```
SQL> SELECT * FROM v$asm_operation;

GROUP_  OPERA STAT POWER ACTUAL SOFAR EST_WORK EST_RATE   EST_MINUTES  ERROR_CODE
NUMBER
------  ----- ---- ----- ------ ----- -------- --------   -----------  ----------
     1  REBAL RUN      5      5    40      734    730            0

ASMCMD> lsop
Group_Name  Dsk_Num  State  Power
DATA        REBAL    RUN    5
```

Ist die View leer, wurde das Rebalancing bereits erfolgreich abgeschlossen. Ansonsten findet man – leider nur bei der Abfrage via SQL*Plus – in der Spalte `est_minutes` eine

Schätzung, wie lange es noch dauert oder in der Spalte `error_code` eine Information über aufgetretene Fehler, beispielsweise:

```
SQL> SELECT * FROM v$asm_operation;

GROUP_  OPERA STAT POWER ACTUAL SOFAR EST_WORK EST_RATE   EST_MINUTES  ERROR_CODE
NUMBER
------  ----- ---- ----- ------ ----- -------- --------   -----------  ----------
     1  REBAL ERRS     5                                               ORA-15041

ASMCMD> lsop
Group_Name  Dsk_Num  State  Power
DATA        REBAL    ERRS   5
```

Bei der synchronen Abarbeitung hängt die SQL*Plus-Kommandozeile im Fehlerfall einfach. Man sollte dann in einer zweiten Session die Fehlernummer abfragen und gegebenenfalls in der Oracle-Dokumentation nachschauen, welcher Fehlermeldung diese entspricht. Leider bekommt man die Fehlernummer nur mittels SQL*Plus geliefert, nicht via `asmcmd`. Eine typische Fehlermeldung ist die im obigen Beispiel:

```
ORA-15041: diskgroup "DATA" space exhausted
```

Dies kann beim Entfernen einer oder mehrerer Disks auftreten, auch in Kombination mit dem Hinzufügen neuer Disks. Das Entfernen konnte nicht durchgeführt werden, weil anschließend der Speicherplatz der Diskgruppe nicht mehr ausreicht. Wichtig zu wissen ist an dieser Stelle, dass die Operation zwar nicht erfolgreich beendet, aber auch nicht komplett abgebrochen wurde. Man hat nun zwei Möglichkeiten:

- Entweder fügt man der Diskgruppe zunächst weitere Disks hinzu. In diesem Fall wird ein neues Rebalancing gestartet. Anschließend versucht man erneut, die betreffende Disk zu entfernen.
- Oder man bricht das Entfernen der Disk mit dem folgenden Befehl komplett ab:

  ```
  SQL> ALTER DISKGROUP data UNDROP DISKS;
  ```

  Auch dieses Kommando löst erneut ein Rebalancing aus. Leider gibt es dazu kein Pendant im `asmcmd`-Werkzeug.

Das kombinierte Hinzufügen und Entfernen von Disks kann auch für Storage-Migrationen von Datenbank benutzt werden. Auf dieses Verfahren wird in Kapitel 14.3.1 näher eingegangen.

Ein Rebalancing kann auch jederzeit manuell angestoßen werden. Der einzige Grund hierfür ist aber, auf diese Weise die Priorität einer bereits laufenden Rebalancing-Operation zu ändern:

```
SQL> ALTER DISKGROUP data
     REBALANCE POWER 5
     WAIT;

ASMCMD> rebal --power 5 -w data
```

3.4.5 ASM-Storage nutzen

Eine Datenbankinstanz kann grundsätzlich ASM-Storage für einzelne oder für alle Datenbankdateien nutzen, wobei der Übersichtlichkeit halber Misch-Konstellationen vermieden werden sollten.

Achtung: Nutzt man die Oracle Standard Edition für eine RAC-Datenbank, ist die Verwendung von ASM für alle Datenbankdateien zwingend erforderlich!

Die SQL-Befehle, beispielsweise zum Erstellen eines Tablespaces oder eine Redolog-Datei sehen genauso aus wie sonst auch, wobei aber als Pfad lediglich der Name der betreffenden ASM-Diskgruppe eingetragen. Die konkreten Unterverzeichnisse und Dateinamen werden automatisch ergänzt, beispielsweise:

```
SQL> CREATE TABLESPACE tools
     DATAFILE '+DATA' SIZE 500M;

SQL> SELECT name FROM v$datafile;

NAME
--------------------------------------------------------------------------------
+DATA/prod11g/datafile/tools.303.723221703
...

SQL> ALTER DATABASE
     ADD LOGFILE GROUP 15 ('+DATA','+FRA') SIZE 50M;

SQL> SELECT group#,member FROM v$logfile;

    GROUP# MEMBER
---------- ------------------------------------------------
...
        15 +DATA/prod11g/onlinelog/group_15.305.723222099
        15 +FRA/prod11g/onlinelog/group_15.304.723222101
```

Verwendet man nur je eine Diskgruppe für Daten und Backup-bezogene Dateien, kann man diese auch in den einschlägigen Serverparametern hinterlegen:

```
db_create_file_dest=+DATA
db_create_online_log_dest_1=+DATA
db_create_online_log_dest_2=+FRA
db_recovery_file_dest=+FRA
```

Der Parameter `db_create_file_dest` liefert einen Standardpfad für neue Datendateien und Temporärdateien, so dass die entsprechenden Klauseln bei der Tablespace- oder Datendateiverwaltung komplett wegfallen können, zum Beispiel:

```
SQL> CREATE TABLESPACE tools DATAFILE SIZE 500M;
SQL> SELECT name FROM v$datafile;

NAME
--------------------------------------------------------------------------------
+DATA/prod11g/datafile/tools.303.723224553
...

SQL> ALTER DATABASE
     DATAFILE '+DATA/prod11g/datafile/tools.303.723224553' RESIZE 700m;
```

```
SQL> ALTER TABLESPACE tools ADD DATAFILE '+DATA' SIZE 500M;
```

Noch einfacher geht es mit sogenannten BIGFILE-Tablespaces, bei denen ein Tablespace aus einer einzigen Datendatei bzw. Temporärdatei besteht. Dabei erzwingt die Datenbank eine 1:1-Beziehung zwischen Tablespace und Datendatei, d. h. ein BIGFILE-Tablespace besteht zwingend aus einer einzigen Datendatei. In diesem Fall beträgt die Maximalgröße einer solchen Datendatei und damit des gesamten Tablespaces das 2^{32}-fache der Oracle-Blockgröße, also z. B. 32TB für Oracle-Blöcke der Größe 8KB. Da bei so großen Dateien tendenziell Verwaltungsprobleme auf Dateisystemebene entstehen (Stichwort: „Inode Locking"), empfiehlt sich diese Funktion vor allem bei der Benutzung von ASM:

```
SQL> CREATE BIGFILE TABLESPACE tools DATAFILE SIZE 500M;

SQL> ALTER TABLESPACE tools RESIZE 700M;

SQL> ALTER TABLESPACE tools ADD DATAFILE SIZE 500M;
ALTER TABLESPACE tools ADD DATAFILE SIZE 500M
*
ERROR at line 1:
ORA-32771: cannot add file to bigfile tablespace
```

Achtung: Abgesehen von dieser sehr bequemen Handhabung bringen BIGFILE-Tablespaces aber auch einen deutlichen Nachteil mit sich: Ein Restore bzw. Recovery einer einzelnen beschädigten Datendatei ist nun nicht mehr effizienter als eine entsprechende Operation auf dem gesamten Tablespace. Für die Wiederherstellungszeiten ist es also günstiger, viele kleine Datendateien ansteller einer großen Datendatei zu nutzen. An diesem Umstand ändert sich auch durch die Benutzung von ASM nichts.

Über den Serverparameter `db_recovery_file_dest` werden neben den archivierten Redolog-Dateien auch Flashback-Logs und die Block Change Tracking-Datei auf die betreffende Diskgruppe gelegt. Auch das Sicherungswerkzeug RMAN nutzt dies als Standardlokation für Backups.

Einzig für die Kontrolldateien ist das Verfahren nicht ganz so elegant. Um die Kontrolldateien in ASM abzulegen, muss die Datenbankinstanz mit den obigen Parametern im MOMOUNT-Status gestartet sein. Im Rahmen der Erstellung der Datenbank oder einzelner Kontrolldateien werden nun diese Pfade genutzt, um eine oder mehrere Kontrolldateien anzulegen. Anschließend muss man die konkreten Pfade in den Serverparameter `control_files` eintragen bzw. ergänzen. Erstellt man eine Datenbank mit dem DBCA, kümmert dieser sich darum:

```
SQL> show parameter control_files

NAME                 TYPE        VALUE
-------------------- ----------- ------------------------------------------------
control_files        string      +DATA/prod11g/controlfile/current.256.722257425
```

3.4.6 Migration auf neue Diskgruppen

Das in Abschnitt 3.4.4 beschriebene Rebalancing lässt sich auch für manche Migrationen nutzen. Kapitel 14.3.1 beschreibt dieses Verfahren genauer. Nicht möglich ist damit aber ein Verschieben in eine andere ASM-Diskgruppe. Zum Glück ist dies auch nicht häufig notwendig. Eine Situation, wo dies passieren kann, ist der Wechsel der Redundanz-Einstellung. So kommt es vor, dass man versehentlich sowohl in der ASM-

Diskgruppe als auch im darunter liegenden Storage-System spiegelt, so dass insgesamt eine vierfache Datenhaltung vorliegt. Um dies abzustellen, muss eine neue Diskgruppe, z. B. ohne Spiegelung, erstellt werden. Anschließend werden alle Dateien in diese neue Diskgruppe verschoben.

Der einfachste Weg hierfür besteht aus einem kombinierten Einsatz von SQL*Plus und RMAN. Im Folgenden wird beispielhaft die Migration einer Datendatei von der Diskgruppe `DATA` in die Diskgruppe `DATANEW` gezeigt. Für die Dauer des Kopierens muss die Datendatei in den Offline-Status gesetzt werden, die Migration erfordert also eine Auszeit.

Für andere Dateiarten (Online Redologs, archivierte Redologs, Kontrolldateien etc.) ist das Verfahren ähnlich und eher einfacher, da keinerlei Auszeit benötigt wird. Beispielsweise werden einfach neue Online-Redolog-Gruppen in der neuen Diskgruppe hinzugefügt, anschließend werden die alten Redolog-Gruppen gelöscht.

Ist die gesamte Datenbank betroffen, lohnt es sich ggf. Methoden wie die in Kapitel 14.6 beschriebenen in Betracht zu ziehen, um die Auszeit gering zu halten.

Zunächst wird die Datendatei Offline gesetzt und auf die neue Diskgruppe kopiert:

```
SQL> ALTER DATABASE DATAFILE
      '+DATA/prod11g/datafile/datats.322.722874945' OFFLINE;

RMAN> COPY DATAFILE '+DATA/prod11g/datafile/datats.322.722874945' TO '+DATANEW';

Starting backup at 28-JUN-10
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=50 device type=DISK
channel ORA_DISK_1: starting datafile copy
input datafile file number=00005 name=+DATA/prod11g/datafile/datats.322.722874945
output file name=+DATANEW/prod11g/datafile/datats.256.722875451
tag=TAG20100628T144410 RECID=2 STAMP=722875458
channel ORA_DISK_1: datafile copy complete, elapsed time: 00:00:15
Finished backup at 28-JUN-10
```

In der Ausgabe des RMAN-Kommandos sieht man sowohl den alten als auch den neuen Pfad der Datendatei. Nun muss die Datenbank über die Pfad-Änderung in Kenntnis gesetzt werden:

```
SQL> ALTER DATABASE
     RENAME FILE '+DATA/prod11g/datafile/datats.322.722874945'
     TO '+DATANEW/prod11g/datafile/datats.256.722875451';
```

Damit wird übrigens automatisch auch die ASM-Datei in der alten Lokation gelöscht. Ist das entsprechende Verzeichnis in der ASM-Instanz anschließend leer, wird auch dieses automatisch mit aufgeräumt. Manuelle Aufräumarbeiten sind also nicht notwendig.

In allen folgenden Kommandos wird nur noch der neue Pfad benötigt. Zunächst wird der neue Pfad gegenüber dem RMAN-Katalog bekannt gegeben, anschließend wird die Datendatei wieder Online gesetzt. Zuvor erzwingt ein Oracle ein Recovery, auch wenn dies eigentlich nicht nötig wäre und daher auch sofort erfolgreich abgeschlossen wird:

```
RMAN> SWITCH DATAFILE '+DATANEW/prod11g/datafile/datats.256.722875451' TO COPY;

RMAN> RECOVER DATAFILE '+DATANEW/prod11g/datafile/datats.256.722875451';
```

```
SQL> ALTER DATABASE DATAFILE
      '+DATANEW/prod11g/datafile/datats.256.722875451' ONLINE;
```

3.5 ASM-Cluster-Dateisystem

Im Rahmen der in Version 11gR2 eingeführten Grid Infrastructure wurde das zuvor ausschließlich für Datenbankdateien benutzbare ASM um einen echten Dateisystemtreiber erweitert.

Dadurch steht nun mit ACFS (ASM Cluster File System) auf verschiedenen Plattformen ein echtes Cluster-Dateisystem zur Verfügung, das beispielsweise für Konfigurationsdateien, Log- und Trace-Dateien, extern gespeicherte Daten (BFILEs, externe Tabellen) oder sonstige Anwendungsdateien verwendet werden kann. Neben den Basisfunktionen eines Cluster-Dateisystems sind auch Snapshots für die Erstellung von Point-in-Time-Kopien möglich.

Im Folgenden werden die Erstellung eines ACFS-Dateisystems, die Konfiguration insbesondere mit dem neuen ASM Configuration Assistant sowie Werkzeuge für die laufende Administration beschrieben.

3.5.1 ACFS-Architektur

Eines gleich vorneweg: ACFS ist kein Ersatz oder Nachfolger für ASM, vielmehr stehen beide komplementär zueinander, wie sich weiter unten noch genauer zeigen wird.

Bei ACFS handelt es sich um ein General-Purpose- (also für beliebige Dateien geeignetes) Cluster-Dateisystem, das außerdem die Anfertigung von Read-Only-Snapshots unterstützt. Auch der Weiter-Export von ACFS-Dateisystemen via NFS oder CIFS ist möglich und unterstützt.

Im Gegensatz zu OCFS2 (Oracle Cluster Filesystem 2), das einen eigenen Cluster-Manager namens O2CB mitbringt, nutzt es denselben Cluster-Manager wie ASM bzw. die Datenbank selbst, nämlich die Oracle Clusterware. Dadurch entfallen mögliche Inkonsistenzen, z. B. bei Split-Brain-Situationen, die von beiden Cluster-Managern unterschiedlich gelöst werden könnten. Außerdem steht ACFS für alle Plattformen zur Verfügung, auf denen ASM verfügbar ist. Es bildet also auch aus Kostenaspekten eine sehr interessante und gleichzeitig auf vielen Plattformen einsetzbare Lösung für ein Cluster-Dateisystem.

Wie Abbildung 3.7 zeigt, baut ACFS direkt auf ASM auf, d. h., es „erbt“ insbesondere die Funktionen des darunter liegenden ASM wie z. B. Spiegelung, Striping oder automatisches Rebalancing. Konkret sieht dies so aus, dass einer oder mehrere Teile einer ASM-Diskgruppe für Volumes reserviert werden, auf denen später je ein ACFS-Dateisystem erstellt wird.

Zwischen ACFS und ASM befindet sich eine Schicht namens ADVM (ASM Dynamic Volume Manager), die den eigentlichen Dateisystemtreiber bildet und auch für die betriebssystemseitige Bekanntmachung der als Volumes reservierten Teile der ASM-Diskgruppe zuständig ist.

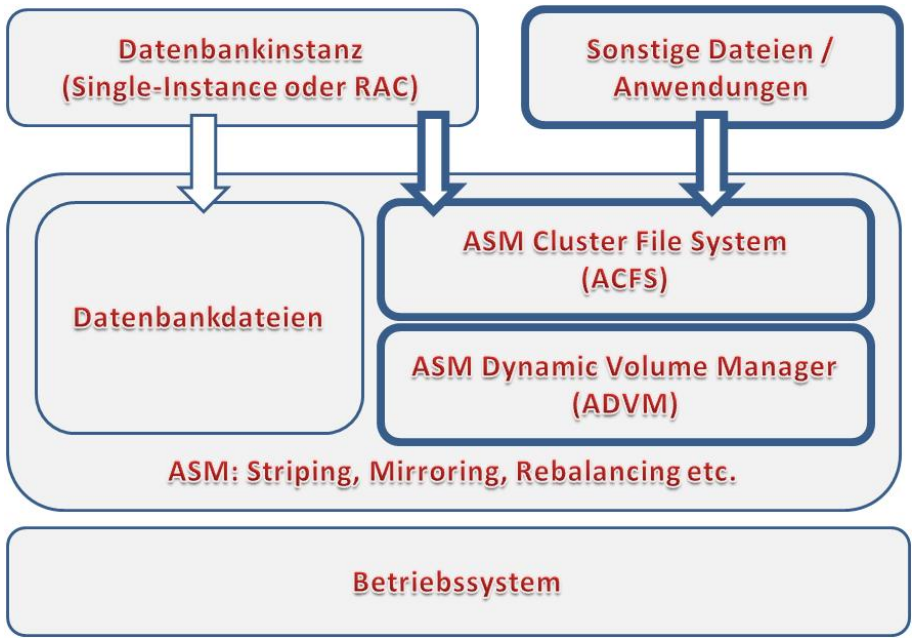


Abbildung 3.7: Architektur von ACFS, fett umrandete Komponenten sind spezifisch für ACFS

3.5.2 ACFS versus ASM

ASM unterstützt seit seiner Einführung ausschließlich die Speicherung von Oracle-Datenbankdateien, also Datendateien, Redologs etc. Hierfür bietet es dank Direct-IO (ungepuffertem IO) eine sehr gute Performance. Im Zusammenspiel mit ACFS ist es nun wichtig zu wissen, dass genau alle Dateien, die für eine direkte ASM-Speicherung unterstützt sind – also die Datenbankdateien –, NICHT (!) für ACFS unterstützt sind. Konkret bedeutet dies also:

- Wenn ACFS benutzt wird, müssen alle Datenbankdateien dennoch direkt in ASM gespeichert werden. Dazu gehören auch RMAN-Backups, Archivelogs, die Serverparameterdatei und – erst seit 11gR2 möglich – die Oracle Cluster Registry (OCR).
- Für alle anderen Dateien darf ACFS verwendet werden, insbesondere für Trace-Dateien und das Alert-Log bzw. für das komplette Automatic Diagnostic Repository (ADR),[17] in Form von BFiles oder externen Tabellen gespeicherte Daten, Anwendungsdateien, Reports, Konfigurationsdateien oder auch für das Oracle Datenbank-Home. Letzteres ist im RAC-Umfeld mit Blick auf Rolling Upgrades aber weniger zu empfehlen.

Erstellt man eine Datenbank mit dem DBCA (Database Configuration Assistant), gibt dieser beim Versuch, die Datenbankdateien in ein ACFS-Dateisystem zu legen, auch eine entsprechende Warnung aus.

17 In diesem Fall setzt man also einfach den Serverparameter `diagnostic_dest` auf ein Verzeichnis innerhalb des ACFS-Dateisystems.

Diese Trennung ist aus Performance-Gründen auch sinnvoll, da ACFS (zumindest in der Version 11.2) kein DirectIO unterstützt.

Das einzige Stück Oracle-Software, das weder von ASM noch von ACFS Gebrauch machen kann, ist das Installationsverzeichnis für die Grid Infrastructure selbst (das Grid Infrastructure Home).

3.5.3 ACFS erstellen und verwalten

Die folgenden Abschnitte stellen zunächst einige Werkzeuge vor, die für die Erstellung und Verwaltung von ACFS-Dateisystemen benutzt werden. Anschließend werden die notwendigen Schritte bis hin zur Erstellung eines ACFS-Dateisystems aufgezeigt. Neben der grafischen Variante (ASMCA) wird stets auch die Benutzung der einschlägigen Kommandozeilenwerkzeuge gezeigt.

Alle Schritte erfolgen – mit wenigen Ausnahmen, die dann jeweils benannt sind – durch den Besitzer des Grid Infrastructure Home (standardmäßig der OS-Benutzer „grid“) oder ein anderes Mitglied der OSASM-Gruppe (standardmäßig die OS-Gruppe „asmadmin“).

Als grafische Oberfläche steht der *ASM Configuration Assistant (ASMCA)* zur Verfügung, der eine bequeme Verwaltung der ASM-Diskgruppen, Volumes und ACFS-Dateisysteme erlaubt. Als Kommandozeilenwerkzeuge werden das bereits aus früheren Datenbankversionen bekannte `asmcmd` sowie das Werkzeug `acfsutil` verwendet.

ACFS-Volumes erstellen

Ein ACFS-Volume kann nur in einer ACFS-Diskgruppe mit bestimmten Mindestkompatibilitäten erstellt werden.[18] Dabei gelten die folgenden Voraussetzungen:

- `compatible.asm` muss mindestens auf dem Wert 11.2 stehen, d. h., der Versionsstand der zugreifenden ASM-Instanz muss mindenstens die Grid Infrastructure 11.2 sein.
- Auch `compatible.advm` muss mindestens auf dem Wert 11.2 stehen. Diesen Parameter gibt es auch erst seit der Grid Infrastructure 11.2. Er gibt den minimalen Versionsstand des ADVM (ASM Dynamic Volume Managers) an, der die Schnittstelle zwischen ASM und ACFS bildet.
- Für `compatible.rdbms` gilt hingegen keine Mindesteinstellung, d. h., es können auch Datenbanken älterer Versionen auf eine Diskgruppe zugreifen, die ACFS-Volumes enthält.

Erstellt man eine neue Diskgruppe, muss dies also wie folgt aussehen:

```
SQL> CREATE DISKGROUP data
     EXTERNAL REDUNDANCY DISKS '...'
     SET ATTRIBUTE 'compatible.asm'  = '11.2',
                   'compatible.advm' = '11.2';
```

Bereits vorhandene Diskgruppen älterer ASM-Instanzen müssen – nach erfolgreichem Versions-Upgrade der ASM-Instanz – noch angepasst werden:

18 Der Serverparameter `compatible` spielt keine Rolle mehr. Stattdessen werden Kompatibilitätswerte pro Diskgruppe gesetzt.

```
SQL> ALTER DISKGROUP data SET ATTRIBUTE 'compatible.asm'  = '11.2';
SQL> ALTER DISKGROUP data SET ATTRIBUTE 'compatible.advm' = '11.2';
```

Zunächst muss im Rahmen des Freiplatzes einer vorhandenen ASM-Diskgruppe ein Volume erstellt werden. Dabei können mehrere Volumes auf einer Diskgruppe definiert werden, es handelt sich also um eine 1:n-Beziehung.

Abbildung 3.8 zeigt die entsprechende Stelle im ASMCA. Im Reiter *Volumes* können vorhandene Volumes eingesehen und konfiguriert werden. Bei der Erstellung eines neuen Volumes müssen die zugrunde liegende Diskgruppe ausgewählt sowie Name und Größe des Volumes angegeben werden.

Mit SQL*Plus oder mit dem `asmcmd`-Werkzeug lässt sich dasselbe erreichen. Zum Beispiel erzeugt jedes der folgenden Kommandos ein ACFS-Volume namens `myacfsvol` mit der Größe 5 GB in der ASM-Diskgruppe DATA:

```
SQL> ALTER DISKGROUP data ADD VOLUME myacfsvol SIZE 5G;
ASMCMD> volcreate -G DATA -s 5G myacfsvol
```

Das Volume wird im Rahmen der Erstellung auch „enabled“, d. h., ADVM blendet eine zugehörige Volume-Datei, die später auch für das Formatieren und Mounten benötigt wird, in das normale Dateisystem ein, z. B. bei Linux unter dem Verzeichnis `/dev/asm`:

```
$ ls -l /dev/asm
brwxrwx--- 1 root asmadmin Sep 12 18:06 myacfsvol-179
```

Die hier vergebene dreistellige Nummer ist über Neustarts hinweg persistent und außerdem konsistent über alle Cluster-Knoten. Dadurch wird ein einfaches automatisches Mounten möglich.

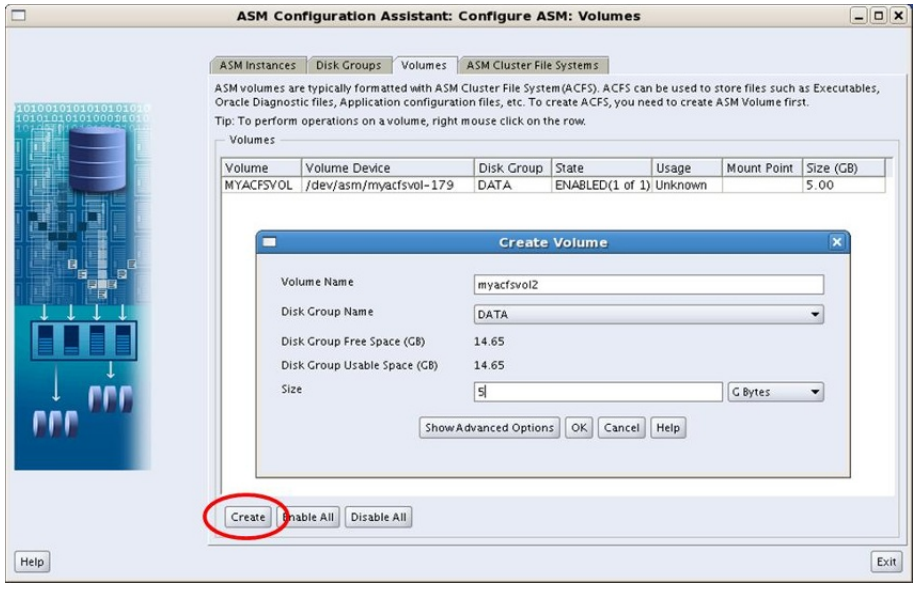


Abbildung 3.8: Erstellen und Verwalten von ACFS-Volumes mit dem ASMCA

Weitere nützliche Kommandos sind:

```
ASMCMD> volstat
ASMCMD> volstat -G DATA
ASMCMD> volinfo -G DATA myacfsvol
ASMCMD> volenable | voldisable -G DATA myacfsvol
```

Damit werden in dieser Reihenfolge

- alle ACFS-Volumes aufgelistet;
- alle ACFS-Volumes der Diskgruppe DATA aufgelistet;
- Informationen zum ACFS-Volume `myacfsvol` in der Diskgruppe DATA geliefert;
- die für das Mounting benötigte Volume-Datei unter `/dev/asm` ein- bzw. ausgeblendet (das Volume wird „enabled“ bzw. „disabled“).

Nach der Erstellung des Volumes kann man dies entweder im ASMCA oder mit den oben genannten Befehlen `volstat` und `volinfo` im `asmcmd`-Werkzeug überprüfen. Oder man nutzt verschiedene V$-Views in der ASM-Instanz. In dieser Situation kommt dafür `V$ASM_VOLUME` infrage, gegebenenfalls im Join mit `V$ASM_DISKGROUP`.

Zwei weitere Views, nämlich `V$ASM_FILESYSTEM` und `V$ASM_ACFSVOLUMES`, klingen zwar ebenfalls vielversprechend, werden aber erst nach dem Erstellen eines Dateisystems und Mounten desselbigen gefüllt.

Wichtig: Das erstellte ACFS-Volume darf nicht weiter partitioniert werden (zum Beispiel mit `fdisk`). Auf die einzelnen Partitionen könnte man auch gar nicht zugreifen, da unter `/dev/asm` nur eine einzige Volume-Datei für das Gesamt-Volume erstellt wird, nicht für jede Partition. Braucht man mehrere Volumes, muss man einfach entsprechend viele Volumes aus einer oder mehreren Diskgruppen erstellen.

ACFS-Dateisysteme erstellen

Ein zuvor erstelltes Volume kann nun mit einem Dateisystem versehen und anschließend auf allen Cluster-Knoten gemountet werden. Außerdem sollte ein automatisches Mounting nach einem Neustart konfiguriert werden.

Im Reiter *ASM Cluster File Systems* des ASMCA lassen sich zum einen vorhandene ACFS-Dateisysteme erstellen und verwalten (siehe Abbildung 3.9). Bei der Erstellung (Abbildung 3.10) muss neben dem zugrunde liegenden ACFS-Volume der zukünftige Mountpoint angegeben werden und ob der Mountpoint „registriert“ werden soll.

Diese Registrierung sorgt nicht nur für ein automatisches Mounten beim Server-Neustart, sondern auch für ein automatisches Mounten auf allen anderen Cluster-Knoten (nach spätestens 30 Sekunden). Der übliche Eintrag in der Datei `/etc/fstab` funktioniert hier nicht, da ja zuerst die ASM-Instanz starten muss. Aus demselben Grunde kann ACFS auch nicht für Dateisysteme verwendet werden, die zum Booten benötigt werden.

Außerdem gibt es neben dem Auswahlpunkt *General Purpose File System*, mit dem einfach ein beliebiges ACFS-Dateisystem erstellt werden kann, noch eine weitere Option *Database Home File System*. Letztere bildet den Spezialfall ab, dass man ein Oracle-Home, also eine Datenbank-Software, in dem ACFS-Dateisystem ablegen möchte. Die Sonderbehandlung dieses Falls ergibt sich daraus, dass die Clusterware darauf achten muss, erst das Dateisystem zu mounten, bevor Listener oder Datenbanken aus diesem Oracle-Home gestartet werden können. Entsprechende Abhängigkeiten werden dann in der Clusterware abgebildet.

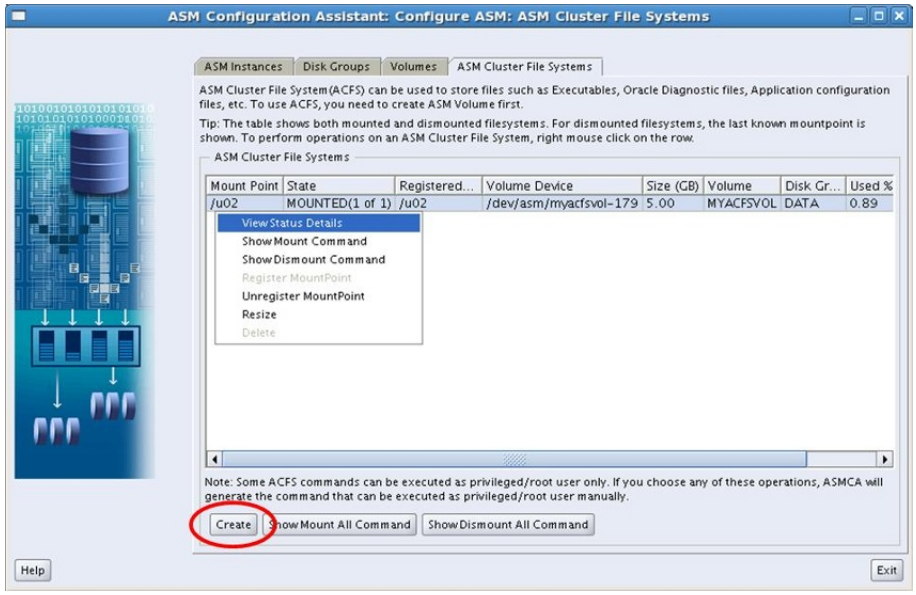


Abbildung 3.9: Übersicht über die ACFS Dateisysteme

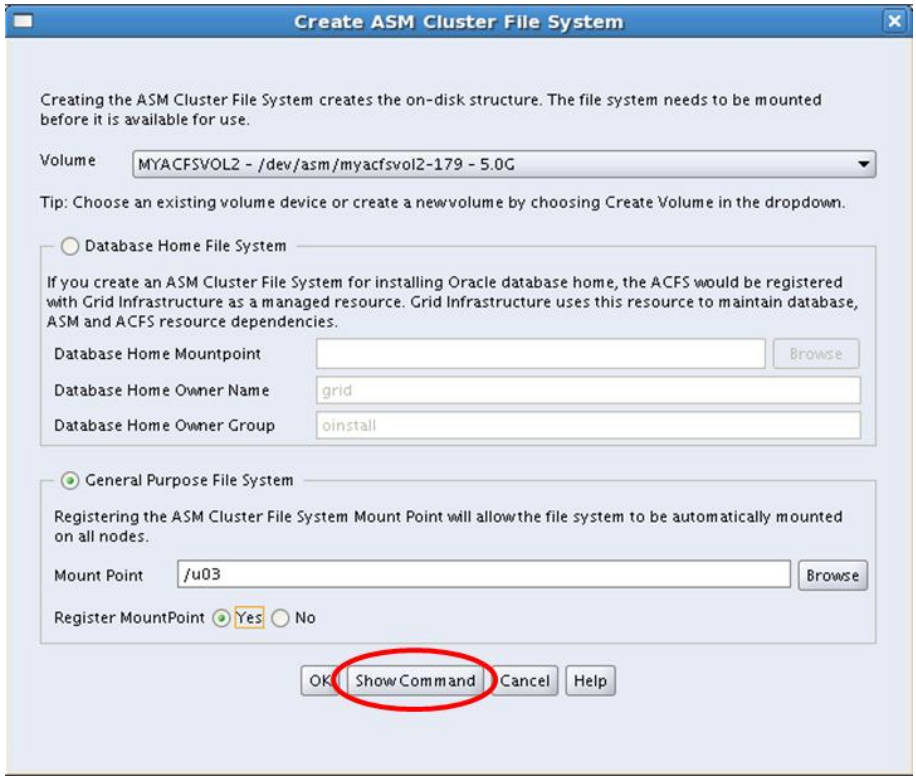


Abbildung 3.10: Erstellen eines neuen Dateisystems

Der Nachteil einer solchen Konfiguration ist allerdings, dass Rolling Upgrade-Verfahren damit nicht mehr möglich sind, daher raten wir von dieser Verwendung eines ACFS-Dateisystems (und ganz allgemein Shared Oracle Homes im Cluster-Umfeld) eher ab.

Bei der Liste vorhandener Dateisysteme fällt auf, dass es für bereits vorhandene Dateisysteme zwei Spalten gibt, die sich auf den Mountpoint für das ACFS-Dateisystem beziehen:

- Unter *Mount Point* wird der aktuelle Mountpoint angegeben.
- Der Eintrag *Registered Mount Point* dagegegen verweist auf denjenigen, der für das automatische Mounting beim Start des Servers benutzt wird.

Bis auf Ausnahmesituationen im Rahmen von Wartungsarbeiten sind in der Praxis beide Pfade identisch.

Die Kommandozeilen-Pendants für die Erstellung des ACFS-Dateisystems sowie das optionale – aber meistens sinnvolle – Registrieren des Mountpoints lassen sich aus dem ASMCA heraus mit dem Button „Show Command“ erzeugen und lauten beispielsweise:

```
$ /sbin/mkfs -t acfs /dev/asm/myacfsvol-179
$ /sbin/acfsutil registry -a -f /dev/asm/myacfsvol-179 /u02
```

Nun sollte das Dateisystem mit normalen Betriebssystemkommandos sichtbar sein, beispielsweise:

```
$ df -k /u02
Filesystem              1K-blocks      Used Available Use% Mounted on
/dev/asm/myacfsvol-179    6291456    102380   6189076   2% /u02
```

Ab hier können auch die Views `V$ASM_FILESYSTEM` und `V$ASM_ACFSVOLUMES` zur Kontrolle verwendet werden.

ACFS-Volumes und -Dateisysteme können dynamisch vergrößert und – falls der Platz noch nicht belegt ist – verkleinert werden. Das Verfahren ist allerdings unterschiedlich, je nachdem, ob schon ein Dateisystem existiert oder es sich noch um ein leeres Volume handelt:

Bei einem leeren Volume genügt einer der beiden folgenden Befehle für eine Vergrößerung auf beispielsweise 8 GB:

```
ASMCMD> volresize -G data -s 8G myacfsvol
SQL> ALTER DISKGROUP DATA RESIZE VOLUME myacfsvol SIZE 8G;
```

Befindet sich dagegen schon ein Dateisystem auf dem Volume, erhält man mit dem obigen Kommando eine zunächst irritierende Fehlermeldung:

```
SQL> ALTER DISKGROUP data RESIZE VOLUME myacfsvol SIZE 8G;
ORA-15032: not all alterations performed
ORA-15476: resize of ACFS volume must use ACFS resize command
```

Hintergrund dafür ist, dass ja auch die Formatierung und Header-Informationen des Dateisystems angepasst werden müssen. Dafür muss stattdessen (nicht zusätzlich) entweder der ASMCA oder das Werkzeug `acfsutil` benutzt werden:

```
$ /sbin/acfsutil size 8G /u02
```

Ein manuelles Mounten und Unmounten des Dateisystems ist durch einen root-Benutzer mit den Standardbefehlen hierfür möglich, wobei – wie bei der Erstellung auch – der neue Dateisystemtyp „acfs“ verwendet wird:

```
$ mount -t acfs /dev/asm/myacfsvol-179 /u02
$ umount /u02
```

ACFS-Snapshots

ACFS bietet die Möglichkeit, für jedes Dateisystem bis zu 63 *Read-Only-Snapshots* zu erstellen. Diese werden mithilfe der *Copy on Write*-Technik implementiert, d. h., bei der Erstellung wird nur Metainformation über den Snapshot gespeichert. Erst wenn sich Daten ändern, werden die alten Blockinhalte („Before-Image“) zuvor gesichert. Verglichen mit der Oracle-Datenbank funktionieren die Snapshots also wie Flashback-Abfragen.

Um beispielsweise für das unter `/u02` gemountete ACFS-Dateisystem einen Snapshot namens „mysnapshot1“ zu erstellen, genügt folgender Befehl:

```
$ /sbin/acfsutil snap create mysnapshot1 /u02
```

Der Inhalt zum Zeitpunkt des Snapshots wird daraufhin unter folgendem Pfad eingeblendet und kann dort zum Beispiel von Backup-Werkzeugen für eine konsistente, statische Ansicht verwendet werden:

```
/u02/.ACFS/snaps/mysnapshot1
```

Neben der View `V$ASM_ACFSSNAPSHOTS` liefert entsprechend auch dieser Befehl eine Auflistung aller Snapshots für den Mountpoint `/u02`:

```
$ ls /u02/.ACFS/snaps
```

Etwas genauere Informationen zu Snapshots, insbesondere über den Speicherplatzverbrauch, erhält man mit einer weiteren Variante des `acfsutil`-Werkzeugs:

```
$ /sbin/acfsutil info fs /u02
/u02
    ACFS Version: 11.2.0.1.0.0
    flags:        MountPoint,Available
    mount time:   Mon Sep 21 15:41:36 2009
    volumes:      1
    total size:   6442450944
    total free:   6337613824
    primary volume: /dev/asm/myacfsvol-179

... lots of stuff ...

number of snapshots:  1
snapshot space usage: 32768
```

Snapshots können in beliebiger Reihenfolge wieder gelöscht werden, wobei Dateien oder Verzeichnisse des Snapshots nicht mehr geöffnet sein dürfen, insbesondere müssen entsprechende Shells geschlossen werden oder in ein anderes Verzeichnis wechseln:

```
$ /sbin/acfsutil snap delete mysnapshot1 /u02
```

Backup von ACFS-Dateisystemen

Für die Sicherung von ACFS-Dateisystemen kommt jede Sicherungssoftware infrage, die Standard-Dateisystem-APIs benutzt, also einfach Read- oder Write-Operationen ausführt.

Die Benutzung anderer Schnittstellen, z. B. VSS (Volume Shadow Copy Service) auf Windows-Rechnern, ist für ACFS nicht unterstützt.

Um eine konsistente Sicherung durchzuführen, bietet sich die Benutzung der ACFS-Snapshots an. Durch ein kleines Integrationsskript ist es einfach möglich, einen ACFS-Snapshot zu erstellen, das Backup von diesem Snapshot durchzuführen und den Snapshot anschließend zu löschen.

Es ist zu hoffen, dass es in Zukunft eine explizite Liste in der Art einer Zertifizierungsmatrix für geeignete Backup-Software geben wird.

3.6 Storage im Cluster-Umfeld

Als Storage-Lösung kommt naturgemäß nur eine SAN- oder NAS-basierte Speicherung infrage. Im High-End-Bereich dominieren SANs. Abgesehen davon kommen NAS-File-Server bei richtiger Konfiguration auf ähnliche Performance-Werte und stellen somit eine interessante, kostengünstige Alternative dar. Hier muss allerdings genau geprüft werden, welche NAS-Server von Oracle unterstützt und zertifiziert sind. Typische Beispiele sind die NetApp FAS-Server, EMC^2 Celerra, IBM N Series oder die Sun StorageTek 5000 Series. Standard-NFS (als Unix-Service) oder Samba-Dienste sind hingegen nicht unterstützt.[19]

Da bei RAC-Datenbanken ein gleichzeitiger Zugriff der beteiligten Instanzen stattfindet, muss die Speicherung der Datenbank-Dateien auf einem cluster-fähigen Dateisystem erfolgen. Bei neuen Installationen fällt die Wahl dabei meistens auf ASM, da es ohnehin Teil der Datenbank-Software ist, also keine zusätzlichen Lizenzkosten anfallen, andererseits auch eine sehr gute Performance und einen hohen technischen Reifegrad hat. ASM ist in Abschnitt 3.4 genauer beschrieben.

Alternativ zu ASM gab und gibt es für Unix und Linux vorrangig das *Veritas Cluster File System (VxFS)* als Teil der *Veritas Storage Foundation for Oracle RAC*[20].

Für Linux-Systeme bietet Oracle außerdem *OCFS2*, die zweite Version des *Oracle Cluster File Systems*. Dabei handelt es sich um ein quelloffenes[21], POSIX-konformes, echtes Cluster-Dateisystem mit Journaling-Funktionen. Im Gegensatz zur Vorgängerversion OCFS ist OCFS2 ein General-Purpose-Dateisystem, es unterstützt also beliebige Anwendungen und Dateien auch außerhalb der Oracle-Datenbank. Entsprechenden Support für OCFS2 bieten sowohl Oracle selbst im Rahmen des Unbreakable Linux Network – aber auch für Kunden mit Redhat Enterprise Linux und Datenbank-Software – als auch Novell für Kunden mit SUSE Linux Enterprise Server.

---

19 Zu dieser Regel gibt es eine kleine, aber wichtige Ausnahme im Kontext von Voting-Disks, die in Abschnitt 3.6.3 beschrieben ist.

20 ehemals Veritas Database Edition Advanced Cluster (DBAC)

21 Der Quellcode für OCFS2 ist unter der GNU General Public License (GPL) Version 2 verfügbar.

Angesichts der besseren Plattformverfügbarkeit, der eindeutigeren Support-Situation sowie der strategischen Bevorzugung von ACFS dürfte OCFS2 aber langfristig eher ein Auslaufmodell darstellen. Wer dennoch weitere Informationen darüber sucht, findet diese gebündelt auf der Seite:

```
http://oss.oracle.com/projects/ocfs2/
```

Bis zur Datenbankversion 11gR1 bestand der Vorteil eines echten Cluster-Dateisystems im Gegensatz zu ASM vor allem darin, dass es:

- auch für Zwecke außerhalb der Datenbank benutzt werden kann,
- eine einfachere Dateiverwaltung als ASM bietet,
- die Speicherung zum Beispiel externer Tabellen oder sogenannte BFILEs[22] im RAC-Umfeld ermöglicht.

Hier muss jeweils im Einzelfall entschieden werden, ob diese Vorteile die ggf. zusätzlichen Lizenzkosten aufwiegen. Mit der Datenbankversion 11gR2 hat sich die argumentative Waagschale aber weiter in Richtung ASM verschoben, da dies jetzt mit ACFS auch als General-Purpose-Dateisystem verwendbar ist.

Die Datenbank-Software, also die Oracle-Homes für Datenbank, ASM und Clusterware, sollten möglichst nicht auf einem zentralen Storage liegen, sondern auf lokalen Verzeichnissen. Der Vorteil dabei ist, dass Rolling Upgrades und Patches durchgeführt werden können, ohne dass der gesamte Cluster heruntergefahren werden muss. Nachteilig ist natürlich der höhere Gesamtaufwand solcher Wartungsarbeiten.

Wichtig: Damit Rolling Upgrades und Patches auch für die ASM-Software möglich sind, muss diese jeweils in einem separaten Oracle-Home installiert sein. Eine Speicherung zusammen mit der Datenbank-Software sollte also vermieden werden. Für die Clusterware gilt dasselbe, hier ist ein separates Oracle-Home aber ohnehin zwingend erforderlich.

3.6.1 ASM und ACFS

Mehr und mehr setzt sich Oracles *Automatic Storage Management*, gerade im Cluster-Umfeld, als Storage-Lösung durch. Seit der Version 11gR2 ist es nicht nur zur Speicherung von Datenbankdateien, sondern auch für sonstige Dateien im Umfeld der Oracle-Datenbank oder auch komplett unabhängig von der Oracle-Datenbank einsetzbar.

Diese beiden Techniken sind in den Abschnitten 3.4 und 3.5 im Detail beschrieben. Ihr Einsatz im Cluster-Umfeld sollte in jedem Falle geprüft werden.

Hat es man mit einer RAC-Datenbank auf Basis der Oracle Standard Edition zu tun, ist die Verwendung von ASM, ggf. zusammen mit ACFS, sogar die einzige unterstützte Storage-Variante.

22 In beiden Fällen werden in der Datenbank lediglich Zeiger auf eine Datei im Dateisystem gespeichert. Diese Datei enthält die eigentlichen Daten. Für den Benutzer ist es aber nicht ersichtlich, dass die Daten nicht aus der Datenbank kommen, sondern lediglich „durchgereicht“ werden.

3.6.2 Oracle Cluster File System

Das angesprochene OCFS2-Dateisystem wird mittlerweile bei SLES-Distributionen als integrierter Bestandteil ausgeliefert. Für RHEL (und damit auch OEL) finden sich auf den Oracle-Seiten entsprechende RPM-Pakete zum Herunterladen.

Nach der Installation auf allen Cluster-Knoten liegen zwei neue Services – „o2cb“ und „ocfs2“ – im Verzeichns `/etc/init.d`.

Der o2cb-Dienst konfiguriert und startet die O2CB-Clusterware, die insbesondere Aufgaben wie Distributed Lock Management (DLM) oder Heartbeating übernimmt, um eine Kommunikation der Cluster-Knoten untereinander zu ermöglichen. Der Dienst ocfs2 übernimmt beim Server-Start das Mounten der OCFS2-Partitionen, so dies vorher konfiguriert wurde.

Achtung: Die O2CB-Clusterware hat nichts mit der „Oracle Clusterware“ zu tun, die als Basis für Oracle RAC und ASM dient. Es handelt sich vielmehr um eine eigenständige Clusterware, die ausschließlich OCFS2 bedient.

Prinzipiell kann OCFS2 auch als normales Dateisystem (ohne Clustering) verwendet werden. In diesem Fall wird der gesamte O2CB-Stack nicht benötigt, und es kann unmittelbar nach der Installation ein Dateisystem angelegt werden, z. B. auf `/dev/sdb1`. Die Option `-M local` gibt an, dass es sich um ein nicht-geclustertes Dateisystem handelt, so dass auch für weitere Aktionen wie Mounten nicht auf laufende Clusterware getestet wird:

```
$ mkfs.ocfs2 -M local -L "myocfs2" -b 4K -C 128K /dev/sdb1
mkfs.ocfs2 1.2.7
Filesystem label=myocfs2
Block size=4096 (bits=12)
Cluster size=131072 (bits=17)
Volume size=4293525504 (32757 clusters) (1048224 blocks)
2 cluster groups (tail covers 501 clusters, rest cover 32256 clusters)
Journal size=67108864
Initial number of node slots: 1
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 1 block(s)
Formatting Journals: done
Writing lost+found: done
mkfs.ocfs2 successful

$ mount -t ocfs2 /dev/sdb1 /ocfs

$ df -k /dev/sdb1
Filesystem    1K-blocks      Used Available Use% Mounted on
/dev/sdb1       4192932     71636   4121296   2% /ocfs
```

Der Normalfall wird aber wohl eher die Benutzung als Cluster-Dateisystem sein. Dazu muss zunächst die O2CB-Clusterware ans Laufen gebracht werden. Die folgenden Schritte müssen als root-Benutzer genau in dieser Reihenfolge durchlaufen werden:

1. Initiales Aktivieren der O2CB-Clusterware. Es ist noch keine Cluster-Konfiguration vorhanden, die Fehlermeldung in der letzten Zeile ist daher nicht zu beanstanden.

```
$ /etc/init.d/o2cb online
Mounting configfs filesystem at /sys/kernel/config: OK
Loading module "ocfs2_dlmfs": OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK
Checking O2CB cluster configuration : Failed
```

2. Initiale Konfiguration der O2CB-Clusterware. Hier wird ein Name für den Cluster vergeben (standardmäßig „ocfs2“) und festgelegt, ob die Clusterware beim Starten des Servers automatisch mitgestartet wird.

   Wichtig: Eine Standardkonfiguration via `chkconfig` reicht hierfür nicht aus! Es muss zusätzlich im unten stehenden Dialog die Option `Load O2CB driver on boot` ausgewählt werden, ansonsten bleiben nachfolgende Aufrufe der Form `./o2cb start` ohne Effekt!

   ```
   $ /etc/init.d/o2cb configure
   Configuring the O2CB driver.

   This will configure the on-boot properties of the O2CB driver.
   The following questions will determine whether the driver is loaded on
   boot.  The current values will be shown in brackets ('[]').  Hitting
   <ENTER> without typing an answer will keep that current value.  Ctrl-C
   will abort.

   Load O2CB driver on boot (y/n) [n]: y
   Cluster to start on boot (Enter "none" to clear) [ocfs2]: mycluster
   Specify heartbeat dead threshold (>=7) [31]:
   Specify network idle timeout in ms (>=5000) [30000]:
   Specify network keepalive delay in ms (>=1000) [2000]:
   Specify network reconnect delay in ms (>=2000) [2000]:
   Writing O2CB configuration: OK
   Checking O2CB cluster configuration : Failed
   ```

3. Definition der Cluster-Knoten. Dies geht am bequemsten über die OCFS2-Konsole (siehe Abbildung 3.11).

   ```
   $ ocfs2console
   ```

   Alternativ lässt sich dieser Vorgang über das Kommandozeilenwerkzeug `o2cb_ctl` skripten.

   Wichtig ist: Der Knotenname muss mit dem Hostnamen identisch sein (die Domäne kann aber weggelassen werden). Die angegebene IP-Adresse muss einer Netzwerkkarte des Knotens zugeordnet sein. Im Zusammenhang mit RAC empfiehlt sich hier die private Adresse für den Interconnect. Der gewählte TCP-Port muss firewall-technisch zwischen den Knoten freigeschaltet sein.

   Im Hintergrund wird nun die Datei `/etc/ocfs2/cluster.conf` erstellt, die auf allen Knoten identisch sein muss. Diese Datei sollte also entweder manuell oder über den Menüpunkt „Propagate Configuration“ in der OCFS2-Konsole verteilt werden.

4. Erstellen eines Cluster-Dateisystems. Dies kann wahlweise über die OCFS2-Konsole oder per Kommandozeile geschehen:

   ```
   $ mkfs.ocfs2 -M cluster -L "myocfs2" -b 4K -C 128K /dev/sdb1 -N 4
   ```

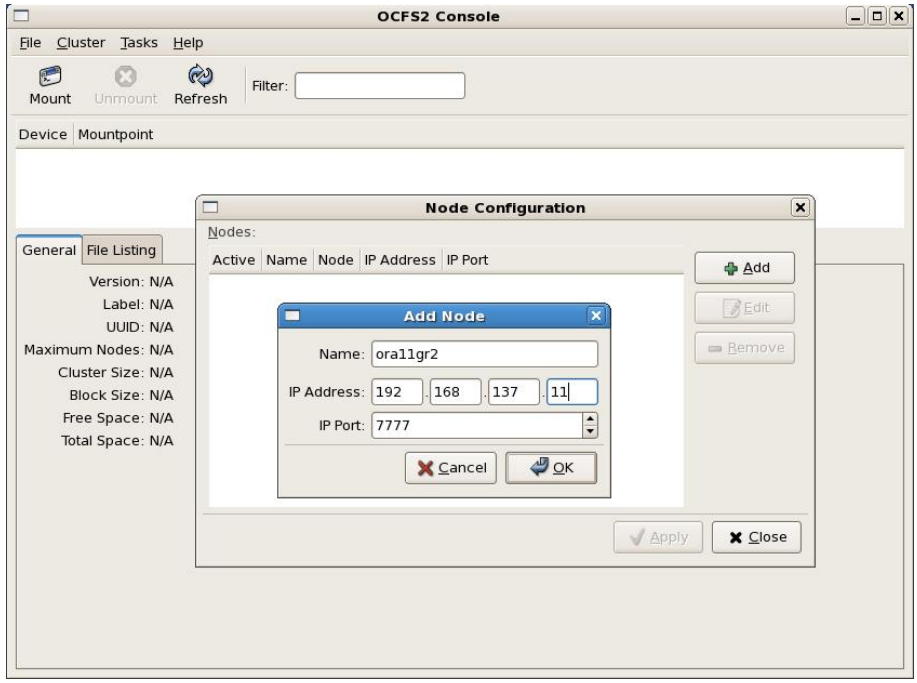


Abbildung 3.11: Definition der Cluster-Knoten mit der OCFS2-Konsole

Hierdurch wird ein Cluster-Dateisystem mit dem Namen `myocfs2` erstellt. Über die Option `-N` wird die Anzahl der Node Slots konfiguriert, d. h. die Anzahl der Cluster-Knoten, die gleichzeitig dieses Dateisystem mounten können. Die Zahl kann nachträglich erhöht, aber nicht verringert werden, wahlweise über die Konsole oder per Kommandozeile `tunefs.ocfs2`. Der Wert sollte nicht unnötig hoch angesetzt werden, da je eine Journal-Datei pro Node Slot benötigt wird und jede Journal-Datei typischerweise zwischen 64 MB und 256 MB Speicherplatz benötigt.

Beim Erstellen eines OCFS2-Dateisystems können mit den Parameteren `-b` und `-C` die Block- und die Cluster-Größe angegeben werden. Für die Blockgröße – die kleinste Speicherungseinheit für Metadaten – empfiehlt sich praktisch immer eine Größe von 4 KB. Für die Cluster-Größe – die kleinste Speicherungseinheit für die eigentlichen Nutzdaten – kann meistens derselbe Wert genommen werden. Für Oracle-Datenbankdateien sollte die Clustergröße aber nicht kleiner sein als die Datenbank-Blockgröße, also z. B. standardmäßig 8 KB, auch etwas größere Werte (16 KB, 32 KB) sind problemlos. Beide Werte können nachträglich nicht geändert werden.

Wie gesagt, handelt es sich bei OCFS2 um ein Journaling-Dateisystem, was den Vorteil einer kürzeren MTTR (Mean Time to Recovery) bietet, aber den Overhead der Journaling-Dateien mit sich bringt. Diese Journaling- und einige weitere interne Dateien werden normalerweise ausgeblendet, können aber mit dem Kommando `debugfs.ocfs2` wie folgt sichtbar gemacht werden (aus Platzgründen sind in dem folgenden Listing einige Zeilen und Spalten abgeschnitten):

```
$ debugfs.ocfs2 -R "ls -l //" /dev/sdb1
        67108864 13-Mar-2009 11:26 journal:0000
        67108864 13-Mar-2009 11:26 journal:0001
        67108864 13-Mar-2009 11:26 journal:0002
        67108864 13-Mar-2009 11:26 journal:0003
```

Eine Übersicht, welche OCFS2-Dateisysteme auf welchen Knoten gemountet sind, erhält man wie folgt:

```
$ mounted.ocfs2 -f
Device       FS     Nodes
/dev/sdb1    ocfs2  ora11gr2

$ mounted.ocfs2 -d
Device       FS     UUID                                 Label
/dev/sdb1    ocfs2  cd3d6581-845e-4397-adcb-f606d20f935e  myocfs2
```

Jeder Knoten meldet sich beim Booten per Heartbeat bei den anderen Knoten. Die O2CB-Clusterware überwacht den Cluster mithilfe dieser Heartbeats und dauerhaften TCP-Verbindungen zwischen den Knoten. Hat ein Knoten zu weniger als der Hälfte aller Knoten, die er über Heartbeats kennt, eine solche Verbindung, zwingt er sich selbst in einen Reboot („I/O-Fencing“).

Für den Grenzfall, dass der Knoten mit genau der Hälfte der anderen Knoten verbunden ist, überlebt er, solange er den Knoten mit der kleinsten Knotennummer sieht. Diese Knotennummer wird von der OCFS2-Konsole automatisch vergeben, kann aber manuell durch Anpassen der Datei `/etc/ocfs2/cluster.conf` geändert werden.

Wird OCFS2 im RAC-Umfeld verwendet, empfiehlt es sich – da die RAC-Clusterware einen ähnlichen Fencing-Mechanismus verwendet –, dass RAC- und O2CB-Clusterware denselben Knoten mit der kleinsten Knotennummer verwenden. Ansonsten könnte es – wenn auch mit sehr geringer Wahrscheinlichkeit – zu widersprüchlichen Entscheidungen der beiden Clusterwares kommen, die zu unnötigen Knoten-Reboots führen.

3.6.3 OCR und Voting-Disks

Zwei spezielle „Dateien“ werden für eine RAC-Umgebung benötigt und müssen ebenfalls zentral für alle Instanzen sichtbar sein:

Die Oracle Cluster Registry (OCR) enthält die aktuelle, verbindliche Konfiguration des Clusters inklusive aller Datenbanken, Instanzen, Listener, virtuellen IP-Adressen, Services etc.

Die Voting Disk(s) helfen bei der Lösung bestimmter Cluster-Probleme, z. B. sogenannter „Split Brain“-Situationen.

Bis zur Datenbankversion 11gR1 konnten sowohl OCR als auch Voting-Disks nicht in ASM gespeichert werden. Um nicht lediglich dafür doch noch ein teures Cluster-Dateisystem zu benötigen, wurden und werden hier häufig Raw-Devices, NFS oder – auf Linux – OCFS2 zur Speicherung benutzt. Die Version 11gR2 schafft hier Abhilfe, d. h., OCR und Voting Disks können nun ebenfalls in ASM abgelegt werden.

Bei den Voting-Disks stellt sich aber grundsätzlich noch eine weitere Herausforderung: Damit diese ihre Funktion sinnvoll erfüllen können, müssen mindestens drei Voting-Disks an physikalisch unabhängigen Storage-Arrays vorliegen. Häufig sind aber nur zwei

Storage-Arrays vorhanden, z. B. zwei Rechenzentren. Für diesen Fall ist es ab der Clusterware-Version 10.2.0.2 unterstützt, eine dritte Voting-Disk (und nur diese!) auf einem Standard-NFS-Server zu betreiben.

Wichtig: Wenn RAC mit der Oracle Standard Edition benutzt wird, ist eine Speicherung aller Datenbank-Dateien (außer OCR und Voting-Disks) in ASM grundsätzlich die einzige unterstützte Variante! Bis einschließlich Version 11gR1 müssen OCR und Voting-Disks in diesem Fall auf Raw- bzw. Block-Devices gespeichert werden (OCFS ist ebenfalls nicht unterstützt).

3.7 Besonderheiten im Cluster-Umfeld

Im Folgenden werden einige Spezialitäten behandelt, die für eine Oracle RAC-Datenbank zu beachten sind. Bei Oracle RAC handelt es sich um eine active/active-Cluster-Lösung auf Basis einer „Shared All"-Plattenarchitektur. Zwei oder mehr Oracle-Instanzen, die auf unterschiedlichen Knoten laufen, teilen sich eine zentrale Datenbank (siehe auch Kapitel 12.3). Diese Konfiguration ist für die meisten Plattformen freigegeben, hat aber zusätzliche hard- und softwaretechnische Anforderungen im Vergleich zu Single-Instance-Datenbanken.

3.7.1 Netzwerk und Interconnect

Zunächst einmal benötigen die Instanzen eine direkte Kommunikationsmöglichkeit, den sogenannten Interconnect. Dabei muss es sich um ein privates Netzwerk zwischen den beteiligten Knoten handeln. Hierfür wird meistens Gigabit Ethernet verwendet, gelegentlich auch Infiniband. Außerdem gibt es zunehmenden Support für 10-Gb-Ethernet. Da der Interconnect eine hochkritische Komponente darstellt, empfehlen sich zwecks Ausfallsicherheit Techniken zur Zusammenschaltung mehrerer Netzwerk-Ports, also NIC-Bonding oder -Teaming, IPMP, EtherChannel oder IEEE 802.3ad.

Die Verbindung muss (auch in einem 2-Knoten-Cluster) zwingend über einen Switch erfolgen, eine direkte Verbindung mittels Crossover-Kabel ist nicht unterstützt.[23] Für den Interconnect sollte in jedem Fall eine dedizierte Netzwerkkarte benutzt werden, damit die erforderliche Bandbreite garantiert ist.

In einer RAC-Konfiguration enthält also jeder Knoten mindestens zwei Netzwerkkarten: je eine für den Interconnect (Private IP-Adresse) und für die normale Kommunikation mit Clients (Public IP-Adresse). Im Laufe der Installation der Oracle-Clusterware wird jedem Knoten eine weitere, virtuelle IP-Adresse zugewiesen und die Netzwerkkarte mit der Public IP-Adresse gebunden. Gegebenenfalls gibt es noch weitere Netzwerkkarten im Rahmen von NIC-Bonding oder für ein Management-LAN, mit dem Administratoren sich auch bei einer hohen Netzbelastung noch an den Server verbinden können.

Bei der Installation und auch bei späteren Patches der Datenbank-Software wird der Installer auf einem beliebigen Knoten ausgeführt. Dort wird die Software installiert und anschließend per SCP auf die anderen Knoten geschoben. Damit das funktioniert, müssen für die einschlägigen OS-Benutzer (z. B. `oracle` und `grid`) SSH-Logins von jedem

23 Für reine Testszenarien ist natürlich eine Crossover-Verbindung ein gangbarer Weg, und technisch kann der Oracle-Kernel ohnehin nicht zwischen einer direkten und einer geswitchten Verbindung unterscheiden. Aber eine direkte Verbindung wird von Oracle eben nicht unterstützt.

Knoten auf jeden anderen Knoten ohne interaktiven Eingriff möglich sein. Dies wird erreicht, indem vor der Erstinstallation für jeden betreffenden OS-Benutzer folgende Prozedur durchlaufen wird:

1. Zunächst wird ggf. ein Schlüsselpaar (Public Key und Private Key) erzeugt, z. B.:

   ```
   $ ssh-keygen -t rsa
   ```

   Wichtig hierbei ist, dass keine Passphrase benutzt wird, da diese ja einen interaktiven Eingriff beim Login erzwingen würde.

2. Die SSH-Konfiguration inklusive des erzeugten Schlüsselpaars liegt standardmäßig im Verzeichnis `~/.ssh` des Benutzers. Dort findet sich der Public Key in der Datei `id_rsa.pub`. Die Public Keys aller Cluster-Knoten müssen nun in eine einzige Datei namens `authorized_keys` konkateniert werden. Dazu läuft man einfach der Reihe nach durch alle Knoten und hängt den Public Key des jeweiligen Knotens hinten an, z. B. mittels:

   ```
   $ cat id_rsa.pub >> authorized_keys
   ```

   Die Datei `authorized_keys` muss jedes Mal auf den nächsten Cluster-Knoten weitergeschoben werden, z. B. auch mittels SCP.

3. Die so gebildete, vollständige Datei `authorized_keys` muss nun wiederum auf alle Knoten verteilt werden, jeweils in das Verzeichnis `~/.ssh`.

4. Zum Schluss sollte man peinlich genau die Rechte der einzelnen Konfigurationsdateien überprüfen: `authorized_keys`, `known_hosts` und `*.pub` müssen die Rechte-Bitmap 644 haben, alle anderen Dateien die Rechte-Bitmap 600. Ansonsten weigert sich SSH grundsätzlich, entsprechende Authentifizierungen durchzuführen, gibt aber meistens keine sprechenden Fehlermeldungen aus!

5. Nun sollte von einem beliebigen Cluster-Knoten aus der Login auf einen beliebigen anderen Knoten durchgeführt und ggf. jeweils der Host-Schlüssel des Zielknotens akzeptiert werden.

6. Anschließend sollte derselbe Login ohne jeden interaktiven Eingriff (keine Eingabe von Passwort oder Passphrase, keine Bestätigung des Host-Schlüssels) durchgehen:

   ```
   $ ssh oranode2
   ```

3.7.2 Cluster-Software

Grundsätzlich können als Cluster-Software plattformspezifische Lösungen wie Solaris Cluster, HP Serviceguard mit der zugehörigen „Extension for RAC“, IBM HACMP oder TruCluster sowie der Veritas Cluster Server als Teil der Veritas Storage Foundation for Oracle RAC verwendet werden. Trotzdem ist in den letzten Jahren ein deutlicher Trend hin zur Verwendung der Oracle-Clusterware zu beobachten. Wesentliche Gründe hierfür sind:

- Die Oracle Clusterware ist in der Datenbank-Software inkludiert, es fallen also keine weiteren Lizenzkosten an.
- Selbst wenn eine anderweitige Cluster-Software bereits lizenziert und installiert ist, muss seit der Datenbankversion 10g die Oracle Clusterware zusätzlich darüber installiert werden. Dadurch erhöht sich die Komplexität der Umgebung deutlich.

- Für RAC-Umgebungen auf Linux oder MS-Windows ist Oracle Clusterware die einzige zertifizierte Lösung.
- Für RAC mit der Oracle Standard Edition ist Oracle Clusterware die einzige unterstützte Variante.

3.7.3 Cluster Verification Utility

Eines der nützlichsten Werkzeuge im Zusammenhang mit Cluster-Installationen ist das *Cluster Verification Utility (CVU)*, das auf dem Installationsmedium der Oracle Clusterware mit ausgeliefert wird, aber auch separat über die Oracle-Webseite heruntergeladen werden kann. Außerdem wird es im Rahmen der Oracle Clusterware mit installiert. Es ist für die Plattformen Linux, Solaris, HP-UX, AIX und MS-Windows verfügbar.

CVU verifiziert die Konfiguration der für den RAC-Betrieb benötigten Infrastruktur, z. B. benötigte Betriebssystem-Packages und Kernel-Parameter, die korrekte SSH-Konfiguration, Zugriff auf Shared Storage etc. Die Überprüfung kann adhoc für eine bestimmte Komponente oder – am Arbeitsablauf einer Installation orientiert – für eine bestimmte Installations- oder Konfigurationsphase passieren.

Technisch gesehen handelt es sich beim CVU um ein Shell-Skript namens `cluvfy` mit einem Java-basierten Back-End. Die Installation und Ausführung des CVU geschieht mit dem Software-Owner der Clusterware-Software, also z. B. dem OS-Benutzer `grid`, `crs` bzw. `oracle`.

Für eine separate Installation muss zunächst ein Installationsverzeichnis angelegt werden, z. B. `~/cvu`. Dieses wird im Folgenden als CV-Home bezeichnet. Die von der Oracle-Seite heruntergeladene Zip-Datei wird einfach dorthin kopiert und ausgepackt. Auf Linux findet sich in einem der Unterverzeichnisse ein RPM-Paket namens `cvuqdisk`, das vor Benutzung des CVU noch installiert werden muss.

Wie gesagt wird im Rahmen einer Clusterware-Installation das CVU mit installiert. In diesem Fall ist das CV-Home mit dem Clusterware-Home identisch. Trotzdem empfiehlt es sich, immer die aktuellste Version von der Oracle-Seite zu benutzen, da auch das CVU ständig weiterentwickelt wird.

Testweise sollte sich das Werkzeug nun wie folgt aufrufen lassen (mit `/u01/app/grid` als Clusterware- bzw. CV-Home):

```
$ /u01/app/grid/bin/cluvfy

USAGE:
cluvfy [-help]
cluvfy stage {-list|-help}
cluvfy stage {-pre|-post} <stage-name> <stage-specific options>  [-verbose]
cluvfy comp  {-list|-help}
cluvfy comp  <component-name> <component-specific options>  [-verbose]
```

Im Folgenden wird eine Übersicht über die wichtigsten Tests im Umfeld von Hardware- und Betriebssystemkonfiguration gegeben. Bei einigen Tests müssen die geplante OSDBA-Gruppe und Oracle Inventory-Gruppe übergeben werden. In den Beispielen lauten diese jeweils `dba` bzw. `oinstall`.

In jedem Fall muss außerdem die Knotenliste als kommaseparierte Liste der Hostnamen der Cluster-Knoten übergeben werden. Um diese Liste nicht jedes Mal eingeben zu müs-

sen, kann alternativ das Schlüsselwort „all“ eingesetzt und über die Umgebungsvariable CV_NODE_ALL die tatsächliche Knotenliste hinterlegt werden.

Optional kann immer die Option `-verbose` ergänzt werden, um mehr Details zu erhalten.

SSH-Konnektivität und Benutzerrechte Damit wird getestet, ob für den angemeldeten Benutzer vom lokalen Knoten aus alle anderen Knoten ohne interaktiven Eingriff per SSH erreichbar sind:

```
$ cluvfy comp admprv -n <Knotenliste> -o user_equiv
  -sshonly
```

Ob außerdem die Benutzerrechte für eine Installation der Clusterware oder Datenbank-Software ausreichen, lässt sich mit folgenden Tests feststellen:

```
$ cluvfy comp admprv -n <Knotenliste> -o crs_inst
  -orainv oinstall
$ cluvfy comp admprv -n <Knotenliste> -o db_inst
  -osdba dba
```

Systemanforderungen Um Kernel-Parameter, Betriebssystem-Packages, OS-Benutzer und Gruppenzugehörigkeiten zu verifizieren, stehen folgende Aufrufe zur Verfügung:

- Für die Installation der Clusterware:

  ```
  $ cluvfy comp sys -n <Knotenliste> -p crs -osdba dba
    -orainv oinstall
  ```

- Für die Installation der Datenbank-Software:

  ```
  $ cluvfy comp sys -n <Knotenliste> -p crs -osdba dba
    -orainv oinstall
  ```

Da die Anforderungen je nach Software-Version unterschiedlich sind, muss für die Verifikation älterer Versionen zusätzlich der Parameter „-r“ gesetzt werden.

Diese und einige weitere Komponententests werden von dem Phasentest `hwos`[24] zusammengefasst, der eine abschließende Verifikation nach einer Hardware- und Betriebssysteminstallation durchführt:

```
$ cluvfy stage -post hwos -n <Knotenliste>
```

Ähnliche Tests überprüfen z. B. die Konfiguration vor oder nach der Clusterware-Installation bzw. vor der Installation der Datenbank-Software:

```
$ cluvfy stage -pre crsinst -n <Knotenliste>
  -c <OCR-Lokation> -q <Voting-Disk>
  -osdba dba -orainv oinstall
$ cluvfy stage -post crsinst -n <Knotenliste>
$ cluvfy stage -pre dbinst -n <Knotenliste> -osdba dba
```

Beginnend mit der Datenbankversion 11gR2 können einige der Tests im Falle eines Fehlschlags auch sofort eine Art Reparaturanleitung ausgeben, ein sogenanntes Fixup-Skript. Allerdings ist der Umfang dieser Maßnahmen bisher noch recht begrenzt, konkret werden Fixup-Skripte für die folgenden Konfigurationsaufgaben erzeugt:

24 `hwos` steht für Hardware + Operating System

- Erzeugen identischer Benutzer und Gruppen auf allen Cluster-Knoten und Setzen der Gruppenmitgliedschaften
- Setzen der benötigten Kernel-Parameter und Shell-Limits
- Setzen des Run-Levels

Mit der Option `-fixup` wird dies aktiviert, zusätzlich wird mit `-fixupdir` ein Zielverzeichnis für das Fixup-Skript angegeben.

3.8 Virtuelle Umgebungen

Auch wenn das Thema Virtualisierung in der IT insgesamt kein neues Thema ist – auf Betriebssystemen wie z/OS oder AIX ist diese Technologie unter dem Namen Partitionierung schon länger im Einsatz –, stellt es für die x86-basierten Prozessoren und deren Betriebssysteme – vorrangig MS-Windows und Linux – eine neuere Entwicklung dar, die erst 1998 mit der Gründung der Firma VMware richtig in Gang gekommen ist. Seitdem hat die Thematik in vielen Unternehmen Einzug gehalten. Der Einsatz virtueller Infrastrukturen läuft dabei häufig in mehreren Phasen ab:

- Am Anfang steht oder stand die Benutzung von Virtualisierungssoftware für Entwicklungs-, Test- oder Schulungsumgebungen, um hier auf teure Hardware verzichten zu können.
- Mit wachsendem Maß an Erfahrung und Vertrauen in die Technologie werden auch einige produktive Systeme auf virtuelle Umgebungen migriert, insbesondere Altsysteme, Niedriglastsysteme oder unkritische Anwendungen.

  Neben dem Wegfall von Legacy-Hardware steht auch hier der Konsolidierungsaspekt im Vordergrund, d. h., statt einer typischen Serverauslastung von 10 – 20 % laufen auf einer physischen Maschine anschließend mehrere virtuelle Maschinen, die eine deutlich höhere Gesamtauslastung bringen. Da der Stromverbrauch eines ausgelasteten Servers nur wenig höher als derjenige eines unausgelasteten Servers ist, führt dies zu erheblichen Einsparungen bei direkten und indirekten Stromkosten (z. B. Klimaanlage) und Server-Stellfläche.
- Nähert sich die Konsolidierungsphase dem Ende, tritt der Aspekt der Flexibilisierung in den Vordergrund. Ein virtueller Server kann – ggf. mithilfe von Templates – innerhalb von Minuten oder Stunden aufgesetzt und verfügbar gemacht werden, im Gegensatz zu den Tagen oder Wochen, die typischerweise für die Bereitstellung eines physischen Servers ins Land ziehen. Auch Migrationen werden mithilfe von Agilitätstechnologien wie beispielsweise VMwares *vMotion* oder Oracles *Secure Live Migration* wesentlich vereinfacht, geplante Auszeiten und Wartungsfenster können erheblich reduziert werden.

Gerade der letztgenannte Aspekt der Flexibilisierung der IT-Landschaft und Vereinfachung des Systemmanagements ist ein langfristiger und strategischer Vorteil virtualisierter Umgebungen. So wichtig die Kosteneinsparungen durch Serverkonsolidierung auch sind, bringen sie doch vor allem kurzfristig Entlastung. Auf lange Sicht steigen Hardware-Anforderungen, Lizenzmengen und damit IT-Kosten auch mit Virtualisierung proportional zu den Anforderungen. Die Virtualisierung spart gewissermaßen „nur“ einen konstanten Faktor ein, der typischerweise zwischen 5 und 10 liegt.

Im Folgenden werden gezielt die Vor- und Nachteile der Virtualisierung von Datenbanken näher beleuchtet und ein paar Tipps für den Betrieb gegeben.

3.8.1 Die Technik

Bei den verschiedenen Virtualisierungslösungen, die hier näher vorgestellt werden (Oracle VM, VMware vSphere und Citrix XenServer), handelt es sich um sogenannte Hypervisor, die direkt mit der Hardware – CPU, Memory, Disk, Netzwerk – kommunizieren. Bei den Kosten und Funktionen gibt es jedoch erhebliche Unterschiede:

Oracle VM ist kostenfrei und basiert auf dem Xen-Hypervisor, ergänzt durch einen Management-Server, der für die Administration zuständig ist, den *OracleVM Manager*. Wie bei allen Xen-basierten Lösungen ist auch eine Administration per Kommandozeile auf dem physischen Server möglich.

Der Zugriff auf den Management-Server geht ausschließlich über eine Weboberfläche (`http://hostname:8888/OVS`), was der Oracle-Philosophie entspricht, allerdings auch Probleme aufwirft. So kann z. B. auf das Gastbetriebssystem bei der Installation oder dem Start nur per VNC-Client zugegriffen werden. Diese Clients strotzen nicht gerade vor Performanz, und außerdem muss man nach dem Start über die Weboberfläche möglichst schnell den Client aufrufen, um entscheidende Schritte (z. B. BIOS-Zugriff oder Boot in den Single-User-Modus) nicht zu verpassen.

Auch die Vielzahl von notwendigen Accounts (Administrator für Weboberfläche, Agent auf dem Server, Utility-Server) trägt nicht gerade zur Vereinfachung bei. Wegen der zugrunde liegenden Xen-Technologie müssen in den virtuellen Maschinen entweder paravirtualisierte Betriebssysteme laufen – bei MS-Windows keine Option –, oder die physischen CPUs müssen Virtualisierungsunterstützung mitbringen (Intel VT bzw. AMD-V Technologie).

VMware vSphere ist sicherlich die teuerste Lösung. Allerdings bietet VMware auch den sogenannten ESXi-Server an, der kostenfrei ist und für eingebettete virtuelle Systeme verwendet werden kann. Im ESXi ist allerdings keine Management-Infrastruktur vorhanden. Administriert wird über den sogenannten vCenter Server. Dieser wird vorzugsweise über den MS-Windows-basierten sogenannten *vSphere Client* erreicht – alternativ, aber mit abgespecktem Funktionsumfang, auch über eine Weboberfläche.

Es gibt keine eigene Benutzerverwaltung, die Anmeldung am vCenter-Server erfolgt einfach mit einem Windows Active Directory-Konto. Über die Konsolenansicht kann eine VM vom ersten Moment an mitverfolgt werden. Durch die verwendete Vollvirtualisierung können auch auf CPUs ohne Virtualisierungsunterstützung native Betriebssysteme, z. B. MS-Windows, gefahren werden.

Citrix XenServer liegt preislich im Mittelfeld. Wie schon am Namen erkenntlich, handelt es sich wie bei Oracle VM um eine Xen-basierte Technologie, insofern gilt einiges vom oben Gesagten. Für das Management steht neben der Kommandozeile wiederum ein Windows-basierter Client namens *XenCenter* zur Verfügung, der aber nicht auf einen Management-Server zugreift, sondern direkt auf die ihm bekannt gegebenen XenServer und von dort alle Verwaltungsinformationen abzieht. Über den Client ist wiederum ein Zugriff im Textmodus oder per VNC möglich, Letzteres mit den oben beschriebenen Nachteilen.

Virtualisierung von Datenbanken

Natürlich gibt es einen Performance-Unterschied zwischen einem physischen Server und einer virtuellen Maschine. Typischerweise beträgt dieser Virtualisierungs-Overhead zwischen 5 % und 15 %, solange das System sich nicht in einem Grenzlastbereich befindet.

Dennoch muss an das Thema Datenbankvirtualisierung grundsätzlich mit Vorsicht herangegangen werden: Virtualisierung ist vor dem Hintergrund ungenutzter CPU-Kapazitäten entstanden, und in der Tat lassen sich in vielen Fällen durch Virtualisierung ehemals physische Systeme so lange konsolidieren, bis die CPU-Reserven ausgeschöpft sind (mit einem gewissen Puffer natürlich). Bei Datenbanken ist aber meistens die I/O-Kapazität der begrenzende Faktor. Bei einer Konsolidierung von Datenbanken in eine virtuelle Umgebung hinein muss also peinlich darauf geachtet werden, dass die I/O-Kapazitäten des physischen Storage-Systems noch mit den Anforderungen der virtuellen Maschinen mithalten können.

Umgekehrt formuliert, hat es in der Praxis schon viele Fälle gegeben, da ehemals physische Datenbank-Server mit eigenem Storage in eine zentrale, virtuelle Umgebung mit zentralem Storage konsolidiert wurden, CPU-, Memory- und Storage-Kapazitäten ausreichend dimensioniert wurden und die Performance anschließend trotzdem verheerend schlecht war. Anstatt dies sofort auf Unzulänglichkeiten der verwendeten Virtualisierungslösung zu schieben, sollten zunächst eine Analyse und ein Vergleich der I/O-Kapazitäten vor und nach der Konsolidierung vorgenommen werden. Sind die I/O-Kapazitäten nicht ausreichend, befindet sich das System sehr wahrscheinlich in einem Grenzlastbereich, wo durch Wartezustände (Stichwort: Disk Queues) deutliche Performance-Verschlechterungen auftreten.

Ungeachtet dessen, ist es in jedem Falle auch empfehlenswert, für virtuelle Maschinen mit kritischen Datenbanken Untergrenzen für die Zuteilung von CPU- und ggf. auch Memory-Ressourcen festzulegen, sogenannte *Reservations*. Der Grund dafür ist, dass der Hypervisor normalerweise eine Überallokation der physischen Ressourcen zulässt, d. h., die Summe der allen VMs zugewiesenen CPU- und Memory-Ressourcen überschreitet teilweise deutlich die physisch vorhandenen Ressourcen. Man spricht von einem „Overcommitment“. Dieser Effekt ist durchaus gewollt – um physische Ressourcen einzusparen – und meistens auch problemlos, da normalerweise nicht alle VMs gleichzeitig eine Lastspitze haben und der Hypervisor ungenutzte Ressourcen fast beliebig zwischen den VMs verschieben kann.

Problematisch wird es im Falle einer echten Ressourcenknappheit auf dem physischen Server. Hat eine VM keine Reservations für CPU und Memory gesetzt, bekommt sie u. U. extrem wenige CPU-Takte bzw. wird aufgrund der Speicherknappheit in ein starkes Paging des Gastbetriebssystems gezwungen („Ballooning“ bei VMware ESX-Server). Neben den offensichtlichen Performance-Auswirkungen gibt es noch zwei weitere unangenehme Effekte, die insbesondere im Oracle-Umfeld hier auftreten können:

- In Linux-VMs kann es bei starker Speicherknappheit zu einem Absturz der Oracle-Instanz kommen, da diese vom Betriebssystem gekillt wird. Dies geschieht durch den sogenannten OOM-Killer (Out-of-Memory-Killer), einen Prozess, der auf Linux-Systemen im Falle extremer Speicherknappheit anfängt, Benutzerprozesse zu killen und dadurch Speicher freizugeben. Vorzugsweise beseitigt der OOM-Killer einen einzigen Prozess mit hohem Speicherverbrauch. Auf Oracle-Servern ist dies

praktisch immer die Oracle-Instanz. In dieser Situation findet sich in der Datei `/var/log/messages` ein Eintrag wie dieser:

```
Sep 1 13:15:28 ora11gr2 kernel: Out of Memory: Killed process 13387 (oracle).
```

In diesem Fall sollte unbedingt eine Memory-Reservation gesetzt bzw. ein evtl. vorhandenes Memory-Limit entfernt oder erhöht werden.

- Im RAC-Umfeld überwacht der oprocd-Prozess u. a., ob genügend CPU-Zyklen zur Verfügung stehen, um z. B. Timeouts bei der Clusterware zu verhindern. Entzieht nun der Hypervisor der VM sehr viel CPU-Kapazität, „spürt" der oprocs-Prozess dies und erzwingt ggf. einen Reboot des Cluster-Knotens, also der VM. Auch hier muss durch eine CPU-Reservation gegengesteuert werden, die einen bestimmten Mindesttakt für die virtuellen CPUs vorgibt.

Zeitbegriff in virtuellen Maschinen

Eine Spezialität im Umgang mit virtuellen Systemen ist der Umgang mit zeitbezogenen Daten, insbesondere mit Statistiken und Performance-Metriken, die pro Zeiteinheit gemessen werden. Die meisten Betriebssysteme sind „Tick"-gesteuert, d. h., sie erwarten von der Hardware eine bestimmte Anzahl von Timer-Interrupts („Ticks") pro Sekunde. Die Gastbetriebssysteme in mehrere VMs auf einem physischen Server teilen sich diese Ticks, da der Tick immer nur diejenige VM erreicht, die zum betreffenden Zeitpunkt gerade läuft. Dadurch fehlen einer VM bereits nach kurzer Zeit Ticks, die Uhr innerhalb des Gastbetriebssystems fällt sofort und ständig zurück.

Die Benutzung eines NTP-Servers aus der VM heraus hat sich nicht als gute Lösung herausgestellt, da in Sekundenintervallen synchronisiert werden müsste, was wiederum erheblichen Overhead und Netzwerkverkehr erzeugen würde.

Als Good Practice gilt daher eher, nur die physischen Server per NTP zu synchronisieren. Die VMs können sich am physischen Server orientieren, indem die VMware-Tools bzw. Xen-Tools in der VM installiert werden. Diese protokollieren verlorene Ticks und senden sie an die VMs nach. Sollte auch dies nicht reichen, wird die Uhr in der VM entsprechend vorgedreht. Im Endeffekt bedeutet dies, dass zeitbezogene Messungen auf sehr kurzen Zeitskalen deutlich fehlerbehaftet sind, da gewissermaßen die Skalierung der Zeitachse in der VM unscharf wird. Auf größeren Zeitskalen, typischerweise ab dem Sekundenbereich, sind dagegen durch die oben beschriebenen Korrekturmechanismen entsprechende Messungen möglich.

Eine weitere Spezialität bei VMs ist die Möglichkeit, per Suspend und Resume vorübergehend angehalten zu werden, um zu einem späteren Zeitpunkt weiterzulaufen. Dies wird vor allem im Umfeld von Test- oder Schulungssystemen gerne genutzt. Im produktiven Umfeld sollte man damit sehr zurückhaltend sein, im RAC-Umfeld verbietet es sich praktisch. Der Grund dafür ist, dass von außen betrachtet sich die ganze Aktion wie ein Zeitsprung der VM darstellt.

Eine VM, die Teil eines RAC-Clusters ist, würde nach einem Suspend nicht mehr von den anderen Cluster-Knoten kontaktiert werden können und insofern aus dem Cluster ausgeschlossen („Node Eviction"). Der Rest des Clusters geht davon aus, dass der betroffene Knoten sich via I/O-Fencing selbst vom Storage-System abkoppelt (das ist entscheidend wichtig, da ja keine Kommunikation über den Interconnect mehr möglich ist) und neu startet, um anschließend wieder dem Cluster neu beizutreten.

Genau dies passiert aber nicht, sondern irgendwann erwacht der Knoten durch einen Resume-Befehl und macht weiter, als wäre nichts geschehen. Insbesondere hat er auch nichts von seinem Ausschluss aus dem Cluster mitbekommen und macht einfach mit dem nächsten I/O-Befehl weiter. Auch wenn nun nach wenigen Sekunden der Irrtum aufgeklärt wird, reicht diese kurze Zeitspanne wahrscheinlich aus, um unsinnige Daten auf das Storage zu schreiben und somit Datenkorruptionen zu erzeugen.

3.8.2 Virtualisierung und Service-Level

Ein wichtiges, aber sehr vielschichtiges Ziel besteht darin, mithilfe von Virtualisierungsverfahren eine höhere Verfügbarkeit und bessere Service-Level zu erreichen. Das Thema ist vor allem deshalb so vielschichtig und bisweilen verwirrend, da eigentlich erst eine genaue Definition der Begriffe Verfügbarkeit und Service-Level stattfinden muss. Dazu können u. a. folgende Fragen helfen:

- Wie ist Verfügbarkeit definiert? Welche Arten von Auszeiten sollen reduziert werden? Geht es vorrangig um ungeplante Auszeiten wie den Absturz eines Servers? Oder spielen auch geplante Auszeiten, also Wartungsfenster, eine Rolle? Hier muss auch gefragt werden, ob es sich um Auszeiten des physischen Servers oder einer VM handelt. Da Virtualisierung ja nur Mittel zum Zweck ist, kommt es aus Kunden- und Anwendersicht primär auf die VMs an. Allerdings gibt es hier Abhängigkeiten: So zieht ein Absturz des physischen Servers den Ausfall sämtlicher darauf laufender VMs nach sich.
- Für den Service-Level ist außerdem die Performance einer Anwendung oder Datenbank wichtig. Auch hier kann Virtualisierung helfen, indem Ressourcen sinnvoll zwischen VMs verschoben werden oder eine VM mit hohen Ansprüchen temporär auf einen anderen physischen Server migriert wird. Im Laufe dieses Abschnitts wird auch diskutiert, wie sich dies zu RAC- und Grid-Techniken in der Oracle-Welt verhält.
- Wo bzw. auf welcher Ebene werden Verfügbarkeit und Service-Level gemessen? Beim Ausfall einer physischen oder virtuellen Maschine macht es einen Unterschied, ob die Zeit bis zur Wiederherstellung des Servers, der Datenbank, der Anwendung oder des vereinbarten Service-Levels gemessen wird.

Geplante Auszeiten des Hosts, also z. B. ein Upgrade oder Patch des Hypervisors oder eine Migration auf andere physische Server, lassen sich mit Agilitätstechnologien wie VMware vMotion und Storage vMotion, Citrix XenMotion oder Oracle Secure Live Migration, vermeiden. Der betroffene physische Server wird „evakuiert“, d. h., die aktuell darauf befindlichen VMs werden vorübergehend auf einen anderen Rechner umgezogen (bei VMwares Storage vMotion sogar auf ein anderes Storage-System), die physische Plattform wird aktualisiert oder ausgetauscht, und zum Schluss werden die VMs wieder ohne Auszeit zurückmigriert. Da die komplette VM inklusive des Hauptspeicherinhalts migriert wird, ist praktisch keine Performance-Einbuße bemerkbar.

Gegen ungeplante Auszeiten helfen diese Techniken jedoch nur begrenzt weiter: Bei Benutzung eines zentralen Storage kann eine VM beim Absturz eines physischen Servers zwar auf einem anderen Server neu gestartet werden, dies macht z. B. VMware HA. Dabei handelt es sich aber faktisch um ein Cold-Standby. Bis z. B. der ursprüngliche Service-Level einer Oracle-Datenbank wiederhergestellt ist, müssen in der VM nicht nur das Betriebssystem gebootet und die Oracle-Instanz gestartet sein. Auch die Caches, ins-

besondere der Buffer-Cache und der Shared-Pool der Oracle-Instanz, müssen erst wieder gefüllt sein, bis ein normales Performance-Verhalten erreicht ist. Wird der Service-Level aus Sicht der Anwendung definiert, müssen auch z. B. Reconnect-Zeiten der Anwendung mit berücksichtigt werden.

Ein solches Cold-Standby ist insofern mit einem herkömmlichen active/passive-Cluster vergleichbar, außer dass die Cluster-Knoten virtuelle Systeme darstellen und das Cluster-Management vom Hypervisor realisiert wird. Wie bei jedem Cluster ist auch hier ein zentrales Storage (SAN oder NAS) obligatorisch.

Sowohl ungeplante Auszeiten im physischen und virtuellen Bereich als auch Wartungsfenster für die VMs – z. B. für Upgrades und Patches des Betriebssystems oder der Oracle-Software – lassen sich mit solchen Methoden offenbar nicht vermeiden. Eine Lösung hierfür ist in vielen Fällen Oracle RAC. Da es sich bei RAC um einen active/active-Cluster handelt, gibt es in vielen, wenn auch nicht allen Fällen die Möglichkeit eines *Rolling Upgrade*. Die Cluster-Knoten werden also nacheinander gewartet, zu einem gegebenen Zeitpunkt ist daher jeweils nur ein einziger Knoten nicht verfügbar, der oder die anderen Knoten arbeiten weiter. Der Unterschied zu einem normalen RAC besteht nur darin, dass die Knoten virtuelle statt physische Maschinen sind.

Für einen solchen virtualisierten RAC ist allerdings noch eine wichtige Randbedingung zu beachten: Die beteiligten VMs müssen stets auf verschiedenen physischen Servern laufen, damit nicht der physische Server zum „Single Point of Failure“ wird, was den Cluster ad absurdum führte. Bei VMware lässt sich das mithilfe sogenannter *Anti Affinity Rules* erzwingen. Diese Regeln sind notwendig, da der Hypervisor zum Zwecke der Lastverteilung ggf. autonom VMs zwischen physischen Servern verschiebt.

Ein active/active-Cluster setzt immer eine für die jeweilige Applikation spezifische Kommunikation der Cluster-Knoten voraus, bei Oracle z. B. im Rahmen der globalen Cache- und Sperrverwaltung. Daher ist eine solche Lösung grundsätzlich nicht mit den Mitteln des Hypervisors alleine realisierbar.

Daraus folgt ganz grundlegend, dass Virtualisierungstechnik und Oracle RAC weniger in Konkurrenz zueinander stehen als vielmehr komplementäre Lösungen darstellen.

Eine Zwischenlösung bieten Umsetzungen mit einer Hot-Standby-VM. Dabei werden entweder zwei VMs exakt synchron betrieben („Lockstepping“), oder die produktive VM wird in eine Standby-VM repliziert. Da die Synchronisation bzw. Replikation auf VM-Ebene erfolgt, ist insbesondere der gesamte Hauptspeicherinhalt, also auch die Caches der Oracle-Instanz, auf der Standby-Seite jederzeit gefüllt. Wenn die Anwender bzw. die Applikation auf die Standby-Seite schwenken müssen, ist insofern vom ersten Moment an dieselbe Performance verfügbar wie vorher.

Für VMware sind beispielsweise die Stratus ftServer für den Lockstep-Betrieb zertifiziert. VMware selbst bietet außerdem eine Technologie namens „Fault Tolerance“, die eine Replikation von VMs leistet. Auch Vizioncores vReplicator dreht sich um diese Thematik, dabei werden allerdings ausschließlich die virtuellen Festplatten, also die .vmdk-Dateien, repliziert. Es ist damit alternativ zu einer u. U. teuren Storage-Replikation zu sehen, schafft aber insofern keinen Hot-Standby.

Zur Sicherstellung der Performance einer Datenbank oder Anwendung ist schließlich die Dynamik virtueller Umgebungen sehr hilfreich. Sind Lastspitzen einer Datenbank bekannt, kann von vornherein eine zeitgesteuerte Live-Migration der virtuellen Maschine eingeplant werden. Ist dieser proaktive Ansatz nicht möglich, gibt es alterna-

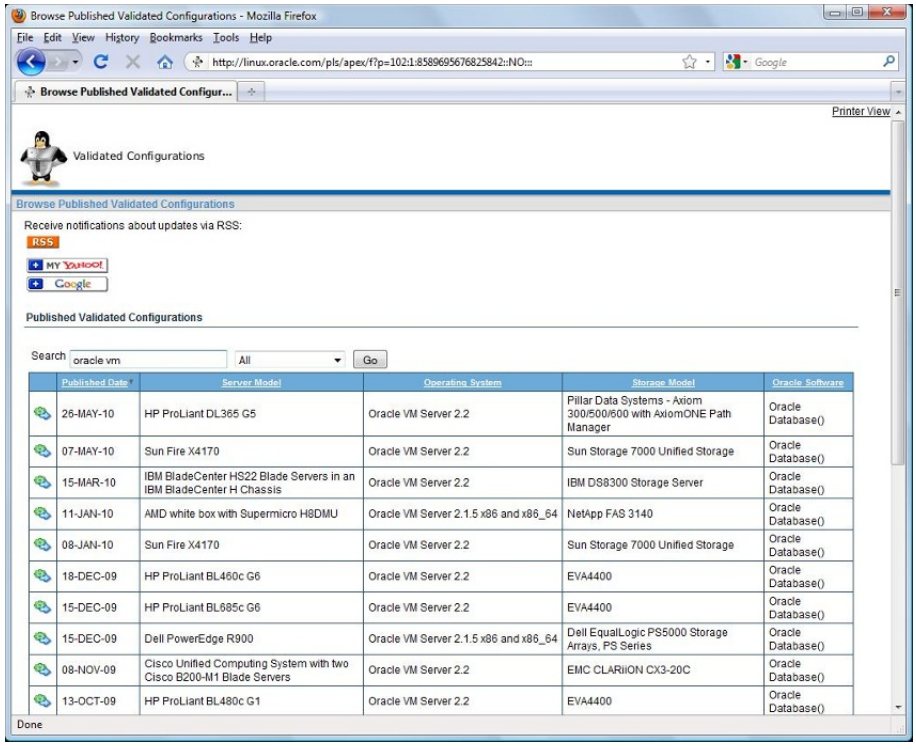
tiv natürlich die reaktive Methode, beispielsweise VMware DRS (Distributed Resource Scheduling), um bei Lastspitzen entsprechende Live-Migrationen adhoc anzustoßen.

Achtung: Beim Thema Live-Migrationen von Oracle-Datenbanken, insbesondere im RAC-Umfeld, muss genau auf die jeweils aktuelle Support- und Zertifizierungssituation geachtet werden! Solche Migrationen sind bisher ausschließlich für Non-RAC-Datenbanken und auch nur bei Benutzung von Oracle VM (Secure Live Migration) unterstützt.

3.8.3 Support und Zertifizierung

Eine entscheidende Frage auch und gerade beim Einsatz in virtuellen Umgebungen ist natürlich die des Supports. Spätestens bei einer produktiven Datenbank stellt sich die Frage, ob der Betrieb in einer virtuellen Maschine von Oracle formell unterstützt ist. Zur Drucklegung dieses Buches stellt sich die Situation wie folgt dar:

- Die Oracle-Datenbank ist, beginnend mit der Version 10.2.0.3 (für Oracle RAC erst ab 10.2.0.4 bzw. 11.1.0.7), für den Betrieb auf der Oracle-eigenen Virtualisierungsplattform *Oracle VM* zertifiziert. Voraussetzung ist außerdem ein 32-Bit- oder 64-Bit-Linux in der virtuellen Maschine. In diesem Falle ist also voller Support gegeben. Weitere Informationen sowie der jeweils aktuelle Stand hierzu sind auf der My Oracle Support-Seite Doc-ID 464754.1 beschrieben.



Published Date	Server Model	Operating System	Storage Model	Oracle Software
26-MAY-10	HP ProLiant DL365 G5	Oracle VM Server 2.2	Pillar Data Systems - Axiom 300/500/600 with AxiomONE Path Manager	Oracle Database()
07-MAY-10	Sun Fire X4170	Oracle VM Server 2.2	Sun Storage 7000 Unified Storage	Oracle Database()
15-MAR-10	IBM BladeCenter HS22 Blade Servers in an IBM BladeCenter H Chassis	Oracle VM Server 2.2	IBM DS8300 Storage Server	Oracle Database()
11-JAN-10	AMD white box with Supermicro H8DMU	Oracle VM Server 2.1.5 x86 and x86_64	NetApp FAS 3140	Oracle Database()
08-JAN-10	Sun Fire X4170	Oracle VM Server 2.2	Sun Storage 7000 Unified Storage	Oracle Database()
18-DEC-09	HP ProLiant BL460c G6	Oracle VM Server 2.2	EVA4400	Oracle Database()
15-DEC-09	HP ProLiant BL685c G6	Oracle VM Server 2.2	EVA4400	Oracle Database()
15-DEC-09	Dell PowerEdge R900	Oracle VM Server 2.1.5 x86 and x86_64	Dell EqualLogic PS5000 Storage Arrays, PS Series	Oracle Database()
08-NOV-09	Cisco Unified Computing System with two Cisco B200-M1 Blade Servers	Oracle VM Server 2.2	EMC CLARiiON CX3-20C	Oracle Database()
13-OCT-09	HP ProLiant BL480c G1	Oracle VM Server 2.2	EVA4400	Oracle Database()

Abbildung 3.12: Liste validierter Konfiguration auf der Oracle-Seite

- Eine Datenbank, die auf einer VMware-basierten virtuellen Maschine betrieben wird, unterstützt der Oracle-Support nur für Probleme, die für das verwendete Gastbetriebssystem als bekanntes Problem gelistet sind. Ansonsten kann der Support verlangen, dass der Kunde das Problem auf einer nicht virtualisierten Umgebung nachstellt. Ist dies nicht möglich, wird an den VMware-Support verwiesen. Der aktuelle Stand hierzu ist auf der My Oracle Support-Seite Doc-ID 249212.1 nachzulesen.
- Die Oracle-eigene Linux-Distribution *Oracle Enterprise Linux* ist auch in einer virtuellen Umgebung auf Basis von VMware vSphere oder Citrix XenServer Enterprise unterstützt. Dies gilt aber nicht pauschal für darauf installierte Oracle-Software. Insofern ist diese Aussage eher wichtig für anderweitige Server, die mit Oracle Enterprise Linux laufen sollen. Details hierzu sind auf der My Oracle Support-Seite Doc-ID 417770.1 ausgeführt.

Die bereits zu Kapitelbeginn erwähnten *Oracle Validated Configurations* stellen auch hier eine gute Hilfe dar, indem bereits getestete und validierte Kombinationen von Hardware, Betriebssystem (in diesem Falle innerhalb einer virtuellen Maschine) und Datenbankversion aufgelistet werden. Abbildung 3.12 zeigt einen Screenshot hierzu.