Nahezu jede Datenbank zieht irgendwann einmal um oder muss überarbeitet werden – sei es durch einen Storage-, Plattform-, oder Server-Wechsel, vielleicht sogar in eine virtuelle Maschine hinein, oder gleich bei einem Umzug in ein anderes Rechenzentrum. Für geschäftskritische Datenbanken fallen manchmal Spezial-Anforderungen an wie die Umstellung auf Unicode-Zeichensatz, Auszeiten im Minuten-Bereich oder Fallback-Möglichkeiten. Dieser Artikel stellt praxiserprobte Migrationsmethoden mit ihren jeweiligen Stärken, Schwächen und Randbedingungen vor.

Migrationsmethoden – von ASM bis Zeichensatz

Patrick Schwanke, CarajanDB GmbH

Export/Import geht immer

Die Werkzeuge „exp/imp“ und „Data Pump“ decken nahezu beliebige Migrations-Szenarien ab, insbesondere alle Versionssprünge zwischen Oracle 6 und 11.2. Data Pump erlaubt Sprünge zwischen 10.1 und 11.2.

Möchte man hingegen von einer bestehenden 11*g*-Datenbank migrieren, ist „exp“ zwar noch verfügbar, aber nicht mehr unterstützt. Spätestens für 11.2 empfiehlt sich ohnehin Data Pump, da in der Enterprise Edition Objekte standardmäßig mit Deferred Segment Creation erstellt werden und dadurch Tabellen, die noch nie Daten enthielten, vom alten „exp“ ohne Fehlermeldung einfach ignoriert werden (siehe MOS Doc-ID 960216.1).

Das benutzte Werkzeug muss der Version der jeweiligen Datenbank entsprechen. Bei einer Migration von 9.2 nach 11.2 muss man also mit einem 9.2-Export-Werkzeug starten und die Daten mit einem 11.2-Import-Werkzeug importieren. Der Hintergrund ist, dass zwar das von exp/imp beziehungsweise Data Pump benutzte Dateiformat abwärtskompatibel ist, nicht aber die Werkzeuge selbst. Häufig ist es sinnvoll, einfach das jeweilige Werkzeug aus dem entsprechenden Oracle-Home des Datenbankservers aufzurufen.

Nachteilig ist in jedem Fall der relativ lange Zeitraum für Export, Übertragung und Import, der eine entsprechend lange Datenbank-Auszeit bedeutet. Beide Werkzeuge sind insbesondere für LOB-Spalten sehr langsam. Der Import läuft typischerweise etwa zwei bis drei Mal so lange wie der Export. Bei Datenbanken im TB-Bereich landet man so schnell bei Auszeiten bis zu mehreren Tagen.

Export, Übertragung der Daten und Import können parallelisiert werden, wobei gleichzeitig das Disk-I/O zum Schreiben und Lesen der Dump-Dateien entfällt. Bei exp/imp geht dies auf Unix-/Linux-Systemen, indem jeweils eine Named Pipe als Dump-Datei angegeben wird und ein weiterer Prozess per SSH die Daten aus der einen Named Pipe in die andere schreibt. Bei Data Pump gibt es hierfür einen eigenen Parameter namens „NETWORK_LINK“. Data Pump erlaubt für die Enterprise Edition auch die Parallelisierung des Exports beziehungsweise Imports selbst mithilfe der PARALLEL-Klausel. Achtung: Die beiden Klauseln „NETWORK_LINK“ und „PARALLEL“ lassen sich erst ab dem Patch-Level 10.2.0.4 kombinieren (siehe MOS Doc-ID 5972794.8).

Rolling-Migration geht meistens

Nahezu beliebige Migrationen decken Werkzeuge ab, die eine rein logische Datenbank-Replikation leisten, also die Transaktionen aus der alten Datenbank rekonstruieren, diese als SQL-Befehle an die neue Datenbank schicken und somit die Datenbestände synchronisieren. Die gesamte Physik der Datenbank (Version, Plattform, Storage, Tablespace-Layout, Tabellenpartitionierung etc.) kann völlig unterschiedlich sein, daher auch die Flexibilität dieser Methode.

Der Autor hat praktische Erfahrungen mit zweien dieser Werkzeuge: SharePlex on Oracle von Quest Software und Oracle GoldenGate. Beide sind von der Funktionsweise her sehr ähnlich, wobei SharePlex als das ältere und auch reifere der beiden Produkte bezeichnet werden kann. GoldenGate hat dagegen den Vorteil, dass es auch außerhalb der Oracle-Welt arbeiten, also zum Beispiel von Oracle zu SQL-Servern replizieren kann.

Die zusätzlich anfallenden Lizenzkosten sowie die höhere Komplexität der Migration muss man gegen zwei Vorteile abwägen, nämlich Auszeit- und Risiko-Minimierung: Zum einen liegt die Datenbank-Auszeit unabhängig von der Größe oder Transaktionslast der Datenbank typischerweise im Minuten-Bereich. Zum anderen erhält man die Möglichkeit, auch Tage nach der Migration bei auftretenden Problemen ohne Datenverlust wieder auf die alte Datenbank zurückschwenken zu können.

Abbildung 1 zeigt den groben Ablauf. Die neue Datenbank legt man zunächst als leere Datenbank an, etwa mit dem Database Configuration Assistant (DBCA). Anschließend startet man die Replikation von der bestehenden in die neue Datenbank und puffert diese sofort, da die Zieldatenbank ja noch leer ist (Schritt 1).

Nun überträgt man einen möglichst aktuellen und vor allem konsistenten Datenbestand, indem man die Werk-

zeuge „exp“ beziehungsweise „Data Pump Export“ mit dem Parameter „FLASHBACK_SCN“ aufruft. Nach erfolgtem Import in die neue Datenbank müssen bei SharePlex bestimmte Datenbank-Objekte wie Trigger oder Datenbank-Jobs dort zunächst deaktiviert werden, damit replizierte Änderungen nicht doppelt ankommen. Die Replikation synchronisiert sich anschließend mit der benutzten „FLASHBACK_SCN“ und fährt alle zwischenzeitlich gepufferten Transaktionen nach (Schritt 2). Bis hierhin ist noch keinerlei Auszeit aufgetreten.

Während der nun folgenden kurzen Datenbank-Auszeit aktiviert man Trigger und Datenbank-Jobs auf der neuen Datenbank, deaktiviert sie auf der alten und dreht die Replikationsrichtung, um sich eine Fallback-Möglichkeit ohne Datenverlust zu sichern für den Fall, dass erst nach einigen Stunden oder Tagen Probleme mit der neuen Datenbank auftreten (Schritt 3). Sodann dürfen sich die Benutzer und Anwendungen auf der neuen Datenbank anmelden.

Sowohl GoldenGate als auch SharePlex haben Restriktionen hinsichtlich der unterstützten Oracle-Versionen, Datentypen und bestimmter Funktionen innerhalb der Datenbank. Für GoldenGate ist ein entsprechendes Check-Skript unter [1] verfügbar. Bei SharePlex fragt man am besten beim Hersteller Quest Software nach.

Reine Server-Migrationen

Für den Umzug auf einen anderen Server genügt es prinzipiell, auf dem neuen Server ein Oracle-Home zu installieren, die Anwendung zu stoppen, mittels RMAN ein Backup der Datenbank zu erstellen und dieses auf dem neuen Server einzuspielen. Wem diese Auszeit zu lang ist, der kann auch auf dem neuen Server eine Physical-Standby-Datenbank aufbauen, die ja einem sehr aktuellen Backup entspricht.

Reine Storage-Migrationen

Läuft die Datenbank bereits auf Basis von Oracle Automatic Storage Management (ASM) und möchte man lediglich auf einen anderen Storage-Hersteller, Storage-Turm und/oder andere Platten migrieren, lassen sich sehr geschickt die Rebalancing-Fähigkeiten von ASM ausnutzen. Bei dieser Methodik entsteht keine Auszeit, sondern bloß zwischenzeitlich eine höhere I/O-Last auf dem Datenbank-Storage.

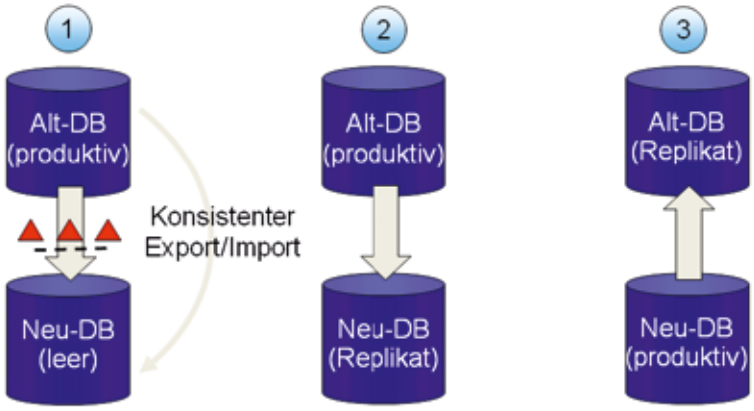

Abbildung 1: Ablauf einer Rolling-Migration

Nach Bereitstellung und Stamping der neuen Platten auf dem Datenbank-Server fügt man für jede ASM-Diskgruppe in einem ersten Schritt alle neuen Platten der Diskgruppe hinzu und entfernt in einem zweiten Schritt alle alten Platten:

```
SQL> ALTER DISKGROUP data ADD
DISK 'ORCL:ASMDISKNEW';
SQL> ALTER DISKGROUP data DROP
DISK 'ASMDISKOLD';
```

Das automatische Rebalancing sorgt jeweils für eine entsprechende Neuverteilung der Daten. Mit der View „V$ASM_OPERATION“ in der ASM-Instanz lässt sich überprüfen, ob beziehungsweise wann das Rebalancing voraussichtlich beendet ist. Anschließend kann man die alten Platten abhängen.

Versions- und Storage-Wechsel

Mit dem Database Upgrade Assistant (DBUA), der mit der Datenbank-Software installiert wird, lässt sich bequem ein Versions-Upgrade bestehender Datenbanken durchführen. Der DBUA prüft vor der Migration automatisch alle erforderlichen Voraussetzungen. Seit der Zielversion 11.1 kann zusätzlich zum reinen Versions-Upgrade auch eine Storage-Migration durchgeführt werden, insbesondere nach ASM. Außerdem ist ein Wechsel von 32-bit- auf 64-bit-Oracle-Software möglich.

Die konkrete Upgrade-Zeit und damit Auszeit hängt von den installierten Datenbank-Optionen und -Komponenten ab. Typische Zeiten liegen zwischen 30 und 90 Minuten. Je weniger Optionen installiert sind (Text-, Spatial-Option, JVM, XML DB etc.), desto schneller das Upgrade. Es handelt sich um ein In-Place-Upgrade, daher ist die einzige Rückfallmöglichkeit das in [2] beschriebene, manuelle Downgrade, für das eine ähnliche Auszeit wie das eigentliche Upgrade veranschlagt werden kann.

Vor dem Beginn des Upgrade sollte man sich vergewissern, dass der geplante Versionssprung überhaupt möglich beziehungsweise unterstützt ist, ob gegebenenfalls vorher ein Patch-Release eingespielt werden muss oder sogar ein Upgrade in zwei Schritten durchgeführt werden muss. Abbildung 2 zeigt die möglichen Schritte beim Upgrade auf Version 11.2; im Zweifelsfall sollte man die offizielle Dokumentation [3] der neuen Datenbank-Version zu Rate ziehen, in der alle unterstützten Versionssprünge beschrieben sind.

Im Vorfeld wird die neue Datenbank-Software installiert und gegebenenfalls gepatcht. Nach dem Aufruf der neuen Softwareversion durch den

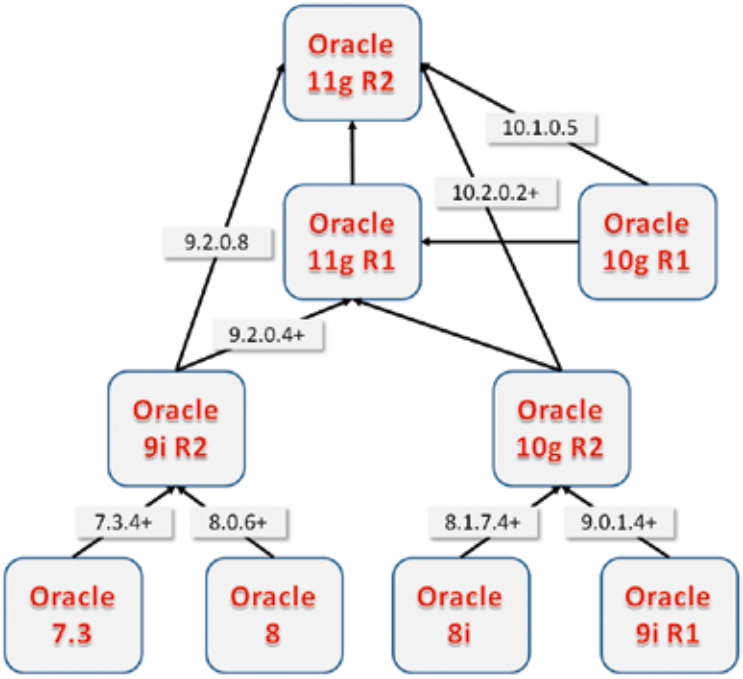

Abbildung 2: Mögliche Upgrade-Pfade mit dem DBUA

DBUA ist der Ablauf Wizard-gesteuert und in weiten Teilen selbsterklärend beziehungsweise in [3] beschrieben. Für umfangreiche Umgebungen bietet der DBUA auch ein Silent-Upgrade.

Nach erfolgtem Upgrade ist der Server-Parameter „compatible" der Datenbank noch auf den alten Versions-Stand gesetzt. Die neue Datenbank nutzt also noch keine neuen Features. Der Pferdefuß dieser Methode ist, dass ein Rückfall auf die alte Version (Downgrade) auch nur möglich ist, solange dieser alte „compatible"-Wert gesetzt ist. Probleme nach dem Upgrade zeigen sich manchmal aber erst nach dem Heraufsetzen des „compatible"-Werts. Das Heraufsetzen ist jedoch eine Einbahnstraße und macht ein nachfolgendes Heruntersetzen oder Downgrade unmöglich. Ohne vorheriges Backup sollte man diesen Weg daher nicht beschreiten.

Plattform- und Storage-Wechsel

Ursprünglich aus dem Data-Warehouse-Bereich stammend bietet die sogenannte „Transportable Tablespace"-Technik (TTS) auch einen Migrationspfad. Seit der Version 10.1 ist hiermit das Kopieren von Tablespaces über die Server-, Plattform- und Storage-Grenzen hinweg möglich, wobei das Backup-Werkzeug „RMAN" die Konvertierung der Datendateien übernimmt.

Mit „Transportable Database" (TDB) geht dies ab der Version 10.2 auch für die internen Tablespaces „SYSTEM" und „SYSAUX", allerdings nur für Plattform-Wechsel innerhalb derselben Endianness-Gruppe. So ist beispielsweise mit TDB ein Wechsel zwischen Linux und Windows (Little-Endian) oder zwischen AIX und HP-UX (Big-Endian) möglich, nicht aber zwischen AIX und Linux.

Jede Datenbank listet in der View „V$TRANSPORTABLE_PLATFORM" alle für TTS in Frage kommenden Ziel-Plattformen auf (dies sind nahezu alle Oracle-Plattformen); die View „V$DB_TRANSPORTABLE_PLATFORM" zeigt alle für TDB unterstützten Ziel-Plattformen, also alle Plattformen mit derselben Endianness. In jedem Falle muss zumindest die Quell-Datenbank als Enterprise Edition lizenziert sein.

Da es sich letztlich um das Erstellen einer physischen Kopie der bestehenden Datenbank nebst Konvertierung auf eine andere Plattform handelt, sind mit dieser Methode weder Versionssprünge noch Zeichensatz-Migrationen oder Reorganisationen möglich. Für die Dauer der Übertragung ist die alte Datenbank „Read-only". Die Auszeit ist also proportional zur Datenbank-Größe, aber deutlich geringer als bei exp/imp beziehungsweise Data Pump. Beide Methoden sind im Detail unter [4] und [5] beschrieben. Abbildung 3 skizziert den Ablauf. Mit einem kleinen Trick können in einem Aufwasch die Datendateien der alten Datenbank gelesen, konvertiert und weggeschrieben werden. Anstatt die alten Datendateien per SCP oder FTP auf den neuen Server zu übertragen, werden sie einfach per NFS exportiert und auf dem neuen Server gemountet – dort wird die Konvertierung angestoßen.

Reine Zeichensatz-Migrationen

Bei Zeichensatz-Änderungen dürfte es sich praktisch immer um Migrationen nach Unicode handeln. Eine solche Umstellung ist entweder innerhalb der bestehenden Datenbank oder durch Aufbau einer neuen Datenbank möglich. Letzteres ist insbesondere dann interessant, wenn weitere Änderungen wie Versions-Upgrade oder Plattform-Wechsel anstehen (also keine reine Zeichensatz-Migration) oder geringe Auszeiten beziehungsweise eine Fallback-Möglichkeit angestrebt werden. In diesem Falle muss man auf einen Export/Import oder eine Rolling-Migration zurückgreifen.

Beim Neuaufbau ist es wichtig, die Tabellen auf der Zieldatenbank zunächst ohne Daten zu erstellen, aber mit Längensemantik „Char" anstelle der Standardeinstellung „Byte", beispielsweise „VARCHAR2(50 CHAR)" anstelle von „VARCHAR2(50 BYTE)", analog für den Datentyp „Char". Hierfür kann man entweder einen Export/Import der Tabellen ohne Inhalte fahren und anschließend mit MOS Doc-ID 313175.1 die Längensemantik anpassen oder man lässt sich auf der alten Datenbank mittels „DBMS_METADATA" oder einem Werkzeug wie „Toad" ein CREATE-Skript für alle Tabellen ohne Semantik-Angabe erstellen und führt diese auf der neuen Datenbank wie folgt aus:

```
SQL> ALTER SESSION SET NLS_
LENGTH_SEMANTICS = ‚CHAR‘;
SQL> -- erstelltes Skript, zum
Beispiel
SQL> CREATE TABLE (
       id NUMBER PRIMARY KEY,
       textcol VARCHAR2(50));
```

Anschließend werden per Import die Daten eingefügt, mit den Klauseln „IGNORE=Y“ beim konventionellen Import beziehungsweise „TABLE_EXISTS_ACTION=APPEND“ beim Datapump-Import. Die Alternative, also die In-Place-Umstellung einer Datenbank auf Unicode, geht dagegen mit dem sogenannten „CSALTER“-Werkzeug vonstatten, das in der Datenbank-Software enthalten ist. Wichtig zu wissen ist, dass dieses Werkzeug ausschließlich Data-Dictionary-Inhalte, also die Metadaten, in den neuen Zeichensatz konvertiert. Benutzerdaten, insbesondere Inhalte der Anwendungstabellen, werden nicht verändert.

Nur wenn sämtliche Benutzerdaten durch die Zeichensatzänderung unverändert bleiben, gibt das „CSALTER“-Werkzeug grünes Licht und ändert den Zeichensatz.

Alle nicht konformen Daten müssen zuvor exportiert, gelöscht und nach der Umstellung wieder importiert werden, wobei die Logik von exp/imp beziehungsweise Data Pump für die Konvertierung sorgt. Diese nicht konformen Daten zu finden ist Aufgabe des „CSSCAN“-Werkzeugs.

Bei einer Unicode-Konvertierung sind praktisch alle Zeilen betroffen, in denen Nicht-ASCII-Zeichen vorkommen. Je größer der Anteil dieser Daten, desto länger der Aufwand und damit die Auszeit mit dieser Methode. Im Extremfall müsste die ganze Datenbank exportiert und importiert werden. In diesen Fällen kommt alternativ nur eine Rolling-Migration in Frage. In jedem Falle sollte man vorbereitend oder als Teil der Auszeit die „VARCHAR2“- und „CHAR“-Spalten wie oben beschrieben auf „Längensemantik=‘CHAR‘“ konvertieren. Details zu dieser Methode stehen unter [6].

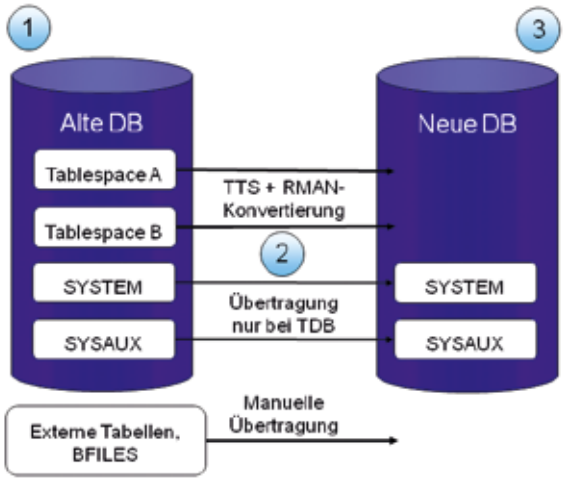

Abbildung 3: Migration mit Transportable Tablespace oder Transportable Database

P2V und V2P

Das Überführen einer bestehenden Datenbank in eine virtuelle Maschine oder auch eine entsprechende Migration zurück lässt sich auf einen der bereits besprochenen Fälle zurückführen. Für kritische Datenbanken empfiehlt der Autor, die Datenbank so aufzubauen, dass sie sowohl von einem physischen als auch von einem virtuellen Server aus gemountet werden kann. Dafür gibt es zwei Wege:

- Die VM greift direkt auf physischen Platten zu – beispielsweise mit VMwares Raw Device Mappings (RDM) – und nutzt diese mittels Oracle ASM für die Datenbank-Dateien.
- Die VM greift direkt auf NFS-Filer zu und nutzt diese mittels NFS-Mounts – ab Oracle 11*g* besser mit Oracle Direct NFS – für die Datenbank-Dateien.

Im einfachsten Falle handelt es sich um eine reine Migration von „physisch“ nach „virtuell“. Liegt die Datenbank wie oben beschrieben in ASM oder auf NFS, kann sie unmittelbar aus der VM heraus gemountet und geöffnet werden. Ist dies nicht der Fall, etwa bei Nutzung virtueller Platten in „.vmdk“- oder „.vhd“-Dateien, greifen die genannten Szenarien für reine Server-Migrationen. Auch wenn die Virtualisierung mit weiteren Änderungen kombiniert wird, greifen die bereits beschriebenen Szenarien analog.

Referenzen

[1] GoldenGate Checkskript: MOS Doc-ID 1298562.1

[2] Downgrade nach DBUA-Upgrade: Oracle Database Upgrade Guide der jeweiligen Zielversion, Kapitel „Downgrading a Database“

[3] Mögliche Upgrade-Pfade: Oracle Database Upgrade Guide der jeweiligen Zielversion, Kapitel „Preparing to Upgrade“

[4] Plattformmigration mittels Transportable Tablespace: http://www.oracle.com/technetwork/database/features/availability/maa-wp-11g-platformmigrationtts-129269.pdf

[5] Plattformmigration mittels Transportable Database: http://www.oracle.com/technetwork/database/features/availability/maa-wp-10gr2-platformmigrationtdb-131164.pdf

[6] In-Place-Zeichensatzmigration: Oracle Database Globalization Support Guide, Kapitel „Character Set Migration“

Patrick Schwanke
CarajanDB GmbH
patrick.schwanke@carajandb.com