Eine Datenbank im Quantensprung

Patrick Schwanke
Quest Software
Köln

Dierk Lenz
Herrmann & Lenz Services GmbH
Burscheid

Schlüsselworte:

Migration, Upgrade, Unicode, Auszeit, Downtime, Shareplex, Flashback

Einleitung

Eine Datenbank im Multi-Terabyte-Bereich und mit hoher Transaktionslast, die weltweit im Zugriff ist und ohne spürbare Auszeit sowie mit minimalem Risiko auf eine andere Plattform und anderes Storage migriert werden soll?

Außerdem müssen in diversen Tabellen NCHAR-/NCLOB-Datentypen in CHAR-/CLOB-Datentypen umgewandelt werden, wobei der Standardzeichensatz auf Unicode geändert wird. Und einige größere Tabellen sind zu partitionieren.

Ist das überhaupt möglich, gar in einem einzigen Schritt? Dieser Projektbericht zeigt, dass es geht und was dabei beachtet werden muss.

Wie kam das alles?

Ein langjähriger Kunde trat im Juli 2009 an uns heran mit der Bitte um Unterstützung bei einer Datenbankmigration.

Wie sich bei der ersten Bestandsaufnahme herausstellte, handelte es sich um eine weltweit genutzte Datenbank, die Konfigurationen, Beschreibungen sowie die Historie jedes ausgelieferten Produkts enthält. Etwa 20000 Anwender sind an diese Datenbank angeschlossen. Auszeiten sind praktisch nur bei aufeinander folgenden Feiertagen möglich, also zu Ostern oder Weihnachten.

Technisch handelte es sich um eine 10.2.0.4-Datenbank auf HP-UX mit 7 TB Datenvolumen. Die Änderungsrate variierte zwischen 100GB und 300GB Redolog-Daten pro Tag. Neben Dutzenden Schnittstellen und angrenzenden Systemen bildet ein Oracle BEA Weblogic Server die Hauptanwendung.

Ausschlaggebend für die Migration waren Performance-Probleme seitens des Weblogic-Servers, deren Ursache Oracle in den verwendeten NCHAR-/NCLOB-Datentypen ausmachte. Da die Datenbank mit ISO 8859-1 als Standardzeichensatz aufgebaut war, waren diese N-Datentypen irgendwann als Workaround eingeführt worden, um insbesondere asiatische und arabische Zeichen und Dokumente speichern zu können.

Quo vadis – und vor allem: Wie kommt man dahin?

Neben dieser Datentyp- und Zeichensatzproblematik sprachen noch weitere Gründe für eine Migration der Datenbank:

- Aufgrund einer Kooperation mit Intel lief bereits ein großer Teil der Unternehmens-IT auf Intel-Hardware. Eine entsprechende Migration von HP-UX nach Linux war für dieses System aber aufgrund der Auszeitbeschränkungen bisher nicht durchgeführt worden.
- Das starke Wachstum in der Vergangenheit ließ die Partitionierung einiger sehr großer Tabellen sinnvoll erscheinen. Aber auch für diese Reorganisation fehlte bisher die notwendige Auszeit.
- Auch das benutzte Storage-System stieß allmählich an seine Grenzen.
- Fast die einzige Konstante bei der Migration sollte die Datenbankversion sein – auf der Zielseite wurde hier auch 10.2.0.4 anvisiert.

Da Auszeiten Mangelware sind, war eine All-In-One-Migration angestrebt, spricht alle Aspekte sollten auf einen Schlag umgesetzt werden. Als weitere Rahmenbedingungen sind zu nennen:

- Vor der Migration sollten realistische Testläufe mit der neuen Datenbank stattfinden.
- Bis zu 2 Wochen nach der Umstellung sollte (für Notfälle, sprich unvorhergesehene Probleme) ein schnelles Fallback auf die alte Datenbank möglich sein, wohlgemerkt: ohne Datenverlust!

Methode	Version	Platt-form	Storage	Zeichen satz	Reorg.	Auszeit	In-Place	Fallback v/n	Komplexität
Standard-Upgrade	X	–	O	–	–	kurz	X	X/O	gering
Export / Import	X	X	X	X	X	sehr lang	–	X/X	gering
Transportable Tablespace	–	X	X	–	–	lang	–	X/X	mittel
Transportable Database	–	O	X	–	–	lang	–	X/X	gering
ASM Rebalancing	–	–	X	–	–	keine	–	X/X	gering
Rolling Upgrade	X	O	X	–	–	sehr kurz	–	X/X	hoch
„Rolling Migration“	X	X	X	X	X	sehr kurz	–	X/X	hoch

Abbildung 1: Upgrade- und Migrationsmethoden im Vergleich

Die Oracle-Datenbank bietet – auf dem Versionsstand 10gR2 – eine ganze Palette von Upgrade- und Migrationsmethoden mit jeweils spezifischen Vor- und Nachteilen an, von denen Abbildung 1 die den Autoren geläufigen auflistet.

Angesichts der beschriebenen Rahmenbedingungen der Migration kamen aber dennoch nur ein Export/Import (mit konventionellem exp/imp oder mit Datapump) und eine sogenannte „Rolling Migration“ infrage. Letzteren Begriff sucht man in der Oracle-Dokumentation vergebens. Es ist vergleichbar mit der Oracle-Technologie „Rolling Upgrade“, allerdings wird statt der Logical Standby-Technik eine unabhängige Replikationstechnologie verwendet – in diesem Falle Quest Shareplex.

Dadurch können Quell- und Zieldatenbank unterschiedliche Plattformen, Zeichensätze, Datentypen, Tabellenpartitionierungen, sonstige physische Strukturen etc. aufweisen.

Hochrechnungen ergaben, dass ein Export/Import selbst mit Parallelisierung mehr als eine Woche und damit mehr als eine Woche Auszeit benötigt hätte, insbesondere aufgrund der vielen LOB-Spalten, die beim Export/Import (sowohl klassisch als auch Datapump) sehr ineffizient behandelt werden. Da dies als nicht tragbar eingestuft wurde, fiel die Wahl auf eine Rolling Migration mit Hilfe von Shareplex als Replikations-Software.

Auch bei dieser Methode ist aber eine Initialbefüllung der neuen Datenbank mit Hilfe von Export/Import notwendig. Um extreme Wartezeiten zu vermeiden, hat die Firma Herrmann & Lenz ein für LOB-Spalten optimiertes Werkzeug, die HLTablePump, entwickelt, die – im Gegensatz zu den Oracle-Standardwerkzeugen – ein Array-Insert auch für Tabellen mit LOB-Spalten durchführt und dadurch um Faktoren schneller arbeitet.

Vorbereitungen

Als Erstes wurde die neue Datenbank auf der neuen Hardware mit Linux-Betriebssystem und Anbindung an das neue Storage-System erstellt. Hierfür wurde ganz normal der DBCA (Database Configuration Assistant) verwendet, wobei Unicode (AL32UTF8) als Standardzeichensatz eingetragen wurde.

Im nächsten Schritt wurden die Tablespaces und User erstellt. Eine Herausforderung bei Unicode-Migrationen besteht darin, dass alle Nicht-ASCII-Zeichen – also auch gewöhnliche deutsche Umlaute – zwei oder mehr Bytes benötigen. Datentypen wie VARCHAR2(50) sehen für die Speicherung aber lediglich 50 Byte vor. Im Unicode-Umfeld muss daher diese standardmäßige „Byte-Semantik“ durch eine „Zeichen-Semantik“ ersetzt werden.

Wie eine solche Datentypkonvertierung ablaufen kann, wird im Vortrag beschrieben.

Die Tabellen mit den angepassten Datentypen wurden nun als leere Tabellen vorerstellt, ggf. als partitionierte Tabellen.

Erst jetzt startet die Shareplex-Replikation, die aus den Online- (oder archivierten) Redologs der Produktionsdatenbank alle Changes rekonstruiert und in Form von SQL-Kommandos an die Zieldatenbank sendet. Da diese noch leer ist, werden die SQLs von der dortigen Shareplexinstanz zunächst gepuffert.

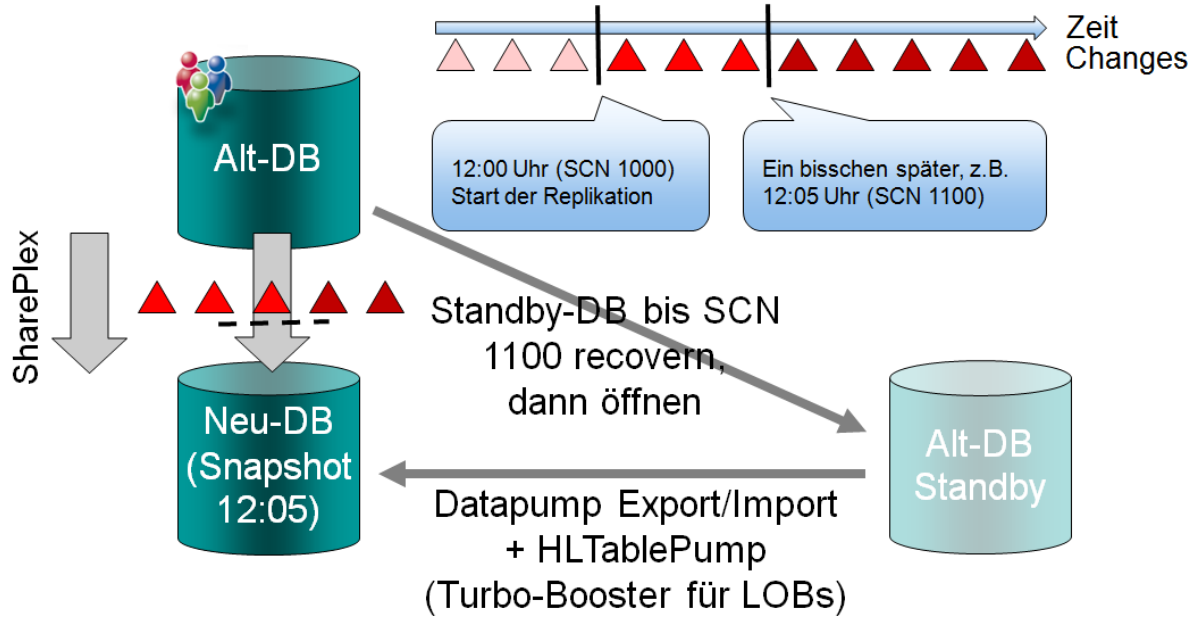

Abbildung 2: Initialbefüllung der neuen Datenbank

Jetzt werden die bestehenden Daten übertragen, wobei das entscheidende Merkmal ein konsistenter Datenexport ist, wie er seit der Oracle-Version 9i beispielsweise mit dem exp- oder expdp-Parameter „flashback_scn“ erzwungen werden kann. Der Export nutzt dabei – genau wie eine Flashback-Abfrage – Undo-Informationen. Dies ist sehr bequem, benötigt aber bei einem langlaufenden Export auch sehr viel Undo-Tablespace.

Da ohnehin eine Physical Standby-Datenbank als Desaster Recovery-System vorhanden war, haben wir diese zweckentfremdet, bis zu einem bestimmten Zeitpunkt recovered, geöffnet und anschließend von dort per Export/Import in die neue Datenbank übertragen. Auch so ist ein konsistenter Datenstand gegeben, nämlich der des Recovery-Zeitpunkts (siehe Abbildung 2).

Die spezielle Herausforderung der Datentypkonvertierung erforderte hier einige ungewöhnliche Maßnahmen, da eine Konvertierung von NCLOB- nach CLOB-Spalten sowohl vom konventionellen Import als auch von Shareplex mit einer charakteristischen Fehlermeldung verweigert wird: `ORA-24806: LOB-Form stimmt nicht überein`.

Das von Herrmann & Lenz entwickelte Werkzeug HLTablePump zur optimierten LOB-Übertragung lieferte hier Abhilfe. Die Firma Quest lieferte außerdem einen entsprechenden Patch für das Shareplex-Werkzeug.

Indem der gewählte Recovery-Zeitpunkt (genauer gesagt, die System Change Number SCN) an Shareplex mitgeteilt wird, löscht dieses diejenigen gepufferten SQL-Kommandos, die jetzt nicht mehr benötigt werden, weil die betreffenden Daten schon im Rahmen der Initialbefüllung in die neue Datenbank gekommen sind. Die restlichen zwischenzeitlich gesammelten SQLs werden nachgefahren (was aufgrund des hohen Transaktionsvolumens wiederum einige Tage dauerte). Anschließend hat man eine fertige Replikation von der bisherigen auf die neue Datenbank.

Diese Replikation arbeitet asynchron, so dass die bestehende Produktion auf keinen Fall unter irgendwelchen Problemen mit der neuen Datenbank leiden muss. Der zeitliche Versatz der beiden Systeme beträgt typischerweise wenige Sekunden, kann aber auch mal größer werden, wenn

beispielsweise Batch-Jobs laufen und die Changes nicht schnell genug „rübergeschoben“ werden können.

Applikationstests

Diese Situation haben wir zunächst für einige Testläufe seitens der Applikation genutzt, wobei es wichtig war, auch Daten verändernde Tests durchführen zu können. Hierfür war die seit Oracle 10g und mit der Enterprise Edition verfügbare „Flashback Database“-Funktion eine große Hilfe. Zunächst musste das Flashback-Logging auf der neuen Datenbank eingeschaltet werden:

```
SQL> ALTER SYSTEM SET db_flashback_retention_target=<dauer in minuten>;
SQL> ALTER DATABASE ARCHIVELOG;
SQL> ALTER DATABASE FLASHBACK ON;
SQL> ALTER DATABASE OPEN;
```

Für jeden Testlauf wurde nun zunächst die Replikation ausgesetzt, so dass wieder alle Changes gepuffert werden. Danach wurde ein Oracle Restore Point erstellt (dies entspricht einem Copy-on-Write-Snapshot des Datenbankinhalts):

```
SQL> CREATE RESTORE POINT vor_test [GUARANTEE FLASHBACK DATABASE];
```

Nun wurde der Test durchgeführt, sprich eine Testanwendung oder ein entsprechender Applikationsserver auf die neue Datenbank umgebogen. Nach abgeschlossenem Test setzt man die Datenbank auf den Restore Point zurück und startet die Replikation wieder an, wodurch die gepufferten Änderungen nachgefahren werden:

```
SQL> FLASHBACK DATABASE TO RESTORE POINT vor_test;
```

Getestet wurde insbesondere:

- Laufen der Weblogic Server und die anderen Schnittstellen?
- Stichprobenartige Validierungen: Sehen die Daten vernünftig aus?
- Sorgfältige Sichtung der Dokumente (CLOBs, vormals NCLOBs)
- Ist die Performance wie erhofft besser geworden?
- Stimmt die Storage-Konfiguration, sprich wie ist die I/O-Performance?

Der Quantensprung

Nach all diesen Vorbereitungen ist die eigentliche Migration recht simpel: Die Benutzer melden sich freiwillig oder forciert von der Datenbank ab, alle Anwendungen und Schnittstellen werden gestoppt. Die IP-Adressen und Hostnamen des alten und neuen Datenbankservers werden getauscht (dies ist nicht notwendig, vermeidet aber Anpassungen der Connect-Strings).

Bevor die Anwendungen und Benutzer auf die neue Datenbank gelassen wurden, haben wir die Replikationsrichtung gedreht, sprich die alte Datenbank wurde per Shareplex weiterhin mitgepflegt. Diese Aktion kostete nur eine weitere Minute, lieferte aber die vom Kunden gewünschte Möglichkeit eines nachträglichen Fallbacks ohne Datenverlust (siehe Abbildung 3).

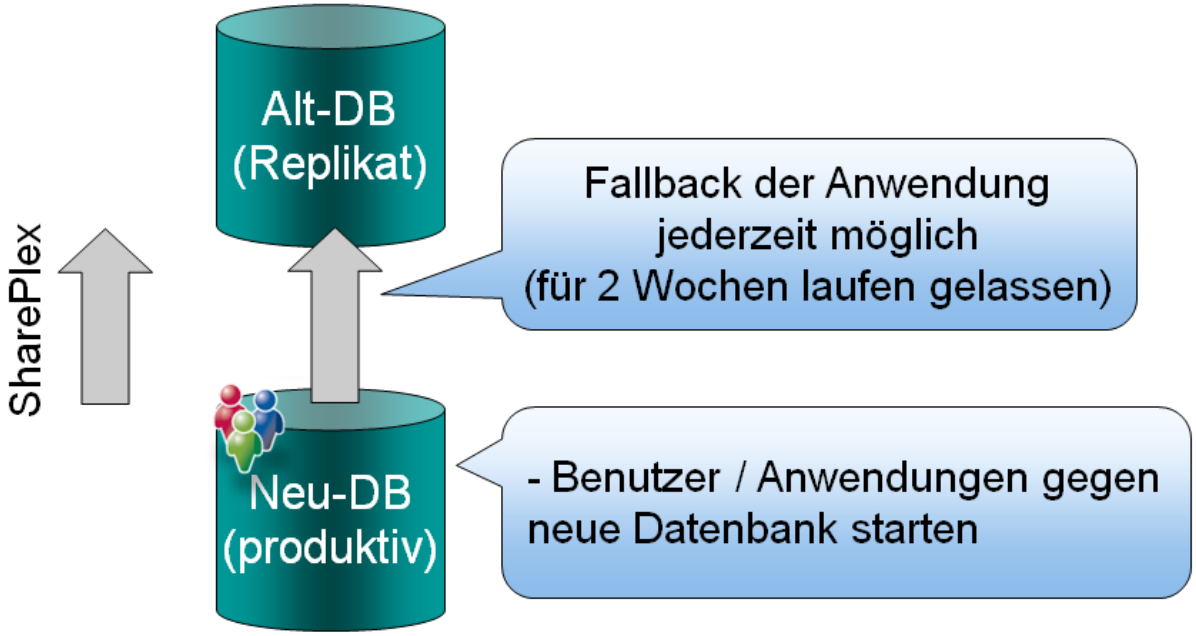

Abbildung 3: Fallback-Replikation nach abgeschlossener Migration

Projektergebnis und Fazit

Der Erfahrung der Autoren nach ist jede Umstellung oder Migration einer geschäftskritischen Datenbank ein Projekt. Die konkreten Rahmenbedingungen bestimmen dabei die jeweils geeignete Migrationsmethode. Bei den verfügbaren Methoden zeigt sich ein Trade-Off zwischen Auszeit und Risiko einerseits sowie Komplexität und Umsetzungsaufwand andererseits.

In diesem Projekt wurde auf minimale Auszeit und Risiko bei einer sehr komplexen Migration Wert gelegt, um den Preis einer erhöhten Komplexität und Gesamtprojektzeit. Die Auszeit aus Service-Sicht war mit ca. 4 Stunden sehr gering (aus Datenbanksicht betrug die Auszeit lediglich wenige Minuten), dafür lagen zwischen Bestandsaufnahme und Projektabschluss etwa 9 Monate.

In dieser Zeit fanden neben Testmigrationen und gescheiterten Versuchen (beispielsweise wegen Konfigurationsproblemen beim Storage der neuen Datenbank) eben auch mehrere funktionale und Performance-Tests sowie Datenvalidierungen in der neuen Umgebung statt. Die Flashback-Database-Funktion hat sich – jedenfalls im Zusammenspiel mit einer Replikation – dafür als hervorragend geeignet erwiesen.

Diese zeigten bereits, dass die ausschlaggebenden Performance-Probleme auf der neuen Datenbank nicht mehr bestanden, was sich nach der tatsächlichen Umschaltung auch bestätigte. Nach 2 Wochen wurde die Fallback-Replikation abgeschaltet.

Kontaktadresse:

Dierk Lenz
Herrmann & Lenz Services GmbH
Höhestr. 37
D-51399 Burscheid

Telefon: +49 (0) 2174-6712 11
E-Mail dierk.lenz@hl-services.de
Internet: www.hl-services.de

Patrick Schwanke
Quest Software GmbH
Im Mediapark 4e
D-50670 Köln

+49 (0) 221-5777 40
patrick.schwanke@quest.com
www.questsoftware.de