Zentralisation & Hochverfügbarkeit im globalen Systembetrieb - Datenbankkonsolidierung und Hot Deployment in der Praxis

Christian Piasecki
PITSS GmbH
Bielefeld

Schlüsselworte
Zentralisation, Hochverfügbarkeit, Datenbank, Forms, Edition-based Redefinition

Einleitung

In diesem Vortrag wird anhand eines konkreten Kunden-Projekts aufgezeigt, welche organisatorischen und technischen Herausforderungen bei einer Zentralisierung von dezentralen Anwendungen auftreten und wie man diese am besten und effektivsten meistert.
Fokus und Zielsetzung des Projekts
Bedingt durch technische Einschränkungen (Hardware, Netzzugang) der letzten Jahrzehnte hat sich in vielen Unternehmen, die über mehrere Standorte verteilt sind, eine dezentrale IT-Landschaft gebildet.
Um Produktion und Tagesgeschäft unabhängig von externen Netzwerken und Netzverfügbarkeit zu ermöglichen hat man auch im Bereich von Oracle Datenbanken und Anwendungen auf dezentrale Lösungen gesetzt: Jeder Standort hat seine eigene Datenbank und eigenen Anwendungen, die unabhängig von den anderen Standorten arbeiten.
Doch eine dezentrale Lösung bringt auch Nachteile mit sich: Es müssen mehrere Systeme überwacht und auf dem aktuellen Stand gehalten werden, was Arbeitsaufwand vervielfacht und Kosten bei Wartung und Betrieb steigen lässt.
Durch die technischen Fortschritte und Weiterentwicklung von Datenbanken und Anwendungssoftware entschließen sich immer mehr Unternehmen, so auch dieser Kunde, dazu, diese Systeme wieder zusammenzuführen und so laufende Kosten verringern.

Rahmenparameter der Anwendung(en) vor der Konsolidierung

- 16 Standorte in unterschiedlichen Zeitzonen
- pro Standort eine Oracle 11gR2 Datenbank EE → 16 Datenbanken
- 16 x 1 (gleiche) Forms-Anwendung

Organisatorische Herausforderungen und Umsetzung

Eine organisatorische Herausforderung ist, dass die 16 Standorte über unterschiedliche Zeitzonen verteilt sind, so dass ein Big Bang Ansatz, bei dem zu einem Zeitpunkt alle Datenbanken zusammengeführt werden, nicht tragfähig wäre, weil es Stillstand an mehreren Standorten bedeuten würde, die zu dem Zeitpunkt eigentlich arbeiten sollten.
Aus diesem Grund wurde die Entscheidung getroffen, die einzelnen Datenbanken im Vorfeld soweit wie möglich für die Zusammenführung vorzubereiten, die tatsächliche Konsolidierung aber nach und nach zu machen. Ausgehen von einer neuen Master-Datenbank werden die einzelnen Standorte nacheinander in diese Datenbank überführt.

Technische Herausforderungen und Umsetzung

Analyse

Wie oben aufgeführt handelt es sich an allen Standorten über die gleiche Forms-Anwendung, die mit den gleichen Datenbank-Schemata in den unterschiedlichen Datenbanken arbeitet. Jetzt könnte man annehmen, dass bei einer zentralen Entwicklung diese Schemata überall gleich sind, doch in der Praxis ist leider nicht immer der Fall. Auch wenn es nur Kleinigkeiten sind, wie z.B.: unterschiedlich angelegte Indexe, könnte das in der zentralen Anwendung große Auswirkungen auf die Performance oder auch die grundsätzliche Funktionalität der Anwendung haben. Aus diesem Grund wurde vor der eigentlichen Umsetzung der Zusammenführung eine Analyse-Phase mit Hilfe des Analyse-Tools PITSS.CON durchgeführt.

Hierbei wurde folgendes untersucht:

Datenbank:

- init.ora
- Tabellen und View Definitionen
- Indexe
- Primary, Foreign und Unique Keys
- Check-Constraints
- Packages, Proceduren und Funktionen
- Hints
- Aufrufe von `dbms_alert`
- Aufrufe von `dbms_pipe`
- `sysdate`
- `Select * from...` –Aufrufe
- `Insert into Tabelle values..` (ohne Spalten-Angabe)

Forms:

- Hints
- Aufrufe von `dbms_alert`
- Aufrufe von `dbms_pipe`
- `sysdate`
- `Select * from...` –Aufrufe
- `Insert into Tabelle values..` (ohne Spalten-Angabe)

Mit Hilfe der Analyseergebnisse wurde folgendes festgestellt und korrigiert:

init.ora:
Hier gab es keine Abweichungen, so dass keine Korrekturen vorgenommen werden mussten.

Tabellen und View Definitionen:
Bis auf eine Ausnahme gab es hier keine Unterschiede in den Datenbanken. Die Abweichung wurde vom Kunden in der betroffenen Datenbank korrigiert.

Indexe:
Unterschiedliche Indexe in den Datenbanken könnten nach einer Zusammenführung der Datenbanken dafür sorgen, dass die Performance nicht mehr gewährleistet ist. Aus diesem Grund musste bei Abweichungen manuell entschieden werden, welche Index-Definition korrekt ist und in die neue Master-DB übernommen wird.

Primary, Foreign und Unique Keys sowie Check-Constraints:
Unterschiedlich definierte Constraints könnten dafür sorgen, dass die Daten nicht korrekt zusammengeführt werden könnten.
Die Herausforderung hier bei war, dass nicht nur auf vorhanden sein untersucht werden musste, sondern die Analyse über die Definition der Elemente gehen musste, da vor allem bei PK und Check-Constraints vom System generierte Namen verwendet werden.
Die einzelnen Abweichungen mussten dann von der IT-Abteilung überprüft und die korrekte Definition in die neue Master-DB übernommen werden.

Packages, Proceduren und Funktionen:
Hier wurden keine gravierenden Unterschiede festgestellt. Einzige Abweichungen waren in Objekten zu finden, die entweder verschlüsselte Werte (Keys) nutzten oder für den Zugriff auf die anderen Datenbanken genutzt wurden.

Hints:
Nutzung von unterschiedlichen Hints oder Hints, deren Indexe unterschiedlich definiert waren könnte zu Performance-Problemen führen.
Es wurde aber nur ein Hint in Forms gefunden, der einen überall gleich definierten Index nutzte, so dass keine Anpassung nötig war.

dbms_alert & dbms_pipe:
Damit diese Funktionen weiter korrekt arbeiten, müssen diese um einen Mandanten erweitert werden, damit mitgeben werden kann, von welchem Standort sie aufgerufen werden. Da die Aufrufe nur gekapselt in der DB stattfinden, sind die Anpassungen gering.

sysdate:
Für das ermitteln von Datumswerten wurde größtenteils eine eigene Funktion geschrieben, welche die korrekte Zeitzone berücksichtigt. Aufrufe von sysdate in der Forms-Anwendung mussten ersetzt werden, weil sonst die Serverzeit genommen wird.

Select * from ... und Insert into Tabelle values...
Da wie später die beschrieben alle Tabellen um eine Mandantenspalte erweitert werden, musste zusätzlich analysiert werden ob select *- und Insert into-Anweisungen ohne Spaltenangabe benutzt werden. Diese würden nämlich Probleme bereiten, sobald es die neue Spalte gibt. Mit Hilfe von PITSS.CON war die Analyse aber kein Problem und die betroffenen Stellen konnten korrigiert werden.

Mandantenfähigkeit

Nach dem die Analysephase und die Korrekturen abgeschlossen waren, musste ein Konzept und die Sourcen für die Zusammenführung in der DB erstellt werden.

Da jeder Systembenutzer genau einem Standort zugeordnet ist, haben wir uns hier für die Einführung einer Mandantenspalte an den Tabellen entschieden. Tabellen die Standort-übergreifende Daten haben,

wurden hiervon ausgenommen, bei allen anderen wird die Mandantenspalte abhängig vom Standort gefühlt.
Doch nicht nur die Tabellen mussten erweitert werden, sondern auch die Constraints und Indexe, damit die Daten weiter eindeutig sind und die Performance unverändert bleibt.
Für die Anpassung dieser Objekte haben wir die Funktion `dbms_metadata.get_dll` benutzt um die vorhandene Definition der Objekte einfach um die neue Spalte zu erweitern.
Damit der Programm-Code nicht verändert werden musste wurden zudem neue Trigger eingeführt, welche abhängig vom User die Mandantenspalte befüllen.
Damit aber weiterhin jeder Benutzer nur die Daten seines Standorts sieht, wurde VPD eigeführt um die Daten zu filtern.

Datenübernahme

Sobald die neue Datenbankstruktur steht, kann es mit der Datenübernahme losgehen. Um den Prozesse möglichst sicher und geringster Ausfallzeit zu bewerkstelligen, wurde er in mehrere Schritte unterteilt.

Damit keine inkonsistenten Daten entstehen können zum Übernahme-Zeitpunkt die Benutzer nicht mit dem Alt- bzw. dem neuen System arbeiten.
Damit die User sich nicht unnötig vom System abmelden, wird vorm Export der Daten nochmal überprüft, ob die Tabellenstruktur (alle benötigten Tabellen und Spalten) auf dem Export-System passt. Sollte diese in Ordnung sein, werden die Daten per DataPump exportiert, falls nicht muss hier nachgearbeitet werden.
Dieser Dump wird dann in ein Staging-Schema der neuen Master-DB importiert.
Da jetzt im Ziel-Schema für die Zusammenführung der Daten alle Indexe, Trigger und Constraints deaktiviert werden, darf jetzt auch hier kein Benutzer angemeldet sein.
Sobald der Import der Daten beendet wurde, werden Trigger und Constraints aktiviert, sowie die Statistiken analysiert und die Indexe neuaufgebaut.
Damit ist der Zusammenführungsprozess abgeschlossen und die Benutzer können wieder arbeiten, in dem sie sich an der zentralen Forms-Anwendung anmelden.

Mehrwert für die Benutzer durch Einführung der Hot Deployment-Fähigkeit

Da solche Projekte nicht auf die IT-Abteilung beschränkt sind, hat man sich beim Kunden auch für die Fachabteilungen, welche mit der Software arbeiten, etwas einfallen lassen, um hier eine breite Unterstützung zu erhalten.
Und zwar bietet man den Usern einen Mehrwert, in dem man die DB durch die “Edition-Based Redefinition”-Funktion Hot-Deployment-fähig macht, so dass zukünftige Updates im laufenden Betrieb eingespielt werden können, ohne dass es eine Downtime des Systems gibt.

Edition-Based Redefinition wurde mit der Datenbank Version 11gR2 eingeführt und ist ein EE-Feature. Es beruht auf dem Prinzip, dass Objekte in der Datenbank in unterschiedlichen Editions vorliegen können, für den User es aber transparent, da er immer nur in einer Edition unterwegs ist.
Neue Funktionen können vom Entwicklungsteam in einer neuen Edition entwickelt werden und nach Fertigstellung den User einfach durch Umschalten der Edition zur Verfügung gestellt werden, ohne dass Objekte invalid werden und recompiliert werden.
Allerdings sind nicht alle Objekte Edition-fähig. So können Tabellen nur in einer Version vorliegen. Aus diesem Grund kommen hier Editioning Views und Crossedition Trigger zum Einsatz. Die Views dienen in unterschiedlichen Edition als Wrapper für die Tabellen, und die Crossedition Trigger sorgen

dafür, dass die Daten konsistent bleiben, obwohl unterschiedliche User zeitgleich in unterschiedlichen Editionen in der Datenbank arbeiten können.

In dem Projekt haben wir uns zu nutzen gemacht, dass die neue Master-DB neu aufgesetzt wurde, so dass wir im Vorhinein, alle Tabellen mit einem Prefix versehen und die Views wie die Tabellen benennen konnten. Das hat den Vorteil, dass der PL/SQL-Code nicht angepasst werden muss. Was noch zu tun war, war das Umhängen der Trigger von den Tabellen an die Views und das Revoken der Rechte an den Tabellen und zuweisen der gleichen Rechte an den Views.

Anmerkung: Als gute Quelle zu diesem Thema diese folgenden Artikel von Tom Kyte aus dem Oracle Magazine:

http://www.oracle.com/technetwork/issue-archive/2010/10-jan/o10asktom-172777.html
http://www.oracle.com/technetwork/issue-archive/2010/10-mar/o20asktom-098897.html
http://www.oracle.com/au/products/database/o30asktom-082672.html

Projektstand und Ausblick

Nachdem die einzelnen Phasen der Vorbereitung abgeschlossen waren, sind die ersten Datenbanken zusammengeführt und die nächsten Übernahmen schon geplant.

Kontaktadresse:
Christian Piasecki
Pitss GmbH
Otto-Brenner-Straße 209
D-33604 Bielefeld

Telefon: +49 521 54679503
E-Mail cpiasecki@pitss.com
Internet: www.pitss.de