Oracle Real Application Clusters für SAP

Peter Sechser, Abocraft

Eine wasserdichte Sicherungsstrategie für unternehmenswichtige Daten ist ein absolutes Muss für jeden Betrieb. Die Notwendigkeit dafür hat sich allerdings noch nicht bei allen herumgesprochen, weshalb es auch immer wieder zu sehr teuren System-Abstürzen mit Datenverlust kommt. Redundanz beim Betrieb von unternehmenskritischen Anwendungen hat auch oft denselben Stellenwert. Für SAP-Systeme kann jedoch seit einiger Zeit Oracle Real Application Clusters (RAC) für eine Ausfallsicherheit im laufenden Betrieb sorgen und sich in eine „Backup & Recovery"-Strategie nahtlos einfügen.

Nachdem SAP Oracle Real Application Clusters seit ein paar Jahren im Stadium „Controlled Availability" Kunden zugänglich gemacht hat, ist RAC seit Anfang 2010 generell verfügbar. Die Kombination von Oracle RAC mit Optionen wie die „Oracle Partitioning Option" boten offensichtlich ausreichend Motivation, um sich nun doch auf dieses Terrain zu wagen, birgt sie doch gegenüber IBMs Ansatz den einen oder anderen Vorteil.

Oracle Dataguard und Oracle RAC im Allgemeinen

Da sich dieser Artikel an die SAP-Community wendet und sowohl Oracle Dataguard als auch Oracle RAC in dieser keineswegs als Standardwissen vorauszusetzen ist, folgt an dieser Stelle nochmal ein kurzer Abriss darüber, wie diese beiden Produkte im Allgemeinen funktionieren und wie sie sich unterscheiden. Diejenigen, die das schon wissen, können diesen Teil einfach überspringen.

Oracle Dataguard und Oracle Real Application Clusters adressieren beide die Problematik der Datenverfügbarkeit. Wie stellt man sicher, dass die Nutzerdaten weiter verwendet werden können, auch dann, wenn plötzlich – aus welchen Gründen auch immer – ein Datenbank-Server ausfällt. Es geht also um die Hochverfügbarkeit des Datenbank-Backends. Oracle Dataguard ist ein Vertreter der „Cold Failover"-Systeme. Die Grundkonzeption entspricht der eines Aktiv-Passiv-Systems. Das bedeutet, man hat in der Regel einen Hauptrechner als Datenbank-Server, der die gesamte Last der Datenbank-Aktivität aus SAP heraus trägt, und einen Zweitrechner, der dann aktiv geschaltet wird, sollte der Hauptrechner sich verabschieden. Für die Dauer der Nutzung des Hauptrechners bleibt der Sekundärrechner in Bezug auf SAP-Nutzer-Aktivität passiv.

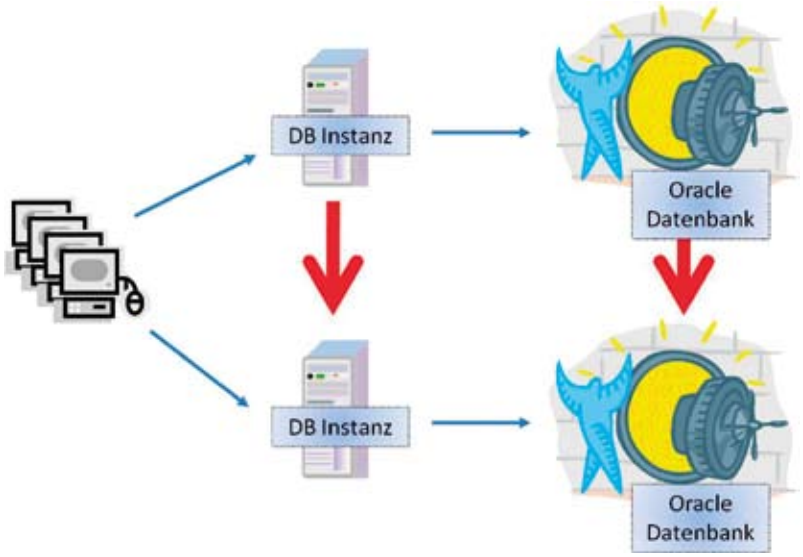

Abbildung 1: Cold-Failover-Architektur mit jeweils eigenen Platten

Synchronisierungsmechanismen zwischen Haupt- und Sekundärrechner stellen sicher, dass alle Daten, die im Rahmen von Transkationen als gespeichert bestätigt wurden, auch auf dem Sekundärrechner vorgehalten werden.

Manche bevorzugen dabei die Methode, Platten und somit die Datenbank ebenso redundant vorzuhalten wie die Rechner selbst. Das hat den Reiz, dass zwischen den Rechnern eine relativ große geografische Distanz liegen kann und daher häufig als Disaster-tolerantes System genutzt wird.

Der nächste logische Schritt besteht nun darin, Haupt- und Sekundär-Rechner nicht als eigenständige Systeme zu betrachten, sondern sie in einen Gesamtverbund zu integrieren und alle Systeme gleichzeitig in vollem Umfang sowohl lesender als auch schreibender Weise zu nutzen. Das ermöglicht Oracle Real Application Clusters (siehe Abbildung 2). Die Limitierung auf nur zwei Rechner entfällt damit ebenfalls.

Bei RAC werden gemäß der Architektur alle verfügbaren Rechner vollständig genutzt. Bei einem SAP-System gibt es jedoch eine leichte Variation, auf die noch später eingegangen wird.

RAC für SAP

Die Architektur von RAC in einem SAP-Umfeld lässt sich relativ einfach beschreiben: In einem 3-Tier-Umfeld liegen die Oracle-Datenbank-Instanzen auf jeweils einem Server des RAC-Clusters, die SAP-Instanzen wiederum befinden sich auf separaten Applikationsservern (siehe Abbildung 3).

In der Realität ist es aber durchaus üblich, mehrere SAP-Instanzen einer bestimmten RAC-Instanz zuzuordnen. Manche SAP-Kunden haben bis

zu zwanzig SAP-Instanzen, die einer einzigen RAC-Datenbank-Instanz zugeordnet sind (siehe Abbildung 4). Natürlich können die SAP-Instanzen auch auf den RAC-Servern installiert werden. Dann entspräche das dem 2-Tier-Ansatz (siehe Abbildung 5).

Im weiteren Verlauf werden wir nicht mehr zwischen 2-Tier- und 3-Tier-Ansatz unterscheiden, denn die Funktionsweise unterscheidet sich konzeptionell nicht. Jedoch gibt es gute Gründe, aus SAP-Sicht den 3-Tier-Ansatz zu verfolgen.

Ausfallszenarien bei 2-Tier- und 3-Tier-Ansätzen

Im Falle eines 2-Tier-Ansatzes fällt bei einem Rechner-Crash nicht nur die Datenbank-Instanz, sondern auch gleich noch die SAP-Instanz mit aus. Genau diesen Fall verhindert die 3-Tier-Architektur: Bei Ausfall eines RAC-Servers kann die SAP-Instanz sauber auf einen der verbleibenden RAC-Server umstellen. Die Nutzer können weiterarbeiten. Da aber häufig die Datenbank-Server viel zu stark konfiguriert sind, werden SAP-Instanzen gerne auf dem RAC-Server installiert. Sollte dann später die Kapazität wiederum nicht ausreichen, werden weitere SAP-Instanzen auf separaten Servern installiert. Dies führt zu einer Mischform von 2-Tier und 3-Tier mit allen Vor- und Nachteilen.

Für den Endanwender kann das bedeuten, dass er im Falle eines 2-Tier-Ansatzes bei Verlust der SAP-Instanz das gleiche Verhalten erfährt, als würde SAP nicht in einem RAC-Umfeld, sondern einfach gegen eine einzelne Datenbank laufen. Er muss sich im schlimmsten Falle neu anmelden. RAC macht hier keinen Unterschied, da es ja nicht die Ausfallsicherheit von SAP-Instanzen als Ziel hat. Zudem werden sich diejenigen SAP-Instanzen anderer Rechner, die mit dem ausgefallenen RAC-Knoten interagiert haben, eine noch verfügbare RAC-Instanz des RAC-Clusters suchen und verbinden.

Im Falle von 3-Tier-Ansätzen ist es jedoch eher unwahrscheinlich, dass die SAP-Instanz eines Applikationsservers und eine RAC-Instanz eines Datenbank-Servers gleichzeitig ausfallen.

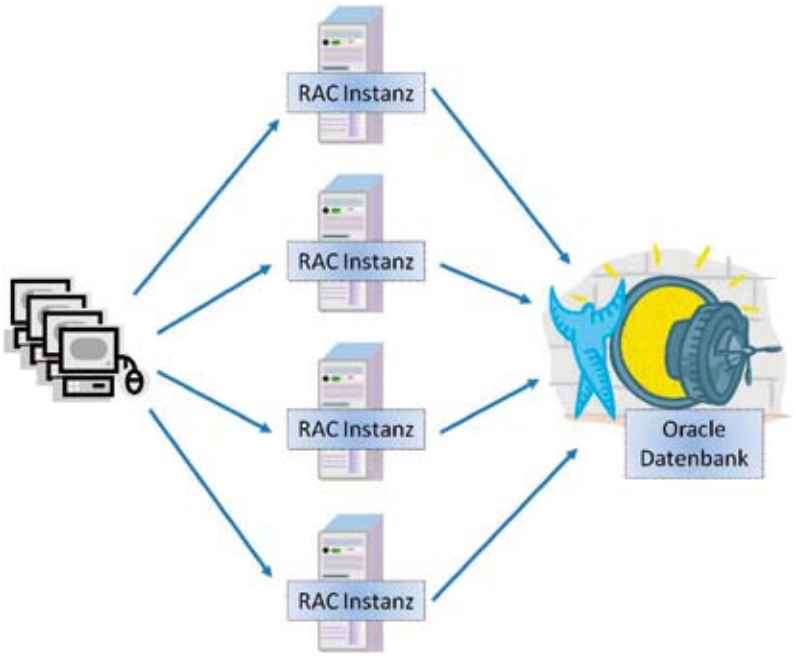

Abbildung 2: Real-Application-Clusters-Architektur

Daher wird der Ausfall eines RAC-Servers zwar bedeuten, dass diese RAC-Instanz nicht mehr nutzbar ist. Aber die betroffenen SAP-Instanzen werden sich jeweils eine noch verfügbare RAC-Instanz suchen und sich mit ihr verbinden. Danach können die Endanwender weiterarbeiten.

Im Falle eines RAC-Server-Crash in einem 3-Tier-Umfeld sind auch nur die Benutzer betroffen, deren SAP-Instanz über den ausgefallenen Knoten Daten bearbeitet hatten. Sie bekommen einen SAP-Short-Dump, die laufende Transaktion wird abgebrochen und der Endanwender muss bei dieser Transaktion von vorne beginnen.

Anwender, deren SAP-Instanz nichts mit dem ausgefallenen RAC-Knoten zu tun hatte, deren SAP-Instanz also mit einer anderen RAC-Instanz kommuniziert hatte, merken von dem Ausfall primär nichts. Ein klarer Vorteil von RAC gegenüber einem einzelnen Datenbankserver.

Erweiterbarkeit eines RAC-Clusters

Neben der Hochverfügbarkeit lässt sich auch eine gute horizontale Skalierbarkeit mit RAC erreichen. Maßzahl ist hierbei der Skalierungsfaktor. Dies bedeutet, dass die Performance-Einbußen bei Erweiterung eines RACs um

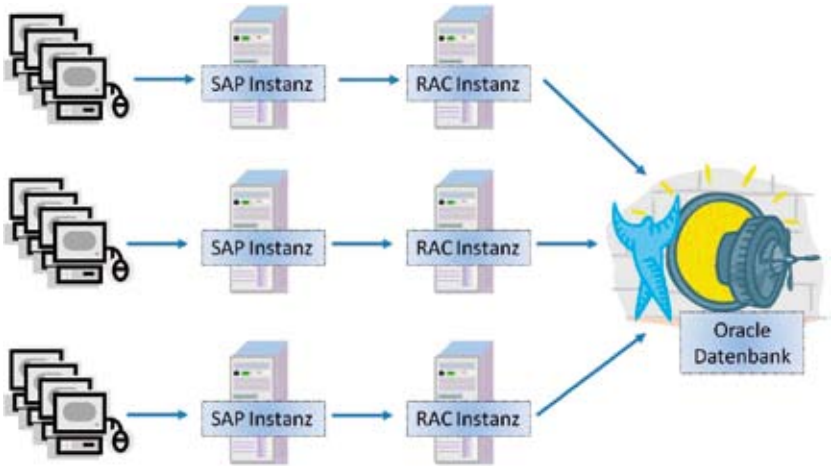

Abbildung 3: 3-Tier-SAP/RAC-Architektur

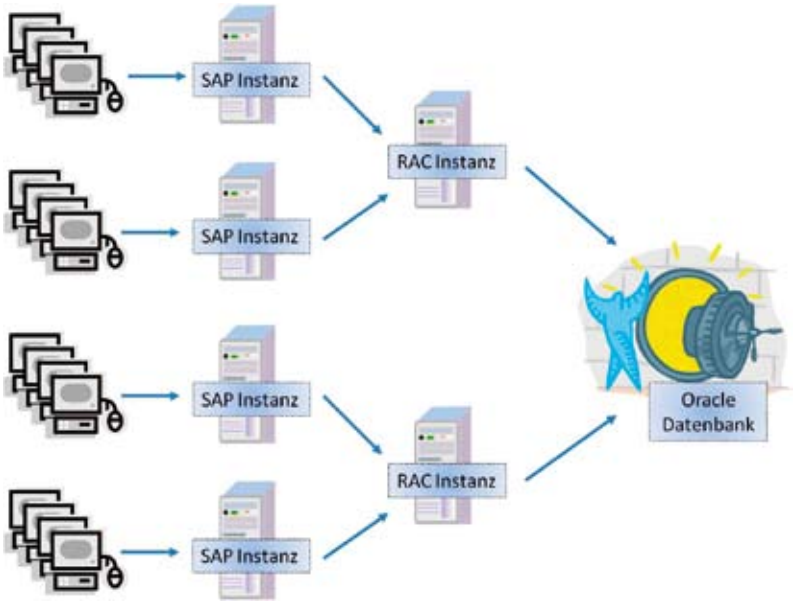

Abbildung 4: Mehrere SAP-Instanzen sind jeweils einer RAC-Instanz zugeordnet

weitere Knoten möglichst niedrig sein sollten. Benchmarks zufolge, die meistens aufgrund ihres hohen Ressourcenverbrauchs mit SD-Usern durchgeführt werden, haben einen Skalierungsfaktor von bis zu über 90 Prozent ergeben.

Zuordnung von Endanwendern zu RAC-Servern

RAC im Allgemeinen heißt, alle Datenbank-Anfragen an beliebige Rechner eines RACs beziehungsweise beliebige RAC-Instanzen eines Clusters verteilen zu können. Wie oben bereits angedeutet, gilt dies in einem SAP-Umfeld nicht uneingeschränkt. Es ist natürlich nicht möglich, einem bestimmten Endanwender einen RAC-Knoten explizit zuzuweisen. Eine bestimmte SAP-Instanz ist zu einem bestimmten Zeitpunkt immer einer entsprechenden RAC-Instanz zugeordnet. Damit wird zum einen eine Aufgabenverteilung auf Ebene von SAP und zum anderen eine Lastverteilung auf bestimmte Knoten eines RAC-Systems vorgenommen. Somit kann die Zuordnung eines RAC-Knotens an einen Endanwender implizit über dessen Tätigkeit vorgenommen werden. Denkbar wäre also beispielsweise eine Konstellation, in der SAP BW RAC-Knoten 1, 2, 3 und 4, Power-Usern Knoten 1 und 2 sowie Batch-Jobs Knoten 5 eines 5-Knoten-RACs zugeordnet sind.

RAC und Dataguard als HA-Lösungen

Mit RAC wird erreicht, dass Batchjobs auch tagsüber laufen können, ohne den operativen Betrieb zu stören, die Aufgaben über alle verfügbaren Hardware-Ressourcen verteilt werden können und Rechnerabstürze nur eingeschränkten Einfluss auf den laufenden Betrieb haben. Während Aktiv-Passiv-Systeme wie Oracle Dataguard nur jeweils einen Teil der Hardware-Ressourcen nutzbar machen können, kann RAC alle Rechnerkapazitäten im vollen Umfang nutzen. Dataguard erlaubt Nur-Lese-Zugriff auf den Sekundärrechner, was zum Reporting ausreichen mag, RAC jedoch erlaubt vollen Lese- und Schreibzugriff auf alle Daten von allen Rechnern aus.

Dataguard kann als Disaster-tolerantes System genutzt werden und braucht hierfür keine besonderen Voraussetzungen außer genügend Netzwerkbandbreite zu erfüllen. Auch RAC kennt die Konzeption eines „stretched Clusters“, also eines RACs über weite geografische Distanzen, benötigt hierfür jedoch unter Umständen spezielle Hardware, die auch nicht unbedingt kostengünstig ist. Aber für SAP-Kunden ist dies ohnehin irrelevant.

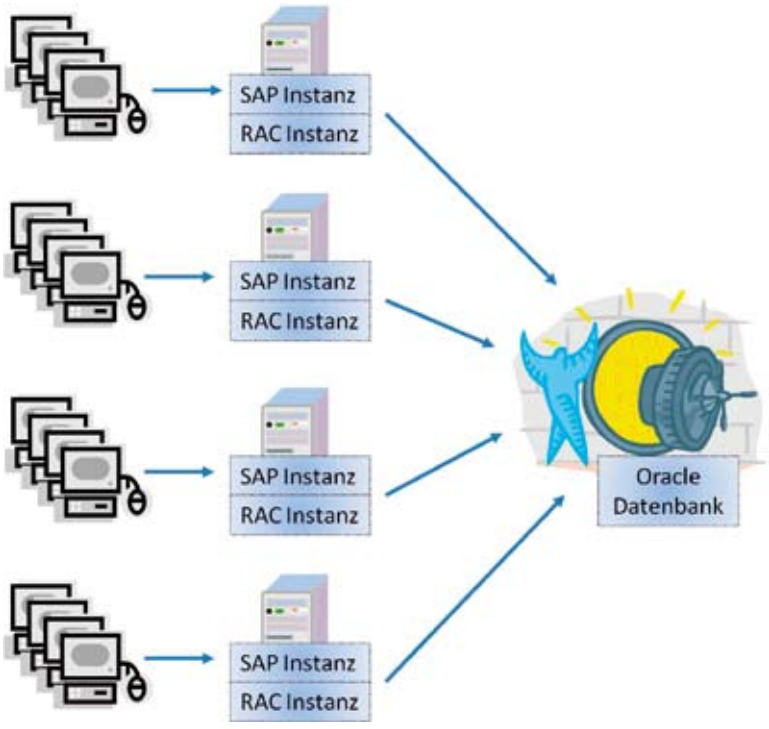

Abbildung 5: 2-Tier-SAP/RAC-Architektur

Fazit

Alles in allem hat Oracle für SAP zwei Hochverfügbarkeitskonzepte mit jeweils unterschiedlicher Zielsetzung. Eine Kombination von beiden ist wohl wieder einmal der goldene Mittelweg.

Kontakt:
Peter Sechser
psechser@abocraft.com