Data-Warehouse-Architekturen mit Exadata weiterentwickeln

Alfred Schlaucher, ORACLE Deutschland B.V. & Co. KG

Es ist schon erstaunlich, dass ausgerechnet bei der Auseinandersetzung mit Wettbewerbern der Blick auf die eigenen Lösungen immer wieder besonders geschärft wird. Kommt SAP mit Hana und ist ekstatisch über etwas mehr Performance, liefern sie ungewollt die Stichworte zu dem, was im klassischen Oracle-Data-Warehouse auf einer Exadata alles machbar ist.

Die meisten Exadata-Data-Warehouse-Einführungen sehen immer noch so aus: Ein Data Warehouse ist zu langsam oder eine Hardware-Ersatzbeschaffung steht an. Dann wird eine Exadata gekauft und das Data Warehouse „1:1" auf die Exadata kopiert. Nach Abschluss des Migrationsprojekts läuft das System um einige Faktoren schneller, und jeder ist zufrieden. Aber es geht noch mehr. Steigt man in die Logik und Architektur des Data Warehouse ein, entdeckt man Möglichkeiten, die den Performance-Schub zusätzlich beflügeln können.

Die Oracle-Empfehlung für eine klassische Data-Warehouse-Architektur ist das Schichtenmodell mit Stage-, Enterprise- und User-View-Layer (Data Marts). Damit lässt sich einerseits ein Data Warehouse fach- und abteilungsübergreifend einsetzen (Enterprise Layer), auf der anderen Seite stellt das Modell über den User-View-Layer fachspezifische Sichten für den Endbenutzer dar (weitere Informationen dazu siehe www.oracledwh.de, 08_konzepte_und_methoden_des_Oracle_dwh und Oracle_dojo4_dwh_konzepte_und_methoden.pdf).

Ungünstige Voraussetzungen ausklammern

Zunächst sollte man einige Konstellationen von der Betrachtung ausschließen, die wenig zusätzlichen Nutzen generieren können: Wer seine Data Marts auf weitere Rechner auslagert, kann nicht über alle Schichten hinweg die extreme Rechenleistung von Exadata nutzen. Das gilt besonders für die zusätzlichen Datenbanken, die – eng gekoppelt an Auswerte-Tools – in einigen Fachabteilungen betrieben werden.

Die zweite Fehlentwicklung entsteht, wenn man das Berechnen, Aggregieren und Kombinieren von Daten in die BI-Werkzeuge verlagert. Fast alle BI-Werkzeuge am Markt folgen diesem Trend. Sie verfügen heute in der Regel über separate Server-Komponenten zum Rechnen und Zwischenspeichern. Solche Server-Komponenten laufen auf separaten Maschinen, die nur einen Bruchteil der Leistung einer Exadata haben. Der Flaschenhals einer Business-Intelligence- und Data-Warehouse-Systemlandschaft wandert damit von der Datenbank auf den Server des BI-Tools. Oft kann man hier wenig ändern, weil sich das Unternehmen strategisch für ein bestimmtes BI-Tool entschieden hat. Man könnte allerdings einen großen Teil der immer wieder gleichen Kennzahlen über Routinen in der schnellen Exadata-Datenbank vorweg berechnen und den BI-Tools diese fertigen Kennzahlen etwa über Views, Materialized Views oder auch über in Table Functions gekapselte PL/SQL-Routinen anbieten.

Dritte Schicht: Ja oder Nein?

Betrachten wir den Fall einer optimalen Data-Warehouse-Architektur, bei der wir alle Warehouse-Schichten auf einer Exadata nebeneinander antreffen. In einem Data Warehouse wird bekanntlich zwischen den Phasen „Laden/Updaten", in der es auf kurze Durchlaufzeiten (Latenzen) ankommt (ETL-Phase), und „Auswerten" mit der Forderung nach guter Abfrage-Performance (Auswerte-Phase) unterschieden. Gute Antwort- und kurze Durchlauf-Zeiten sind jedoch entgegengesetzte Ziele (siehe Abbildung 1):

- Um die Abfrage-Performance zu steigern, wurde bislang sehr viel aggregiert, Kennzahlen vorberechnet etc. Auch die Konstruktion des Starschemas (in Abgrenzung zum Snowflake) war teilweise ein Tribut an die Abfrage-Performance.
- Zur Minimierung der Durchlaufzeiten empfiehlt Oracle seit Jahren, die Zahl der Transformationsschritte zu minimieren.

Bei schnellen Maschinen wie Exadata kann man diese Ambivalenz etwas entschärfen. Es gilt die These „Wenn die Abfrage-Performance durch Exadata ausreichend hoch ist, kann man auf bestimmte vorberechnete Aggregate oder sogar komplett auf Starschemata beziehungsweise Data Marts verzichten und diese durch Views ersetzen. Die ETL-Phase wird noch kürzer, als sie

Abbildung 1: Die klassischen unterschiedlichen Ziele bei Data-Warehouse-Architekturen

mit Exadata bereits ist.“ Auch schon ohne Exadata-Einsatz plädiert Oracle für eine stärkere Durchlässigkeit von Enterprise- und User-View-Layer (Data Mart). Große Bewegungsdaten-Tabellen muss man nicht immer in eine Fakten-Tabelle kopieren, wenn sich an den Inhalten nichts ändert.

Es drängt sich sofort die Frage nach dem Sinn und Zweck des User-View-Layer (Data Marts) auf. Er wird oft als „Performance-Layer“ bezeichnet (was natürlich zu kurz greift …). Die Techniken zur Performance-Optimierung im Data Mart sind:

- Vorberechnete Aggregate stellen Kennzahlen bereit
- Dimensionen mit mehreren Hierarchie-Leveln stellt man als denormalisierte Tabellen dar und minimiert so die Anzahl der Joins
- In einem Oracle-Datenbank-basierten Starschema verhindert die Star-Transformation einen Full-Table-Scan auf die Fakten-Tabelle
- In der Oracle-Datenbank können komplette multidimensionale Datenräume als OLAP-Cube vorberechnet sein

Alle vier Verfahren benötigen persistente Datenbank-Objekte, die im Verlauf des ETL-Prozesses entstehen, der sich dadurch verlängert (größere Latenzen). Alle vier Techniken sind auch alternativ beispielsweise mit Views lösbar. Bisher scheute man vor solchen Gedankenspielen zurück, vielleicht weil das Drei-Schichten-Modell so selbstverständlich vorkam oder weil es noch keinen Anlass gab, darüber nachzudenken (siehe Abbildungen 3).

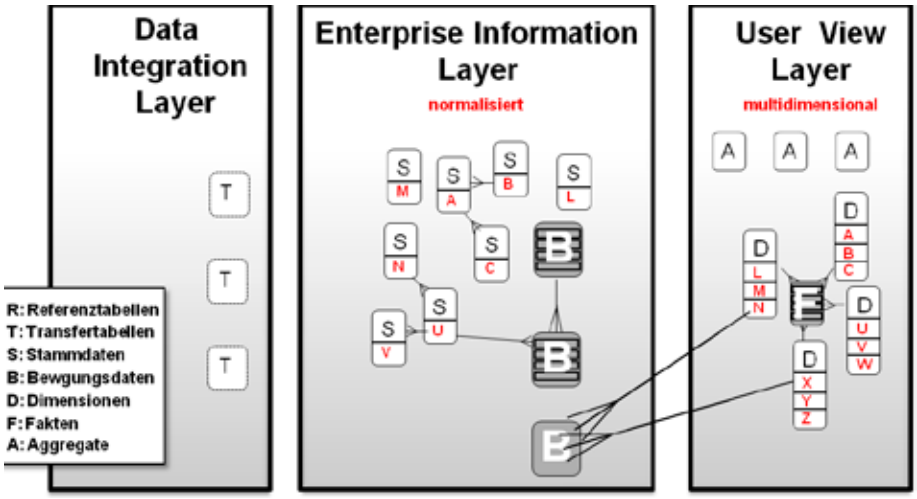

Abbildung 2: Klassisches Drei-Schichten-Modell

Die Gegenargumente

Es wird sofort eine Reihe von Einwänden gegen diese Betrachtung geben. Man hört: „Wir trennen Data Marts von der Kern-Schicht, weil keine Endbenutzer ins Warehouse sollen. Gründe sind Performance und Security.“ Diese Pauschalisierung ist zu ungenau: Denn zum einen ist Exadata gerade dafür gemacht, eine extreme Performance und viele gleichzeitige Benutzer-Prozesse zu unterstützen, und zum anderen löst man Security-Anforderungen nicht über das Schichten-Modell, sondern mit dedizierten Zugriffsschutzmechanismen der Datenbank wie Mandantenfähigkeit, Data Masking oder spalten- und zeilenbezogenen Zugriffsschutz.

Dann kommt die Aussage: „Ein Data Mart wird gebraucht, weil Informationen hier anders organisiert sind als im Kern-Data-Warehouse, und zwar orientiert an den Erwartungen der Endbenutzer. Ein Hilfsmittel dafür ist das multidimensionale Modell, bei dem die denormalisierten Dimensionen die realen Geschäftsobjekte der Fachanwender widerspiegeln.“ Dies ist ein gutes Argument, das oft vergessen wird, wenn es nur um Performance geht. Aber den Informationsgehalt einer Dimension kann man auch über eine View ausdrücken. Dazu ist keine persistente Tabelle notwendig. Hinzu kommt, dass in vielen BI-Werkzeugen die multidimensionale Sicht sowieso wieder nachgebildet wird, sodass man fragen muss: „Braucht man eine multidimensionale Struktur in der Data-Warehouse-Datenbank?“

Es gibt noch dieses Argument: „Dimensionen werden nicht nur in einem Data Mart, sondern über dessen Grenzen hinweg benutzt, weil sie sachgebietsübergreifende Abfragen ermöglichen sollen (conformed dimensions). Das ist zwar prinzipiell auch mit Views möglich. Views werden jedoch leicht unübersichtlich. Außerdem kann man die Konsistenz der Daten schwieriger sicherstellen, weil sich die Tabellen-Inhalte hinter den Views unkontrolliert ändern können.“

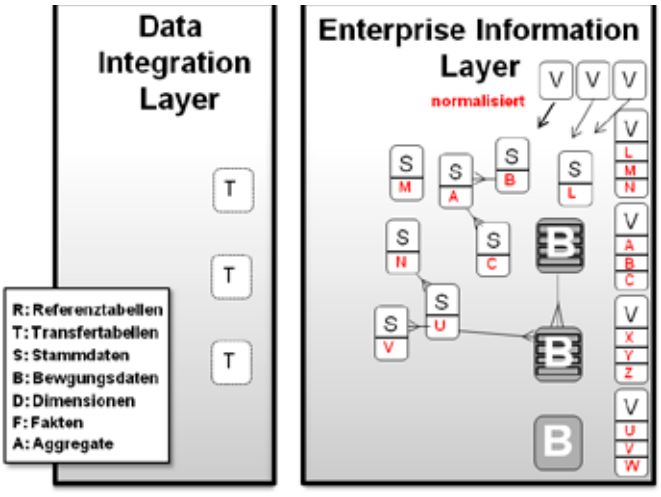

Abbildung 3: Reduziertes Schichtenmodell mit der Ersetzung der Dimensionen und Aggregate durch Views

Dieses Argument ist sicher akzeptabel. Der Verwaltungsaufwand bei einem View-Konzept steigt. Bildet man über den ETL-Prozess echte Dimensions-Tabellen, so hat man

zu einem definierten Zeitpunkt (Ladelaufzeit des ETL-Prozesses) einen fixen Stand. Der drohenden Komplexität kann man aber über eine saubere Metadaten-Verwaltung gegensteuern.

Zu guter Letzt bleibt noch die Historisierung der Stammdaten: „In einer persistenten Dimensions-Tabelle lässt sich die „Slowly Changing Dimension"-Anforderung leichter umsetzen als über das View-Konzept." Das Argument lässt sich allerdings schnell ins Gegenteil umdrehen, denn diese Art der Stammdaten-Historisierung ist in den Tabellen der Kernschicht (Enterprise-Layer) leichter umsetzbar, zumal Data Marts heute oft flüchtig angelegt werden und sich ihre Struktur häufiger ändert, also alle Daten dann komplett ersetzt werden und in dem Data Mart keine Historien-Verfolgung stattfindet.

Was soll man jetzt machen?

Im Prinzip findet man an allen genannten Aspekten etwas Gültiges. Man sollte also nicht gleich seine Star-Schemata (Data Marts) abschaffen, sondern folgende Optionen betrachten:

- Prüfen, ob aufgrund der gegebenen Abfrage-Performance die eine oder andere Aggregation notwendig ist oder zugunsten einer geringeren ETL-Latenz abgeschafft werden kann.
- In seinen ETL-Prozessen nach „1:1"-Kopiervorgängen suchen. Dies ist meist ein Indiz für unnötige ETL-Läufe und Plattenplatz-Verschwendung.
- Prüfen, ob Abfragen auf ein Star-Schema auch dann noch schnell laufen, wenn die Dimensionen über Views abgebildet werden und bei jedem Zugriff im Hintergrund ein Sub-Select über die View erfolgt. Die Star-Transformation bleibt davon unberührt, auch wenn Views anstatt Dimensions-Tabellen genutzt werden.
- Auch die Rewrite-Fähigkeit und die Performance von Materialized Views prüfen, wenn diese sich über Joins zwischen einer Fakten-Tabelle und Views versorgen.
- Wenn keine Nachteile für die Abfrage-Performance beziehungsweise das Materialized-View-Verhalten auftauchen, sollte man die potenziellen Zeitersparnisse in der ETL-Phase prüfen, wenn man auf Aggregate beziehungsweise Star-Schemata verzichtet. Ist die Ersparnis deutlich, hat man ein Argument für ein Redesign; wenn nicht, und falls es auch sonst keine Vorteile bietet, dann kann man das bestehende Verfahren beibehalten.

Wie alles bei einer Architektur-Betrachtung ist diese Darstellung sehr pointiert und vereinfacht. Sie soll aber zum Nachdenken bewegen. Die Rechner-Systeme haben in den letzten Jahren wieder einen gewaltigen Sprung hin zu mehr Performance gemacht. Unsere Vorstellung darüber, wie schnell diese Maschinen sind, hinkt in den meisten Fällen hinterher. Erst recht das, was wir daraus machen.

Fazit

Manches, was ursprünglich nur zur Performance-Optimierung erfunden wurde, hat sich als State-of-the-Art in den Köpfen etabliert. Das immer wieder „Infragestellen" ergibt Sinn. Die Diskussion um Exadata ist in den letzten Jahren stark von der Infrastruktur-Seite beziehungsweise oft nur über die Hardware geführt worden. Den Aspekt „Mehr Performance" in die Architektur-Planung mit einzubeziehen, kann noch mehr Potenziale bei der Exadata öffnen: Schnellere Rechner erfordern auch ein Umdenken bei den Architekturen.

Alfred Schlaucher
alfred.schlaucher@oracle.com