Können "fast changing monster dimensions" gezähmt werden?

Dr. Andrea Kennel
InfoPunkt Kennel GmbH
CH-8600 Dübendorf

Schlüsselworte:

Data Warehousing, Dimensionen, Performance, Slowly Changing Dimensions.

Einleitung

Unsere Kundin ist im Inkassobereich tätig. Dabei geht es um einzelne Fälle, die besagen, wer wem welchen Betrag schuldet. Es geht also um offene Rechnungen. Ein Fall kann längere Zeit im System gepflegt werden, bis er abgeschlossen werden kann. Je Fall werden dazu verschiedene Daten benötigt. So Daten des Schuldners, Daten des Gläubigers, offene Beträge, Zahlungen, Status des Falles, Zinsen und vieles mehr.
Damit verschiedene Auswertungen, die direkt ab dem produktiven System bezogen wurden, schneller werden und verlässlicher sind, wurde beschlossen ein Data Warehouse aufzubauen. Bei der Modellierung stellten wir fest, dass viele Daten direkt mit dem Fall verknüpft sind und die Dimension Fall mehrere Hierarchien ausweist. So haben wir in einem ersten Schritt diese Dimension CASE mit allen Hierarchien in einer einzigen Tabelle gespeichert. Bereits im Piloten mussten wir feststellen, dass diese Dimension CASE etwa viermal so gross wurde wie die grösste Fakten Tabelle.
So beschlossen wir beim Release1 einen Umbau dieser Dimension vorzunehmen. Dabei teilten wir diese eine Dimension in mehrere kleinere Dimensionen auf, so dass jede Dimension nur noch eine Hierarchie enthält.
Der vorliegende Artikel beschreibt, wo genau die Probleme einer grossen „Monsterdimension“ liegen und stellt die beiden Lösungsansätze „Monsterdimension“ und „Horde von Drachen“ (mehrere kleinere Dimensionen) einander gegenüber.

Slowly Changing Dimensions, eine Kurzeinführung

Im Data Warehousing spricht man von Fakten und Dimensionen. Die Fakten sind die Kennzahlen, die ausgewertet werden. Die Kriterien, nach denen die Fakten selektiert und ausgewertet werden, sind in den Dimensionen gespeichert.
Ein Beispiel: wir möchten wissen, wie viel gestern auf Fällen bezahlt wurde, die im Status Mahnphase sind und dem Kunden XY gehören. Das Faktum ist der Betrag der einzelnen Zahlung. Diese Beträge selektieren wir über das Datum und über die Dimension Fall (CASE), wobei wir nur die Fälle selektieren, die gestern im Status Mahnphase waren und dem Kunden XY gehören.
Somit haben wir in der Dimension CASE zwei Hierarchien. Einerseits haben wir den Status, über den wir Fälle gruppieren und selektieren können, anderseits haben wir den Kunden.
In diesem Beispiel gehört ein Fall immer einem einzigen Kunden. Sollte, aus welchem Grund auch immer, der Kunde geändert werden, so interessiert uns nur der aktuelle Kunde. Das heisst, wir können im Data Warehouse den Kunden überschreiben, falls dieser sich ändert. Diese Art von Änderungen heisst nach Definition Kimball [KIM01] Slowly Changing Dimension Typ I (SCD I). Das bedeutet, dass sich die Dimension in dieser Hierarchie zwar ändern kann, wir aber immer nur den aktuellen Wert brauchen.

Anders sieht es bei der Hierarchie Status aus. Hier wollen wir Änderungen speichern, respektive wir wollen zu jedem Zeitpunkt wissen, in welchem Status ein Fall war. Das bedeutet, dass Änderungen im Data Warehouse als Versionen gespeichert sein müssen. In dem Fall sprechen wir von Slowly Changing Dimension Type II (SCD II).

Nach diesem kurzen Exkurs über Slowly Changing Dimensions kommen wir endlich zum versprochenen Beispiel unserer Monsterdimension und zu der Frage, ob diese gezähmt werden kann.

Die Dimension CASE als Monsterdimension

Ein Fall (Case) kann verschiedene Stati durchleben. So gibt es beispielsweise eine Eröffnungsphase, eine Mahnphase und einen Fallabschluss. Im Data Warehouse interessiert die gesamte Geschichte, also die ganze Historie. So haben wir hier einen typischen Fall für eine SCD II.
Konkret heisst das, dass wir in der Dimension CASE für jeden unterschiedlichen Status eine Version mit einem gültig_von und gültig_bis speichern.
Weiter gehört ein Fall zu einem Produkt. Das Produkt hängt nicht mit dem Status zusammen und kann sich im Verlaufe der Zeit mehrfach ändern. Auch hier interessieren alle Produkte eines Falls (Case). Auch diese Hierarchie ist eine SCD II.
Betrachten wir mal bis hier hin das Datenmodell und ein paar Beispieldatensätze:

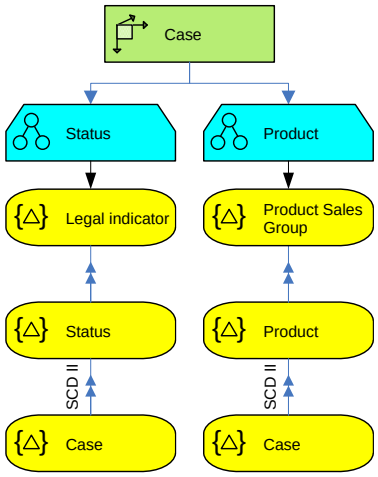

Abb. 1: Zwei Hierarchien in einer Dimension

case_id	case_nr	status	product	valid_from	valid_to
1	100	1	1	14.03.2009	21.03.2009
2	100	1	2	22.03.2009	25.03.2009
3	100	2	2	26.03.2009	15.04.2009
4	100	3	2	16.04.2009	

Abb. 2: Mögliche Datensätze der Dimension

Somit haben wir zwei SCD II in einer Dimension. Ändern sich diese Hierarchien nicht gleichzeitig, so erhalten wir relativ viele Versionen und damit viele Datensätze zu einem einzelnen Fall.

Bei unserer Kundin haben wir für die Dimension CASE sogar 8 verschiedene Hierarchien, wobei 6 davon vom Typ SCD II sind.

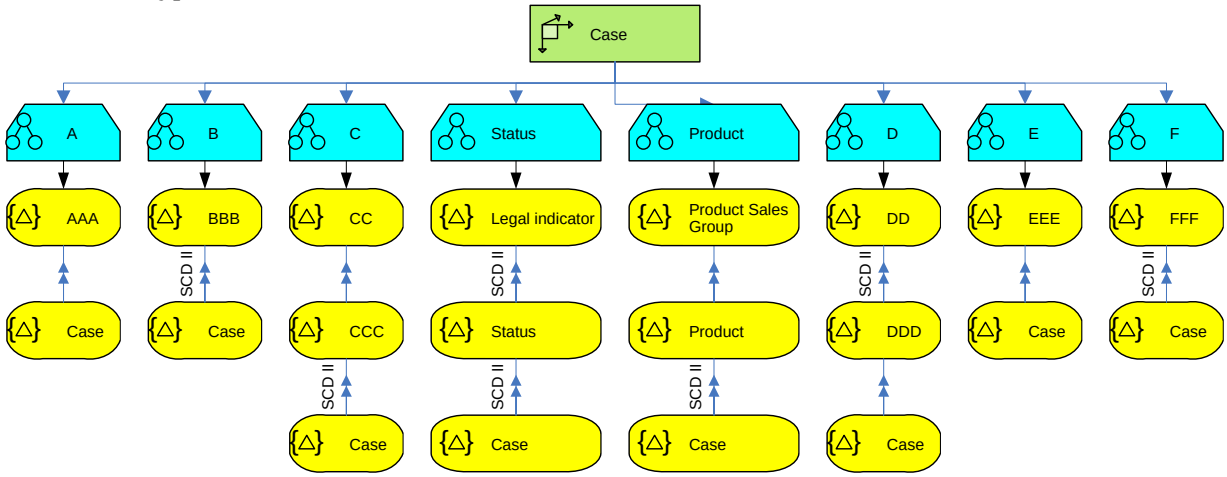

Abb. 3: Die Monsterdimension

Folgende Abfrage zeigt, wie viele Einträge je Case in der Dimension CASE abgelegt sind:

```
select case_nr, count(*)
from d_cases
group by case_nr
order by 2 desc
```

Bei unserer Kundin gibt es Cases, die bis zu 121 Versionen enthalten.

Doch überlegen wir uns, wozu ein Data Warehouse genutzt wird. Es geht darum, dass Daten einmal geschrieben und dann viele Male gelesen werden. Ein Data Warehouse soll also bewusst Redundanzen enthalten, wenn diese für das Lesen von Vorteil sind.
Beispielsweise wollen wir wissen, wie viel auf Fälle im Monat April bezahlt wurde, die am ersten April Status 2 und Produkt 2 hatten. Mit unserer Monsterdimension finden wir sehr schnell, dass hier auch Zahlungen der case_nr 100 dazu gehören.

Eine Horde kleiner Drachen

Die Monsterdimension habe aber auch klar Nachteile. Will ich nur auf eine Dimension einschränken, so finde ich normalerweise mehrere Datensätze in der Dimension über die dann die Faktentabelle gelesen wird. Erschwerend kommt hinzu, dass die Dimensionstabelle gross wird und damit schon eine Abfrage nur auf der Monstertabelle recht langsam sein kann.
In unserem Beispiel ist die Monstertabelle deutlich grösser als die Faktentabelle. Das hat wohl auch damit zu tun, dass die Faktentabelle relativ klein ist, da nicht auf jedem Case jeden Tag Buchungen statt finden und als Fakten vor allem Buchungen interessieren. Da die Monsterdimension als einzige Dimension grösser ist als die Faktentabellen und sogar 4 mal grösser als die grösste Faktentabelle beschlossen wir, eine andere Lösung zu suchen.
Weiter war die Implementation der Ladeprozeduren für die Monsterdimension nicht gerade trivial und damit auch nicht sehr performant und fehleranfällig.

Wir beschlossen das Monster zu zähmen, indem wir dieses in mehrere kleinere Dimensionen aufteilten. So haben wir nicht ein grosses Monster, sondern kämpfen mit einer Horde kleiner Drachen.

Hier das neue Datenmodell mit den kleinen Dimensionen (Drachen) und ein Datenbeispiel

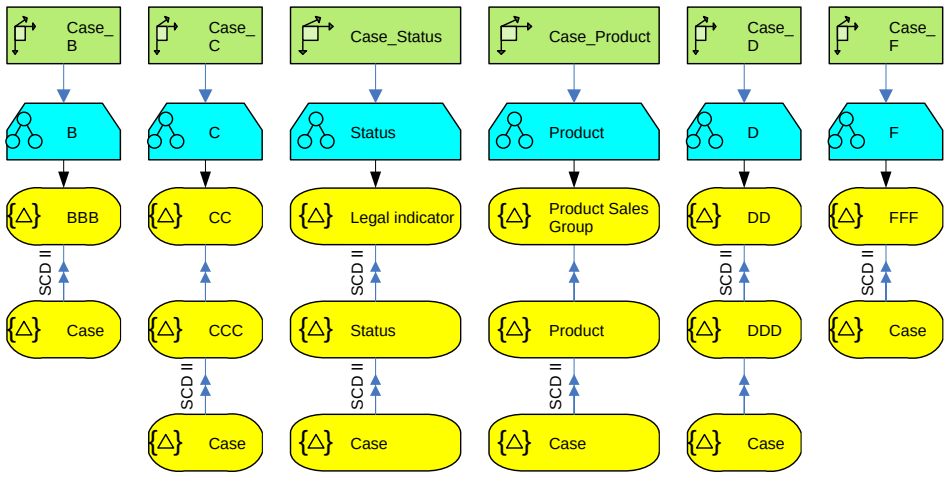

Abb. 4: Mehrere kleinere Dimensionen

id	case_nr	status	valid_from	valid_to
11	100	1	14.03.2009	25.03.2009
12	100	2	26.03.2009	15.04.2009
13	100	3	16.04.2009	

id	case_nr	product	valid_from	valid_to
21	100	1	14.03.2009	21.03.2009
22	100	2	22.03.2009	

Abb. 5: Mögliche Datensätze der Dimension

So haben wir für jede Hierarchie, die vom Typ SCD II ist, eine eigene Dimension definiert. Im Gespräch mit der Kundin stellten wir fest, dass bei vielen Abfragen die Geschichte eigentlich nicht interessiert, sondern nun die heute aktuelle Hierarchie. So beschlossen wir, eine weitere Dimension CASES anzulegen, die alle Hierarchien als SCD I enthält, also diejenigen, für die wir eine eigene Dimension mit SCD II definiert haben, und die Dimensionen, die nur als SCD I vorkommen.

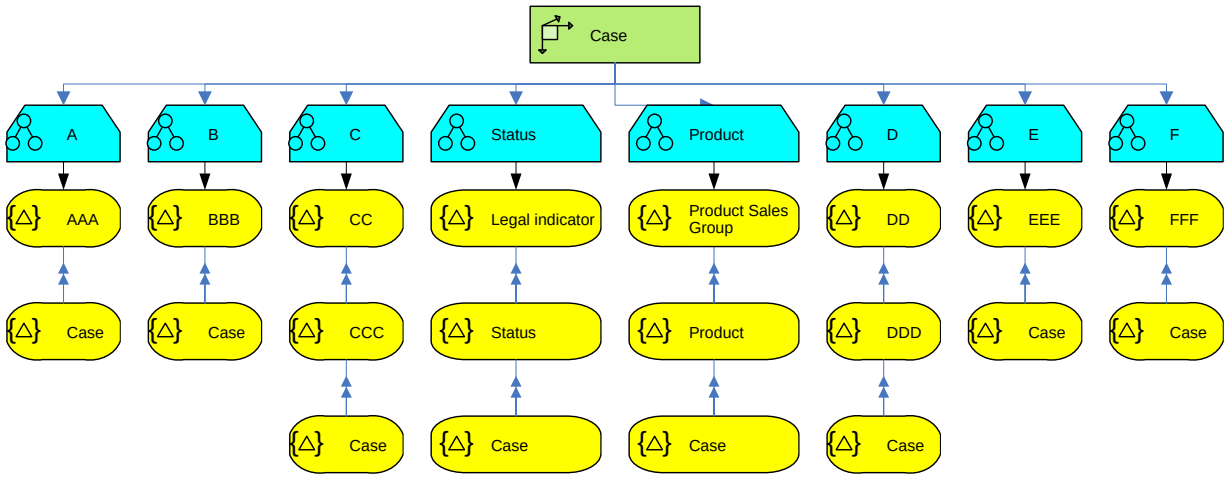

Abb. 6: Eine einfach Dimension mit allen Hierarchien als SCD I

Bei diesem Datenmodell sind nun klar die Dimensionen kleiner. Dafür werden aber die Faktentabellen, die auf CASES verweisen etwas grösser. In den Fakten gibt es neu nicht mehr nur einen Fremdschlüssel für die Dimension CASES sondern gleich deren 7, 6 Fremdschlüssel für die 6 Dimensionen mit SCD II und einen für die Dimension mit SCD I.

Vergleich der Implementation

Bei der Monsterdimension muss für einen Fall immer dann eine neue Version in CASES geschrieben werden, wenn sich eine der 6 SCD II geändert hat. Dabei muss der Ladeprozess so umgesetzt werden, dass auch mehrere Tage gleichzeitig geladen werden können. So kann es sein, dass gleich mehrere Versionen für ein Case neu geladen werden müssen, weil sich seit dem letzten Ladeprozess mehrere Hierarchien an unterschiedlichen Tagen verändert haben.
Nun ist eine solche Versionierung über mehrere veränderbare Hierarchien nicht sehr einfach und in einem Schritt fast nicht lösbar. Daher haben wir je Hierarchie eine Tabelle in der Staging-Area angelegt. In diesen Tabellen werden die neu zu ladenden Daten je Hierarchie gespeichert. Basierend auf diesen Daten werden dann die neuen Versionen erstellt und schlussendlich in die eigentliche Dimension geladen.
Der Ladeprozess für die kleinen Dimensionen ist hier klar einfacher. Wie bei der Monsterdimension werden die neu zu ladenden Daten je Dimension in der Staging-Area gesammelt. Nun können diese relativ einfach als neue Versionen in die kleinen Dimensionen geladen werden. Zum Laden der Dimension mit den SCD I Daten kann nun einfach bei allen Dimensionen der letzte gültige Wert gelesen werden und die Dimension SCD I geschrieben werden.
So ist der Ladeprozess sowohl für den ersten Ladevorgang mit allen Daten (Full Load) als auch für den regelmässigen Ladevorgang (Delta Load) mit dem neuen Datenmodell mit den kleinen Dimensionen einiges einfacher und damit übersichtlicher und weniger fehleranfällig.
Dies zeigt sich aber auch in der Performance.

Ladevorgang	**Laufzeit Monsterdimension**	**Laufzeit kleine Dimensionen**	**Differenz**	**%**
Full Load (gesamter Load)	17:49:00	10:54:00	06:55:00	39%
Delta Load (nur CASES)	00:30:10	00:19:08	00:11:02	36%

Abb. 7: Vergleich Ladezeiten

Wir haben gesehen, dass die Implementation der Monsterdimension klar komplexer ist. So erstaunt es auch nicht, dass der Ladevorgang mit den kleinen Dimensionen deutlich schneller ist als das Laden der Monsterdimension.

Vergleich der Abfragen

Nachdem wir nun gesehen haben, wo die Unterschiede beim Implementieren und Laden sind möchten wir den wohl kritischsten Punkt betrachten. Welche Abfragen sind bei welcher Lösung performanter?
Dazu nehmen wir drei Abfragen als Beispiel. Dabei haben wir Abfragen gewählt, die bei der Lösung mit kleinen Dimensionen unterschiedlich viele Dimensionen lesen muss.
Hier die drei Aufgabenstellungen für Abfragen im DWH:

1. Summe der Zahlungen für ein Produkt für Monate Juni und Juli
2. Summe der Zahlungen für Produkt, Kunde, Status für Monate Juni und Juli
3. Summe der Zahlungen mit Einschränkung auf allen 6 SCD II Dimensionen

Bei der ersten Abfrage muss für die neue Lösung nur eine kleine Dimension mit den Fakten verknüpft werden. So können wir davon ausgehen, dass hier die Lösung mit den kleinen Dimensionen schneller ist als die Monsterdimension. Fraglich ist, wie sich die beiden Lösungen bei den Abfragen über drei oder gar 6 Dimensionen bewähren.

Messungen der Abfragen haben gezeigt, dass bei der Monsterdimension die Laufzeiten für alle drei Abfragen ähnlich lange dauern. Dies ist nachvollziehbar, da Oracle zuerst die Einschränkungen auf der Monsterdimension durchführt und dann auf die Fakten zugreift.
Bei den kleinen Dimensionen wird bei der ersten Abfrage neben der Faktentabelle nur eine Dimensionstabelle benutzt. So ist diese tatsächlich deutlich schneller als die Monsterdimension. Wie zu erwarten haben die Messungen gezeigt, dass sobald mehrere Dimensionen gebraucht werden, die Abfragen langsamer werden. Doch auch die Abfrage, die sechs Dimensionen verwendet ist schneller als die Abfrage über die Monsterdimension. Diese Abfragen könnten aber klar noch beschleunigt werden, wenn mit Star-Optimierung gearbeitet wird, was hier noch nicht genutzt wurde.

Welche Lösung haben wir heute?

Im aktuellen Release des Data Warehouses haben wir die Lösung mit den kleinen Dimensionen umgesetzt. So sind die Ladeprozesse übersichtlicher, einfacher und auch klar schneller. Damit sind Erweiterungen, die sicher kommen werden, auch einfacher und schneller umgesetzt.
Doch ein Problem haben wir mit unserer Horde kleiner Drachen. Das Werkzeug, das für die Auswertungen eingesetzt wird, kann besser mit einer Monsterdimension umgehen als mit einer Horde Drachen. So haben wir zu unseren kleinen Drachen eben noch ein Monster gestellt. Konkret heisst das, dass wir zuerst immer die kleinen Dimensionen füllen. Für die Monsterdimension haben wir eine Monsterview definiert, die aus den kleinen Dimensionen eben die Monsterdimension zusammenstellt. Damit die Daten aber schnell gelesen werden können, schreiben wir nach dem Laden der kleinen Dimensionen die Daten der Monsterview in die Monstertabelle.

Fazit

Mit unserer Lösung der kleinen Dimensionen und zusätzlicher Bereitstellung der Monsterdimension sind wir sehr flexibel. Der Ladeprozess ist einfach und überschaubar und Abfragen können wahlweise über die kleinen Dimensionen oder über die Monsterdimension erfolgen.
Diese Mischlösung hat natürlich auch Nachteile. Dadurch dass wir die Daten der Dimension CASES sowohl in den kleinen Dimensionen als auch in der Monsterdimension speichern, brauchen wir mehr Speicherplatz. Dies ist aber ein lösbares Problem.

Wenn wir in einer nächsten Erweiterung des Data Warehouses wieder eine Dimension entwerfen, die mehrere Hierarchien mit SCD II hat, werden wir diese von Anfang an in mehrere Dimensionen aufteilen. Falls es aus irgendwelchen Gründen dann doch nötig wird eine Monsterdimension zu erstellen, so kann dies nachträglich ergänzt werden. Die Umkehrung, aus der Monsterdimension mehrere kleinere Dimensionen zu erstellen, ist weniger einfach, da es auch Anpassungen in den Fakten mit sich zieht.

So haben wir dank der Horde von kleinen Drachen die Monsterdimension gebändigt.
Gerne bedanken wir uns bei unserer Kundin, die uns ermöglicht hat, die vorliegende Studie durchzuführen. Es ist wichtig, dass eine Firma den Mut hat, neue Konzepte zu erarbeiten und konsequent umzusetzen. Meine Auftraggeberin ist hier als mittelgrosses Unternehmen wohl führend. Werden doch in verschiedenen Projekten konsequent neue Konzepte eingesetzt, dies mit Erfolg.

Literatur:

[KIM01] The Data Warehouse Toolkit. Ralph Kimball. Wiley 2002. ISBN 0-471-20024-7

Kontaktadresse:

Dr. Andrea Kennel
InfoPunkt Kennel GmbH
Bahnhofstr. 48
CH-8600 Dübendorf

Telefon: +41 (0) 44 820 71 40
E-Mail andrea@infokennel.ch
Internet: www.infokennel.ch