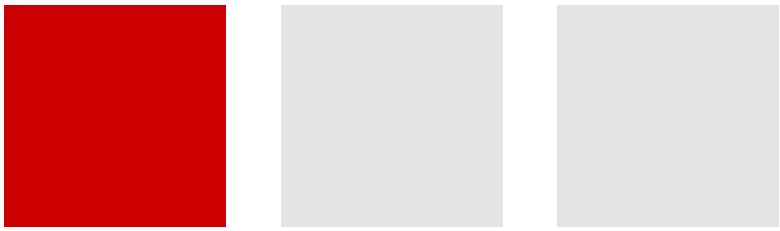


trivadis
makes IT easier.

Welche Daten gehören ins Data Warehouse?

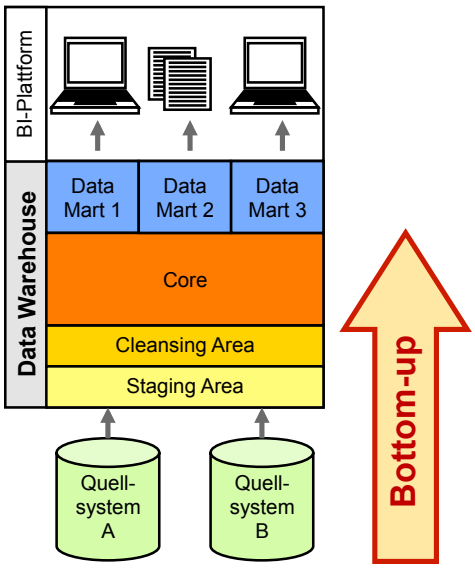
Dani Schnider
Principal Consultant
9. Januar 2012

In vielen DWH-Projekten stellt sich die Frage, welche Daten im Data Warehouse gespeichert werden sollen und wie dieser Datenumfang festgelegt werden soll. Es gibt unterschiedliche Vorgehensweisen, die in diesem Artikel beschrieben und verglichen werden. Die Wahl der Vorgehensweise hat einen wesentlichen Einfluss auf das Datenmodell des Data Warehouses.

Ein Data Warehouse (DWH) ist die Datenbasis für Business Intelligence (BI) und stellt Informationen für verschiedene Auswertungen und BI-Applikationen zur Verfügung. Die Daten können aus einem oder mehreren Quellsystemen stammen. Typischerweise werden alle Daten in einem zentralen Core gespeichert, das für die Integration und Historisierung der Quelldaten verwendet wird. Aus dem Core werden dann die Data Marts für die unterschiedlichen BI-Anwendungen beliefert.

Vor allem bei Data Warehouses, die von Grund auf neu erstellt werden („auf der grünen Wiese", wie es oft so schön heisst), ist während der Planungs- und Konzeptphase oft noch ziemlich unklar, wie die Datenmodelle von Core und Data Marts aussehen werden und welche Datenbestände darin gespeichert werden sollen. Je nach Zusammensetzung und Wissensstand des Projektteams sowie nach vorhandenen fachlichen Anforderungen für die einzelnen Data Marts werden dabei zwei grundsätzlich verschiedene Vorgehensweisen gewählt: Die DWH-Datenmodelle werden basierend auf den Quellsystemen (Bottom-up) oder basierend auf den Anforderungen an die BI-Applikationen (Top-down) erstellt. Beide Ansätze haben Vor- und Nachteile, die sorgfältig gegeneinander abgewogen werden müssen.

Quellsystemgetriebene Modellierung (Bottom-up)



Bei der quellsystemgetriebenen Modellierung werden die Datenmodelle der Quellsysteme analysiert und daraus ermittelt, welche Tabellen und Attribute für das Data Warehouse relevant sind. Der scheinbare Vorteil liegt darn, dass bereits mit der Datenmodellierung des Core und der Realisierung der ETL-Prozesse begonnen werden kann, bevor die fachlichen Anforderungen der BI-Applikationen und Data Marts definiert sind.

Die Relevanz der Quelldaten lässt sich jedoch ohne konkrete fachliche Anforderungen kaum beantworten. Deshalb wird oft ein pragmatischer Weg eingeschlagen und alles, was im Quellsystem an fachlichen Attributen verfügbar ist, ins Core übernommen. Auf diese Weise soll vermieden werden, dass das Datenmodell bei neuen Anforderungen erweitert werden muss. Dieser Ansatz wird oft verwendet, wenn das DWH nur ein (Haupt-) Quellsystem hat. Der Vorteil besteht hauptsächlich darin, dass die Erstellung des Core-Datenmodells sowie die Realisierung der ETL-Prozesse relativ einfach erscheint – zumindest auf den ersten Blick.

Bei mehreren Quellsystemen wird die Sache allerdings einiges komplexer, denn die Daten der unterschiedlichen Quellen müssen verknüpft und unter Umständen in eine einheitliche Form transformiert werden. Welches ist nun das führende System für die Datenstrukturen im Core? In welcher Form sollen die Daten im Core gespeichert werden?

Die Problematik soll anhand des folgenden Beispiels aufgezeigt werden: Zwei unterschiedliche Quellsysteme liefern Kundenadressen, allerdings in unterschiedlicher Form. Das Quellsystem A liefert Adressen in strukturierter Form und enthält je ein Attribut für Strasse, Hausnummer, Postleitzahl, Ortschaft und Land. Das Land wird als ISO-Code geliefert. Das Quellsystem B hingegen verwendet einen anderen Ansatz und liefert die Adressen in Form von mehreren Adresszeilen, die genau so formatiert sind wie die Adressen in gedruckter Form. Das Land wird dementsprechend als Bezeichung geliefert.

Quellsystem A	
Strasse	Lindenweg
Hausnummer	27b
Postleitzahl	9876
Ortschaft	Neu-Unterdorf
Land	CH

Quellsystem B	
Adresszeile 1	Lindenweg 27b
Adresszeile 2	9876 Neu-Unterdorf
Adresszeile 3	Schweiz

Welche Datenstruktur ist nun die richtige für das Data Warehouse? Weil diese Frage ohne Kenntnisse der fachlichen Anforderungen nicht beantwortet werden kann, wird vermutlich die technisch einfachere Variante gewählt. Da die Stuktur von Quellsystem A problemlos in Adresszeilen formatiert werden kann, die Transformation der Adresszeilen von Quellsystem B in eine strukturierte Form hingegen aufwendig ist, werden alle Adressen als Adresszeilen gespeichert. Oder noch schlimmer: Die Adressen werden in zwei separaten Adresstabellen im Core abgelegt, die jeweils die Struktur von Quellsystem A bzw. B besitzen. Mit Datenintegration hat dies jedoch nichts mehr zu tun.

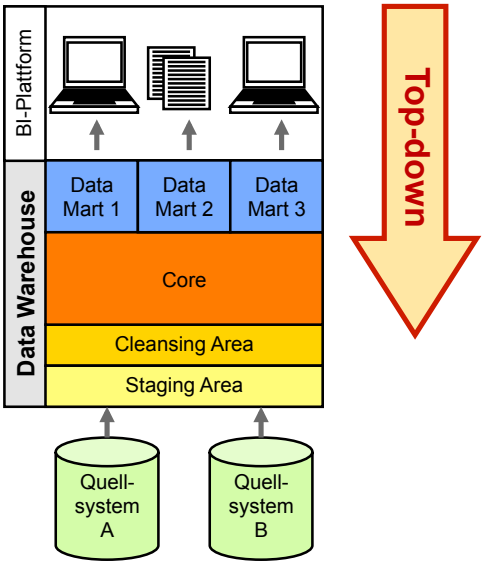
Spätestens bei der Definition der ETL-Prozesse für den ersten Data Mart wird dann ersichtlich, dass der Transformationsaufwand doch nicht so klein ist wie angenommen. Denn nun müssen die Daten so umgeformt werden, wie sie in den Auswertungen und BI-Applikationen benötigt werden. Für unser Beispiel kann dies bedeuten, dass nun die Adresszeilen wieder in eine strukturierte Form umgewandelt werden müssen oder dass zum Laden des Data Marts unterschiedliche ETL-Prozesse für die Quellsysteme A und B notwendig sind. Der eingesparte Aufwand bei der Modellierung und Implementierung des Core muss somit bei der Erstellung des Data Marts zusätzlich investiert werden. Da ein Data Warehouse typischerweise mehr als einen Data Mart enthält, müssen die erforderlichen ETL-Prozesse für jeden Data Mart erneut implementiert werden, selbst wenn in den meisten Fällen die gleichen Transformationsregeln verwendet werden.

Dies ist ein wesentlicher Nachteil der quellsystemgetriebenen Datenmodellierung. Weil von Anfang an die Datenstrukturen der Quellsysteme in Betracht gezogen werden, ist das Core-Datenmodell oft nichts anderes als ein historisiertes Abbild der Quellsysteme. Die Integratitions- und Transformationsschritte erfolgen dann meistens erst beim Laden der Data Marts. Somit entfällt ein wesentlicher Vorteil der DWH-Architektur, nämlich ein Core als zentraler Integrationslayer, der es ermöglicht, dass die komplexen Transformationen nur einmal durchgeführt werden müssen.

Trotzdem wird diese Vorgehensweise in vielen DWH-Projekten angewendet. Neben fehlenden oder unvollständigen Anforderungen zu Projektbeginn ist ein häufiger Grund, dass man möglichst alle verfügbaren Daten im Core historisieren möchte, um nachträgliche Anpassungen des Datenmodells und das Nachladen von historischen Daten zu vermeiden.

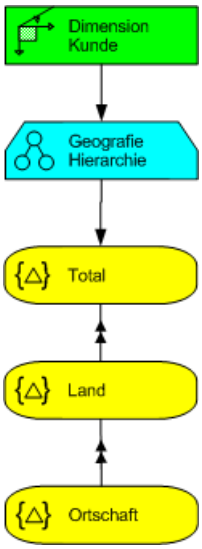
Mit der quellsystemgetriebenen Modellierung wird zwar erreicht, dass mit vertretbarem Aufwand ein Core implementiert werden kann, das Daten aus einem oder mehreren Quellsystemen laden und historisieren kann. Ob es sich dabei um die fachlich relevanten Informationen oder um einen „Datenfriedhof" handelt, ist zu diesem Zeitpunkt noch nicht ersichtlich. Bevor die Anwender der BI-Applikationen die Informationen zur Verfügung gestellt haben, müssen erst geeignete Data Marts gebaut werden. Dieser Schritt kann sehr komplex werden. Dies führt typischerweise zu Verzögerungen und höheren Projektkosten bei der Erstellung der Data Marts, mit dem Ergebnis, dass dann irgendwelche „Schnellschüsse" implementiert werden (z.B. ein direkter Zugriff auf das Core, vgl. Artikel [2]), um die Daten doch noch rechtzeitig für die Fachbereiche bereitstellen zu können.

Anforderungsgetriebene Modellierung (Top-down)



Für die anforderungsgetriebene Modellierung ist es notwendig, zuerst die fachlichen Anforderungen zu ermitteln. Dies erfolgt idealerweise zusammen mit Personen aus den Fachbereichen und aus dem DWH-Entwicklungsteam. Basierend auf den fachlichen Anforderungen werden Hierarchien, Hierarchiestufen und Attribute der Dimensionen sowie Kennzahlen und Aggregationsregeln der Fakten definiert. Diese Vorgehensweise beginnt also mit den Datenmodellen der Data Marts. Erst in einem zweiten Schritt wird dann ermittelt, aus welchen Tabellen und Attributen der Quellsysteme die erforderlichen Informationen abgeleitet werden können. Daraus können das Core-Datenmodell und die Transformationsregeln zwischen Quellsystemen und Core definiert werden.

Ein gut geeignetes Hilfsmittel für die Anforderungsanalyse ist die ADAPT-Notation[1]. Damit können fachliche Zusammenhänge zwischen Dimensionen und Kennzahlen sowie die einzelnen Hierarchien und die Verknüpfungen der Hierarchiestufen in einer anschaulichen Form dargestellt werden. Die Notation dient einerseits als Diskussionsgrundlage während der Anforderungsanalyse, andererseits als Basis für die Datenmodelle der Data Marts.



Für unser Beispiel wurde in Zusammenarbeit mit dem Fachbereich festgelegt, dass die Kunden nach Wohnort aggregiert werden sollen und dass ein Drill-up auf Länderstufe und auf Stufe Total (über alle Länder) möglich sein soll. Dies wird im ADAPT-Diagramm (siehe Abbildung rechts) dargestellt. In der Kundendimension des Data Marts werden die Attribute Ortschaft (ohne Postleitzahl) und Land (ausgeschriebene Länderbezeichnung) gespeichert. Da weitere Angaben wie Strasse oder Hausnummer nicht benötigt werden, werden sie bei einem reinen Top-down-Ansatz weder in die Data Marts noch ins Core übernommen, da diese Informationen offenbar fachlich nicht relevant sind.

Die anforderungsgetriebene Modellierung hat einige Vorteile gegenüber dem quellsystem-getriebenen Ansatz. Das Core wird übersichtlicher, da es nur fachlich relevante Daten enthält.

[1] ADAPT = Application Design for Analytical Processing Technologies, siehe [1]

Die ETL-Prozesse zum Laden der Data Marts werden einfacher, da das Core-Datenmodell aus den Data Marts abgeleitet wird und somit näher bei der Zielstruktur als bei der Struktur der Quellsysteme ist. Komplexer werden hingegen die ETL-Prozesse zum Laden der Daten aus den Quellsystemen in das Core, da unter Umständen aufwendige Transformationen durchgeführt werden müssen, wenn die Daten nicht in der geforderten Form im Quellsystem zur Verfügung stehen. Diese Transformationen müssen aber nur einmal implementiert werden, nicht wie beim Bottup-up-Ansatz für jeden Data Mart.

Häufig kommt es bei dieser Vorgehensweise vor, dass während der Spezifikation der ETL-Prozesse für das Core erkannt wird, dass Zusatzinformationen notwendig sind, die in den vorgesehenen Quellsystemen nicht verfügbar sind. In unserem Beispiel trifft dies bei den Adressen aus Quellsystem A zu. Dort ist das Land nur als ISO-Code verfügbar, wir benötigen aber die ausgeschriebene Bezeichnung der Länder. Um den entsprechenden Code-Lookup vom ISO-Code zur Bezeichnung des Landes durchführen zu können, muss als weitere Datenquelle eine geeignete Codetabelle eingelesen werden. Dies kann eine zusätzliche Tabelle aus Quellsystem A sein, aber auch ein externes File, das von einer anderen Quelle zur Verfügung gestellt wird.

Idealerweise wird das Core-Datenmodell erst nach Definition aller bekannten Data Marts erstellt, um Strukturanpassungen des Core möglichst klein zu halten. Trotzdem lassen sich bei dieser Vorgehensweise nachträgliche Erweiterungen des Core-Datenmodell nicht vermeiden. In jedem Data Warehouse werden im Laufe der Zeit neue oder geänderte Anforderungen eintreten, die zu Erweiterungen oder zur Neuerstellung von Data Marts führen. Hier liegt der hauptsächliche Nachteil der anforderungsgetriebenen Vorgehensweise. Neue Anforderungen haben oft zur Folge, dass fachlich relevante Information im Core nicht verfügbar ist und unter Umständen aufwendige Anpassungen der Datenstrukturen und ETL-Prozesse notwendig sind. Auch das Nachladen der fehlenden Attribute – sofern überhaupt möglich – ist komplex und zeitintensiv, und falls das Quellsystem nur den aktuellen Stand der Daten liefern kann, ist eine nachträgliche Historisierung von Stammdaten nicht möglich.

Der Top-down-Ansatz wird typischerweise dann verwendet, wenn die Anforderungen relativ klar sind und wenn nur einer oder mehrere ähnliche Data Marts realisiert werden sollen. Bei umfangreichen DWH-Systemen oder Enterprise Data Warehouses mit unterschiedlichsten, teilweise nur vage bekannten Anforderungen steht oft der Nachteil der aufwendigen Erweiterbarkeit im Vordergrund.

Bottom-up oder Top-down?

Die Frage nach der geeigneten Vorgehensweise für die Datenmodellierung führt in vielen DWH-Projekten zu Meinungsverschiedenheiten und unterschiedlichen Ansichten, welche Daten nun in welcher Form im Data Warehouse abgelegt werden sollen. Leider gibt es keine perfekte Lösung, die alle Bedürfnisse abdeckt. Beide hier vorgestellten Modellierungsansätze haben Vor- und Nachteile, aber aufgrund von Erfahrungen aus verschiedenen DWH-Projekten gibt es ein paar allgemeine Empfehlungen, die sich bewährt haben.

Die anwendungsgetriebene Modellierung führt erfahrungsgemäss schneller zu fachlich brauchbaren Resultaten und einem Data Warehouse, das von den Anwendern akzeptiert wird. Sie setzt aber voraus, dass möglichst früh im Projekt die fachlichen Anforderungen in einer Form erarbeitet und dokumentiert werden, die sowohl für die Fachbereiche als auch für das DWH-Team verständlich ist. In vielen Projekten ist dies die grösste Herausforderung. Während

der Realisierungsphase treten dann oft Detailprobleme auf, wenn es darum geht, woher und nach welchen Transformationsregeln die Daten aus den Quellsystemen geladen werden können. Die gleichen Probleme treten auch bei der quellsystemgetriebenen Modellierung auf, dann allerdings erst in späteren Projektphasen.

Der grösste Nachteil der Top-down-Modellierung besteht darin, dass bei zusätzlichen Anforderungen oft Erweiterungen des Core-Datenmodells und ein aufwendiges Nachladen von historischen Daten aus Quellsystemen notwendig wird. Um hier den Aufwand möglichst klein zu halten, wird empfohlen, eine Kombination aus Top-down und Bottom-up anzuwenden. Ein bewährter Ansatz kann zum Beispiel darin bestehen, dass als zusätzliche DWH-Schicht pro Quellsystem ein sogenannter Data Store implementiert wird. Darin werden alle Daten des Quellsystems geladen und historisiert, allerdings ohne fachliche Transformationen und ohne Integrationsschritte. Ins Core werden jedoch nur diejenigen Informationen geladen, die aufgrund der aktuellen Anforderungen notwendig sind. Werden die fachlichen Anforderungen erweitert, so stehen die historisierten Daten bereits im Data Store zur Verfügung und können somit mit vertretbarem Aufwand ins Core nachgeladen werden. Das Prinzip eines solchen Data Stores ist im Artikel [3] beschrieben.

Fazit

Die Wahl der richtigen Vorgehensweise für die Datenmodellierung in einem Data Warehouse ist oft schwierig und führt in vielen DWH-Projekten zu Diskussionen. Sowohl die Top-down-Modellierung als auch die Bottom-up-Modellierung haben Vor- und Nachteile. In der Regel ist ein Top-down-Ansatz basierend auf den fachlichen Anforderungen zweckmässiger, aber in vielen Fällen ist eine kombinierte Lösung aus Top-down und Bottom-up zu empfehlen.

Weitere Informationen zum Thema DWH-Datenmodellierung werden im Trivadis-Kurs „Data Warehousing mit Oracle“ (O-DWH) sowie im Buch „Data Warehousing mit Oracle – Business Intelligence in der Praxis“ (Hanser Verlag, ISBN 978-3-446- 42562-0) vermittelt.

Viel Erfolg beim Einsatz von Trivadis-Know-how wünscht Ihnen

Dani Schnider
Trivadis AG
Europa-Strasse 5
CH-8152 Glattbrugg
Internet: www.trivadis.com

Tel: +41(0)44-808 70 20
Fax: +41(0)44-808 70 21
Mail: info@trivadis.com

Literatur und Links

[1] Trivadis-Blog „ADAPT your little multidimensional world“, Andreas Nobbmann, Februar 2009
http://blog.trivadis.com/blogs/andreasnobbmann/archive/2009/02/04/adapt-your-little-multidimensional-world.aspx

[2] Trivadis-Artikel „Data Warehouse – schnell gemacht“, Dani Schnider, Mai 2009
http://www.trivadis.com/uploads/tx_cabagdownloadarea/DWH_schnell_gemacht.pdf

[3] Trivadis-Artikel „Die generierte Zeitmaschine – Historisierung auf Knopfdruck“, Dani Schnider, November 2010
http://www.trivadis.com/uploads/tx_cabagdownloadarea/DOAG_2010_Historisierung_auf_Knopfdruck.pdf