Unabhängigkeit von Datenwarenhäusern mit Sternschema

Jens Lechtenbörger, Gottfried Vossen
Lehrstuhl für Informatik
Universität Münster, Steinfurter Straße 107
D-48149 Münster, Germany
email: {lechten,vossen}@helios.uni-muenster.de

Mai 1999

Zusammenfassung

Datenwarenhäuser stellen nach derzeitiger Praxis aus einer technischen Sicht materialisierte relationale Sichten über Basisdatenbanken dar. Idealerweise sind derartige Sichten *selbstwartbar* oder sogar *unabhängig* und können sich dann selbst aktualisieren und sogar Anfragen nach Quelldaten beantworten. Es wird gezeigt, wie Unabhängigkeit als Entwurfsziel für ein Datenwarenhaus speziell im Kontext von Sternschemata erreichbar ist.

1 Einleitung

Im Kontext von Datenwarenhäusern, die von Unternehmen zum Zweck der Entscheidungsunterstützung und als Basis für OLAP-Anwendungen (On-Line Analytical Processing) eingerichtet werden, haben materialisierte Datenbanksichten in letzter Zeit zunehmende Aufmerksamkeit erfahren [18]. Typischerweise wird ein Datenwarenhaus vom technischen Standpunkt aus als redundante Sammlung materialisierter relationaler Sichten mit replizierten Daten aus möglicherweise verteilten und unabhängigen operationalen Datenbanken angesehen [4, 14, 18]. Daher kommt der effizienten Wartung der Warenhaussichten besondere Bedeutung zu. In diesem Beitrag zeigen wir, wie die Eigenschaft der *Unabhängigkeit* durch die Verwendung eines *Komplements* eines als Sternschema spezifizierten Datenwarenhauses sichergestellt werden kann und welche Vorteile sich aus diesem Vorgehen im Kontext eines Sternschemas ergeben.

Trotz der vielfältigen Resultate zur inkrementellen Wartung materialisierter Sichten (man vergleiche [10] für einen Überblick) erschwert die Entkopplung von operationalen Datenbanken und einem Warenhaus dessen Wartung und kann zu Anomalien führen [21]. In dieser Situation hat sich das Konzept der *Selbstwartbarkeit* von Datenwarenhäusern als attraktiv erwiesen [9, 13, 17, 19]; dieses wurde von uns in [15] zu (Anfrage- und Änderungs-) *Unabhängigkeit* verallgemeinert. Dabei sind Selbstwartbarkeit und *Änderungs*unabhängigkeit äquivalente Eigenschaften, die informal besagen, daß die aus Änderungen an den Basisrelationen resultierenden Änderungen am Warenhaus allein aus dem alten Warenhauszustand und Informationen über die Änderungen, jedoch ohne Rückfragen an die operationalen Datenbanken (inkrementell) berechnet werden können. *Anfrage*unabhängigkeit bedeutet dagegen, daß *jede* an die operationalen Datenbanken gerichtete Anfrage lokal im Warenhaus beantwortet werden kann. In

[17] wurde erstmals vorgeschlagen, Selbstwartbarkeit durch die Speicherung zusätzlicher Informationen im Warenhaus zu garantieren; es wurde jedoch lediglich gezeigt, wie Selbstwartbarkeit für eine einzelne PSJ-Sicht (also einen Verbund [J] mehrerer Basisrelationen, gefolgt von einer Selektion [S], gefolgt von einer Projektion [P]) erzielt werden kann. Offen bleibt, wie die dort beschriebenen Resultate auf Mengen von Sichten zu verallgemeinern sind. Demgegenüber wurde von uns in [15] vorgeschlagen, zu einer Menge von PSJ-Warenhaussichten zusätzlich ein (bezüglich einer geeigneten Ordnung auf Sichten) *minimales Komplement* [3] zu speichern, wodurch über die Selbstwartbarkeit hinaus auch Anfrageunabhängigkeit gewährleistet wird. Weiterhin wurde in [15] gezeigt, wie ein minimales Komplement einer Menge von PSJ-Sichten berechnet werden kann.

In dieser Arbeit greifen wir den Begriff der Unabhängigkeit eines Datenwarenhauses auf der Grundlage von Komplementen auf und untersuchen anhand einer Fallstudie seine Bedeutung für ein Warenhaus, das basierend auf einem *Sternschema* [4, 14] definiert ist. Weil für die Definition eines auf einem Sternschema basierenden Datenwarenhauses über mehreren Datenbanken jedoch zusätzlich zu Projektion, Selektion und Verbund auch die Vereinigung benötigt wird, wird die in [15] präsentierte Berechnung von Komplementen entsprechend erweitert. Darüber hinaus geben wir hier die Einschränkung auf, daß Komplemente auf denselben Schemata definiert werden wie die Basisrelationen, und definieren Komplemente auch auf Teilschemata der Basisrelationen, wodurch intuitiv „kleinere“ Komplemente erzeugt werden können. Formale Betrachtungen hinsichtlich der Größe der generierten Komplemente sind jedoch nicht Gegenstand dieser Arbeit.

In Abschnitt 2 formalisieren wir nach der Vorstellung einiger Notationen die zentralen Begriffe „Unabhängigkeit“, „Komplement einer Sicht“ und „Warenhaus mit Sternschema“ und zeigen anschließend, wie ein Komplement eines Datenwarenhauses mit Sternschema berechnet werden kann. Danach beschreiben wir in Abschnitt 3 eine Fallstudie zur Spezifikation eines unabhängigen Warenhauses. Zunächst werden in Abschnitt 3.1 ein Anwendungsszenario präsentiert und ein Sternschema entworfen. In Abschnitt 3.2 untersuchen wir das aus diesem Sternschema resultierende Warenhaus dann hinsichtlich seiner Unabhängigkeitseigenschaften. Abschnitt 4 beendet diesen Beitrag mit abschließenden Bemerkungen und Anregungen für zukünftige Arbeiten.

2 Notationen, Unabhängigkeit, Komplemente

Wir betrachten im folgenden eine Menge von n relationalen Datenbanken, die zumindest das Teilschema $D = \{R_1, \ldots, R_k\}$ gemeinsam haben, wobei jedes $R \in D$ ein Relationenschema ist und die in den einzelnen Datenbanken verwendeten Schlüssel global eindeutig sind [20]. (Das Schema D kann auch ein Exportschema einer Datenbank mit beliebigem Datenmodell sein.) Falls diese Voraussetzung nicht gegeben ist, muß zunächst ein Integrationsprozeß durchgeführt werden, etwa gemäß [6, 7, 5]. Um die Relationenschemata der verschiedenen Datenbanken unterscheiden zu können, verwenden wir die Notation $DB_j.R_i$, wenn wir von Relationenschema R_i zu Datenbank j sprechen. Die Menge der Attribute eines Relationenschemas $R \in D$ wird mit $attr(R)$ bezeichnet. Weiterhin nehmen wir an, daß für jedes Relationenschema R genau ein Schlüssel $key(R) \subseteq attr(R)$ definiert ist. Zur Definition von Sichten (und insbesondere von Datenwarenhäusern) benutzen wir die relationale Algebra.

In diesem Beitrag wird das Warenhaus, gängiger Praxis folgend [4, 14], mittels eines

sogenannten Sternschemas durch materialisierte relationale Sichten über operationalen Datenbanken organisiert, in dem es eine zentrale Faktentabelle und eine weitere Tabelle für jede relevante Dimension gibt. Die Tupel der Faktentabelle, auch als Fakten bezeichnet, bestehen aus zwei Teilen:

1. Einem Punkt im mehrdimensionalen Raum, angegeben durch Koordinaten in Form von Verweisen auf Dimensionstabellen und

2. numerischen Informationen (auch Maße genannt) zu dem Punkt im mehrdimensionalen Raum, darstellbar als Funktion der Dimensionen.

Wir gehen in unserem Szenario von der (auch in [9, 15, 17] vertretenen) Annahme aus, daß ein Datenwarenhaus über *einer* operationalen Datenbank durch PSJ-Sichten definiert werden kann, also durch Ausdrücke der Form $\pi_A(\sigma_\phi(R_{i_1} \bowtie \ldots \bowtie R_{i_m}))$ für geeignete Attributmengen A und Selektionsbedingungen ϕ. Aufgrund der unterstellten Homogenität der Datenbankschemata bei n operationalen Datenbanken beschreiben wir die Fakten- und Dimensionstabellen durch *Vereinigungen* von PSJ-Sichten. Ferner nehmen wir (wie üblich) an, daß es zumindest eine Dimension gibt, in der ortsbezogene Information gehalten wird, und daß zu jedem Faktum gespeichert wird, aus welcher operativen Datenbank es stammt. Weitere Relationen werden im Warenhaus dann aufbauend auf diesen Tabellen definiert und unterliegen keinerlei Einschränkungen, insbesondere können Aggregationen verwendet werden. Wir vernachlässigen im folgenden jedoch Relationen des Warenhauses, die dort lokal oder aufbauend auf den Fakten- und Dimensionstabellen definiert sind, weil deren Wartung lokal im Warenhaus durch bekannte inkrementelle Algorithmen gewährleistet werden kann. Wir schreiben eine Warenhausdefinition als Sternschema somit in der Form

$$W = \{Fakten, Dim_1, \ldots, Dim_d\},$$

wobei die Faktentabelle $Fakten$ und die Dimensionstabellen Dim_i als Vereinigungen von PSJ-Sichten über relationalen Datenbanken mit Schema D definiert (und über schlüsselbasierte Inklusionsbedingungen miteinander verbunden) sind.

Von einem technischen Standpunkt aus kommt der Wartung des Datenwarenhauses, also der Aktualisierung der Fakten- und Dimensionstabellen nach Änderungen in den operationalen Datenbanken, eine große Bedeutung zu, um Auswertungen auf konsistenten Datenbeständen garantieren zu können. In diesem Kontext hat sich zunächst die Eigenschaft der Selbstwartbarkeit als attraktiv erwiesen [17, 9, 13, 19], die von uns in [15] zu (Änderungs- und Anfrage-) Unabhängigkeit erweitert wurde. Formal lassen sich diese Konzepte wie folgt definieren:

Definition 2.1 Sei W ein Datenwarenhaus über Basisrelationen D.

1. W ist *änderungsunabhängig* (oder *selbstwartbar*), wenn es die folgende Bedingung erfüllt: Wenn eine Änderung u den Zustand der Basisrelationen von d zu d' überführt, dann gibt es einen Warenhauszustand w' so, daß w' basierend auf dem aktuellen Warenhauszustand und der Änderung u ausgedrückt werden kann und daß w' der zu d' korrespondierende Warenhauszustand ist, also $w' = W(d')$.

2. W ist *anfrageunabhängig*, falls es für jede Anfrage Q über D eine Anfrage $\overline{Q}$ über W gibt mit $Q = \overline{Q} \circ W$, das heißt $Q(d) = \overline{Q}(W(d))$ für alle Zustände d von D.

□

In [15] haben wir gezeigt, daß die Anfrageunabhängigkeit eines Warenhauses dessen Änderungsunabhängigkeit impliziert und daß Anfrageunabhängigkeit durch die zusätzliche Speicherung eines *Komplements* erzwungen werden kann. Der Begriff des Komplements einer Sicht geht auf [3] zurück und wird im relationalen Kontext folgendermaßen definiert:

Definition 2.2 Sei $D = \{R_1, \ldots, R_k\}$ eine Menge relationaler Schemata und $V = \{V_1, \ldots, V_l\}$ eine Menge relationaler Sichten über D. Eine Menge relationaler Sichten $C = \{C_1, \ldots, C_m\}, m \geq 0$, über D heißt *Komplement* von V (bezüglich D), wenn jedes $R_i \in D$ als relationale Sicht über $V \cup C$ ausgedrückt werden kann, also wenn jede Basisrelation aus den Sichten V und ihrem Komplement berechnet werden kann. □

Demnach reduziert sich die Spezifikation eines (änderung- und anfrage-) unabhängigen Warenhauses auf die Definition der für Anwendungen benötigten Sichten, gefolgt von der Berechnung eines zugehörigen Komplements. Es sei an dieser Stelle lediglich darauf hingewiesen, daß eine Menge von Sichten im allgemeinen viele Komplemente besitzt, so daß man versuchen wird, ein unter gewissen Voraussetzungen „minimales" Komplement auszuwählen. Wir vertiefen diese Problematik hier nicht, sondern beschränken uns auf eine plausible Art der Berechnung „guter" Komplemente.

Die Resultate aus [15] sind in dem hier betrachteten Kontext nicht unmittelbar anwendbar, weil dort nur PSJ-Sichten ohne Vereinigung betrachtet wurden. Im Unterschied dazu besteht in unserer Situation die Schwierigkeit, daß in den Komplementärrelationen darüber Buch geführt werden muß, aus welcher Datenbank ein Tupel ursprünglich stammt. Darüber hinaus sind die in [15] vorgestellten Minimalitätsanforderungen für unsere Zwecke zu restriktiv, weil Komplementärsichten dort dieselben Schemata wie die Basisrelationen besitzen. Es ist jedoch intuitiv klar, daß man beispielsweise für eine Sicht

$$V = \pi_A(R),$$

wobei $key(R) \subset A \subset attr(R)$, mit

$$C = \pi_{(attr(R) \setminus A) \cup key(R)}(R)$$

ein Komplement von V bezüglich R auf einem Teilschema von R erhält (mit $R = V \bowtie C$), das „kleiner" als R selbst ist. Daher lassen wir hier auch (Teilmengen von) Projektionen der Basisrelationen als Komplementärsichten zu.

Wir setzen zur Vereinfachung der nachfolgenden Komplementberechnung voraus, daß alle Informationen der Basisrelationen im Warenhaus als gleichermaßen relevant gelten und daß deswegen keine Selektionen benötigt werden. Eine Erweiterung auf Sichten mit Selektionen ist durch Hinzunahme weiterer Komplementärsichten möglich, aber für die Fallstudie in Abschnitt 3 nicht erforderlich.

Zentral für eine Komplementberechnung sind folgende Beobachtungen: Durch die referentielle Integrität zwischen Faktentabelle und Dimensionen besitzt jedes Faktum in jeder Dimension einen Verbundpartner, so daß bei der Berechnung des natürlichen Verbundes zwischen Fakten und Dimensionen alle Informationen der Faktentabelle erhalten bleiben. Unter der Annahme, daß Informationen über die Herkunft eines Tupels aus einer speziellen operationalen Datenbank in einem eigenen, hier mit DB bezeichneten Attribut der Faktentabelle und in genau einer zugehörigen Dimensionstabelle abgelegt werden, sind umgekehrt Dimensionstupel für die Komplementberechnung nur dann

nützlich, wenn sie ein Faktum als Verbundpartner besitzen. Daher befinden sich alle Informationen aus den operationalen Datenbanken, die nicht im Komplement gespeichert werden müssen (weil sie schon im Warenhaus vorliegen), auch im natürlichen Verbund aller Sichten in W. Dieser wird im weiteren mit JW bezeichnet, die Menge seiner Attribute mit A_W. Um den Anteil einer Basisrelation R in Datenbank b zu bestimmen, der nicht in JW enthalten ist und daher in Komplementärsichten zu R benötigt wird, betrachten wir die gemeinsamen Attribute von R und $JW, A := attr(R) \cap A_W$, und unterscheiden drei Fälle:

(1) „$key(R) \subset A \subset attr(R)$:“ In JW ist eine (Teilmenge einer) Projektion von R gespeichert, die den Schlüssel von R beinhaltet. Für die Tupel dieser Projektion werden die fehlenden Attribute aus der Basisrelation benötigt. Die vollständigen Tupel sollen effizient mit einem *Erweiterungsverbund* [12] berechnet werden. Zu diesem Zweck wird eine Sicht benötigt, die über den fehlenden Attributen und dem Schlüssel von R definiert ist, also über $\overline{A} = (attr(R) \setminus A_W) \cup key(R)$, und die zu den in JW gespeicherten Tupeln diese Attribute aus R extrahiert. Eine solche Sicht ist

$$C_{R_1} = \pi_{\overline{A}}(\pi_{key(R)}(\sigma_{DB=b}(JW)) \bowtie R).$$

Tupel aus R in Datenbank b, deren Schlüssel in JW nicht vorkommt (und demnach auch nicht in C_{R_1}), werden gespeichert in der Sicht

$$C_{R_2} = R \setminus (\pi_A(\sigma_{DB=b}(JW)) \bowtie C_{R_1}).$$

(2) „$A = attr(R)$:“ Die Projektion $\pi_{attr(R)}(\sigma_{DB=b}(JW))$ enthält eine Teilmenge von R in Datenbank b. Die fehlenden Tupel aus R werden in

$$C_{R_2} = R \setminus \pi_{attr(R)}(\sigma_{DB=b}(JW))$$

gespeichert. (C_{R_1} ist hier leer.)

(3) Falls weder (1) noch (2) zutreffen, fehlen Attribute von R in A_W, die *nicht* durch einen Erweiterungsverbund berechnet werden können. Daher wird

$$C_{R_1} = R$$

gespeichert. (C_{R_2} ist hier leer.)

Der folgende Satz faßt die obigen Überlegungen zusammen, wobei wir oben erwähntes Attribut DB explizit in die Darstellung der Faktentabelle aufnehmen:

Satz 2.1 Sei $D = \{R_1, \ldots, R_k\}$ eine Menge von Relationenschemata für n Datenbanken und $W = \{Fakten, Dim_1, \ldots, Dim_d\}$ eine Warenhausdefinition mit Sternschema, d.h.

$$Fakten = \bigcup_{i=1}^{n} \pi_{attr(Fakten)}(DB_i.R_{j_1} \bowtie \ldots \bowtie DB_i.R_{j_m} \bowtie \{\langle DB : i\rangle\}),$$

$$Dim_l = \bigcup_{i=1}^{n} \pi_{attr(Dim_l)}(DB_i.R_{l_1} \bowtie \ldots \bowtie DB_i.R_{l_{d_l}}),\ l \in [1, d].$$

Für $i \in [1,k]$ sei $A_i = attr(R_i) \cap A_W$ und $\overline{A_i} = (attr(R_i) \setminus A_W) \cup key(R_i)$. Dann ist $C = \{C_{i_j}^b \mid i \in [1,k], j \in [1,2], b \in [1,n]\}$, mit

$$C_{i_1}^b = \begin{cases} \pi_{\overline{A_i}}(\pi_{key(R_i)}(\sigma_{DB=b}(JW)) \bowtie DB_b.R_i), & \text{falls (1) gilt} \\ \emptyset, & \text{falls (2) gilt} \\ DB_b.R_i, & \text{sonst} \end{cases}$$

$$C_{i_2}^b = \begin{cases} DB_b.R_i \setminus (\pi_{A_i}(\sigma_{DB=b}(JW)) \bowtie C_{i_1}^b), & \text{falls (1) gilt} \\ DB_b.R_i \setminus \pi_{attr(R_i)}(\sigma_{DB=b}(JW)), & \text{falls (2) gilt} \\ \emptyset, & \text{sonst} \end{cases}$$

ein Komplement von W bezüglich D.

Die Ausdrücke der relationalen Algebra, mit denen die Basisrelationen aus den Warenhaussichten und dem Komplement berechnet werden können, lauten:

$$DB_b.R_i = \begin{cases} (C_{i_1}^b \bowtie \pi_{A_i}(\sigma_{DB=b}(JW))) \cup C_{i_2}^b, & \text{falls (1) gilt} \\ C_{i_1}^b \cup C_{i_2}^b, & \text{sonst.} \end{cases}$$

□

Es sei bemerkt, daß als einfache Optimierung der Berechnung der Komplementärsichten einer Basisrelation R der Verbund JW durch einen im allgemeinen kleineren Verbund ersetzt werden kann, indem diejenigen Dimensionen entfernt werden, die keine Attribute mit R gemeinsam haben oder deren gemeinsame Attribute mit R bereits in der Faktentabelle auftreten. Weiterhin lassen sich Selektionen der Form $\sigma_{DB=b}$ in den Verbund hineinziehen und direkt an der Faktentabelle auswerten. In Abschnitt 3.2.2 nutzen wir diese Vereinfachungen aus.

3 Eine Fallstudie

In diesem Abschnitt illustrieren wir die Spezifikation eines unabhängigen Datenwarenhauses auf der Basis eines Sternschemas anhand einer Fallstudie. Als Anwendungsszenario dient eine fiktive Großhandelskette mit $n \geq 1$ Filialen in aller Welt. Alle Filialen seien entsprechend dem Entity-Relationship-Diagramm (ERD) in Abbildung 1 identisch organisiert, was, wie erwähnt, gegebenenfalls durch vorbereitende Schritte erreicht werden kann. Auf Attribute für die einzelnen Entity-Typen kommen wir unten zu sprechen.

Die einzelnen Filialen beziehen von *Lieferanten* ihre *Produkte*, die wiederum von *Mitarbeitern* an *Kunden* weiterverkauft werden. Die Kunden werden hinsichtlich einiger *Typen* klassifiziert. Die Geschäftsbeziehungen zwischen Filialen und Kunden werden auf *Rechnungen* festgehalten, die bezüglich einzelner Produkte in *Rechnungspositionen* aufgeschlüsselt sind. Weitere Entitäten, die den Lagerbestand oder Bestellungen bei Lieferanten betreffen, seien hier vernachlässigt.

3.1 Darstellung der Unternehmensdatenbanken und Definition eines Datenwarenhauses

Den Ausgangspunkt unserer Überlegungen bilden die aus dem ERD in Abbildung 1 abgeleiteten Datenbankschemata der operationalen Datenbanken in den Unternehmensfilialen. Nach deren Beschreibung untersuchen wir, welche Daten dieser Datenquellen für Anwendungen im Bereich OLAP in Frage kommen und wie diese als mehrdimensionale Fakten in einem zentralen Datenwarenhaus repräsentiert werden können. Dabei

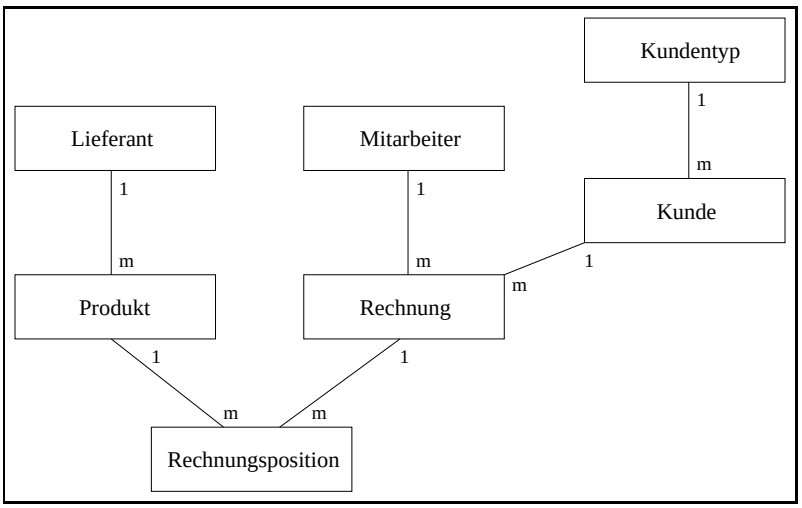

Abbildung 1: Das Anwendungsszenario als ERD.

geht es zunächst ausschließlich um die Auswahl und Darstellung dieser Fakten im Hinblick auf deren Analyse und die sich daraus ergebenden Anforderungen; insbesondere berücksichtigen wir Integrationsaspekte und Unabhängigkeitseigenschaften, die sich aus der gewählten Darstellung ergeben, erst im folgenden Unterabschnitt.

Operationale Datenbanken. Jede der n Filialen des Unternehmens verfüge über ihre eigene operationale Datenbank, wobei alle Datenbanken zumindest ein aus dem ERD der Anwendungswelt abgeleitetes Schema gemeinsam haben, das für jeden Entitätstypen aus Abbildung 1 ein entsprechendes Relationenschema besitzt und in dem die Beziehungstypen durch Fremdschlüssel abgebildet werden:

Lieferant(LID, LAnschrift, Ansprechpartner, Bankverbindung)
Produkt(PID, LID, PName, Kategorie, Preis)
Mitarbeiter(MID, MName, Position, Gehalt, PersDaten)
Kunde(KID, KTyp, KName, KAnschrift, Kontostand)
Kundentyp(KTyp, Einzelkunde)
Rechnung(RID, KID, MID, Datum, Summe)
Rechnungsposition(RID, PID, Anzahl, Zwischensumme)

In obigem Schema sind Schlüsselattribute unterstrichen, und gleichnamige Attribute verschiedener Relationenschemata implizieren eine Fremdschlüsselbeziehung; diese Konvention halten wir auch im folgenden ein. Die Attribute zu den Relationenschemata *Lieferant*, *Produkt* und *Rechnung* sind selbsterklärend. In dem Attribut *PersDaten* von *Mitarbeiter* werden persönliche Daten zusammengefaßt, die im folgenden nicht von Interesse sind (etwa Alter, Geschlecht, Familienstand). Jeder *Kunde* wird einem Typ zugeordnet (z.B. eigener Mitarbeiter, bestimmte Berufsgruppen, Universitäten, Behörden, Krankenhäuser), und es wird festgehalten, ob es sich um Einzelpersonen oder Institutionen handelt (binäres Attribut *Einzelkunde*). Eine *Rechnungsposition* gibt für jedes gekaufte Produkt an, in welcher *Anzahl* es erworben wurde und welche *Zwischensumme* sich daraus ergibt.

Im folgenden legen wir dar, welche (mehrdimensionalen) Daten zur Management-Unterstützung in einem Datenwarenhaus mit Sternschema verwaltet und untersucht

werden könnten. Es sei an dieser Stelle darauf hingewiesen, daß es bisher keine formale Methode gibt, die es erlauben würde, Maße und Dimensionen zu identifizieren und klar voneinander zu trennen [2]. Daher stellen die nachfolgenden Überlegungen die Resultate eines Entwurfsprozesses dar, der von subjektiven Entscheidungen geleitet wird.

Datenwarenhaus-Maße. Die Daten, denen in dieser Fallstudie strategische Bedeutung beigemessen wird, sind Verkaufszahlen einzelner Produkte mit dem dazugehörigen Umsatz, wie sie als Attribute *Anzahl* und *Zwischensumme* der Relation *Rechnungsposition* in den Filialdatenbanken gespeichert werden. Für diese Maße untersuchen wir im folgenden, welche Dimensionen sie bestimmen und wie sie gemeinsam mit den identifizierten Dimensionen in einer Faktentabelle im Warenhaus abgelegt werden können.

Datenwarenhaus-Dimensionen. Es stellt sich sodann die Frage, hinsichtlich welcher Dimensionen die zuvor identifizierten Maße als mehrdimensionale Fakten repräsentiert werden sollen. Zunächst kommen offensichtlich alle mit den Fakten verknüpften Informationen der operationalen Datenbanken in Betracht. Aus diesen gilt es, die relevanten Attribute auszuwählen, also diejenigen, deren Einfluß auf die Maße Aufschluß über zukünftige Maßnahmen zur Umsatzsteigerung geben könnte.

Den Ausgangspunkt bildet in diesem Szenario die Tatsache, daß die Maße *Anzahl* und *Zwischensumme* in der Relation *Rechnungsposition* funktional von den Attributen *RID* und *PID* abhängen, also von Rechnungen und Produkten, die deshalb als Dimensionen in Betracht kommen. Für statistische Auswertungen ist die Rechnungsnummer sicherlich weniger interessant als vielmehr die auf der Rechnung enthaltenen Angaben bezüglich *Kunde*, *Mitarbeiter* und *Datum*. Außerdem wird es für Auswertungen, die das gesamte Unternehmen betreffen, notwendig sein, die Fakten verschiedenen Filialen, Orten oder Ländern zuordnen zu können.

Im vorliegenden Anwendungsbeispiel verwenden wir die Dimensionen *DRechnung*, *DProdukt*, *DMitarbeiter*, *DKundentyp*, *DZeit* und *DFiliale*, die durch folgende Relationenschemata definiert sind:

DRechnung(RID)
DProdukt(PID, LID, Kategorie, Preis)
DMitarbeiter(MID, Position, Gehalt)
DKundentyp(KTyp, Einzelkunde)
DZeit(Datum, Monat, Quartal, Jahr)
DFiliale(FID, Stadt, Land, Staat)

Die Auswahl genau dieser Dimensionen läßt sich folgendermaßen begründen: Das Attribut *RID* wird benötigt, um verschiedene Positionen einer Rechnung einander zuordnen zu können (etwa für Warenkorbanalysen), wohingegen das Attribut *Summe* im Datenwarenhaus überflüssig ist, da es aus den *Zwischensummen* zu einer Rechnung durch Aggregation abgeleitet werden kann. Die Dimension *DProdukt* ermöglicht es zum einen, den Umsatz hinsichtlich bestimmter Warenkategorien zu ermitteln, wodurch Veränderungen im Sortiment angestoßen werden können. Zum anderen läßt sich auch die Akzeptanz von Produkten derselben Kategorie verschiedener Hersteller oder Preisklassen vergleichen. Produktnamen betrachten wir hier nicht als wesentlich (obwohl sie, beispielsweise bedingt durch Werbeaktionen, nicht unerhebliche Bedeutung besitzen könnten).

Unter den Merkmalen der Mitarbeiter halten wir *Gehalt* und *Position* für relevant, interessieren uns jedoch nicht für deren persönliche Daten (obwohl persönliche Daten wie

Qualifikationen, Geschlecht, Alter oder Ehestand interessante Auswertungen erlauben könnten). In der Dimension *DKundentyp* abstrahieren wir von allen Details der Kunden außer deren Typen, über die sich eventuell Zielgruppen auffinden lassen. Die Dimension *DZeit* wird aus dem Rechnungsdatum abgeleitet, lokal im Warenhaus erzeugt und dient dem Zweck, Auswertungen und Trendanalysen bei Verkaufszahlen und Umsatz auf monatlicher, quartalsweiser oder jährlicher Basis zu vereinfachen.

Eine Sonderstellung nimmt die Dimension *DFiliale* ein, die so nicht in den operationalen Datenbanken zu finden ist. Dennoch liegt es auf der Hand, daß die Beurteilung der Effizienz verschiedener Filialen oder der Bedürfnisse von Kunden bestimmter Regionen von strategischer Bedeutung ist. Deshalb werden die im Datenwarenhaus gespeicherten Fakten zusätzlich zu den aus den Filialinformationen extrahierten Dimensionen hinsichtlich ihrer Herkunft aufgeschlüsselt. Zu diesem Zweck identifiziert das Attribut *FID* die Filialen durch Nummern von 1 bis n (und übernimmt somit die Rolle des Attributs DB aus Abschnitt 2). Entsprechend wird die Datenbank der Filiale i im folgenden als DB_i bezeichnet.

Datenwarenhaus-Fakten. Damit Analysen auf Basis der Informationen des Warenhauses später möglichst detailliert durchgeführt werden können, werden die numerischen Maße originalgetreu (also ohne jegliche Aggregation) in die Faktentabelle aufgenommen. Weiterhin müssen die Schlüsselattribute der Dimensionstabellen enthalten sein, um die Daten hinsichtlich der oben genannten Dimensionen anordnen und untersuchen zu können, wodurch sich folgendes Schema ergibt:

Fakten(RID, PID, KTyp, MID, Datum, FID, Anzahl, Zwischensumme)

Diese Faktentabelle bildet zusammen mit den Dimensionstabellen das in Abbildung 2 gezeigte Sternschema des Datenwarenhauses und damit die Grundlage zum OLAP in der Unternehmenszentrale. Man beachte, daß dieser Datenbestand alle Informationen der Filialdatenbanken umfaßt, die im Datenwarenhaus genutzt werden können. Aufbauend auf diesem Datenbestand lassen sich jetzt einzelne aggregierte Sichten (beispielsweise in SQL mit Group-By-Anfragen) bis hin zu vollständigen Datenwürfeln berechnen, materialisieren und auswerten, und zwar in einer dedizierten Warenhaus-Datenbank losgelöst vom operativen Geschäft. Speicherung, Auswertung und Aktualisierung von materialisierten Sichten oder Datenwürfeln verfolgen wir hier nicht weiter; für eine eingehende Behandlung dieser Thematik verweisen wir auf [1, 2, 8, 11, 16, 20].

3.2 Unabhängigkeitsaspekte

Nach der Entscheidung über das Schema des Datenwarenhauses kann dieses in einer relationalen Datenbank implementiert werden, die wiederum mit einem initialen Datenbestand geladen wird. Damit die aus den Warenhausdaten gewonnenen Erkenntnisse immer den aktuellen Entwicklungen in den Filialen entsprechen, ist es natürlich notwendig, das Warenhaus ständig zu aktualisieren bzw. zu *warten*. Dieser Aufgabenbereich wird in Abschnitt 3.2.1 behandelt. Parallel zu den eher technischen Wartungsabläufen werden im neu geschaffenen Datenwarenhaus die eigentlichen Auswertungen auf Basis der Faktentabelle durchgeführt. Insbesondere können sich dabei Fragestellungen ergeben, für deren Lösung Analysten Informationen der Filialdatenbanken benötigen, die nicht im Warenhaus abgebildet sind. Abschnitt 3.2.2 beschreibt, in welcher Weise Anfrageunabhängigkeit in dieser Situation hilfreich ist.

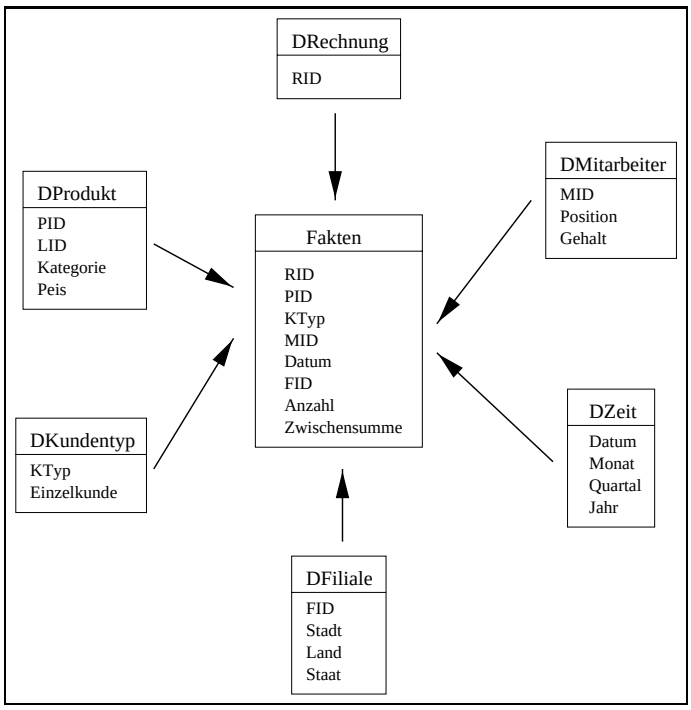

Abbildung 2: Das Datenwarenhaus als Sternschema.

Bevor wir uns der Wartung und Auswertung des Datenwarenhauses zuwenden, übertragen wir die vorliegende Situation von operationalen Datenbanken, Dimensionstabellen und Faktentabelle in den Kontext aus Abschnitt 2, um die dort genannten Resultate anwenden zu können.

Die Menge D aller Basisrelationenschemata, über denen das Datenwarenhaus definiert wurde ist, läßt sich schreiben als:

$$D = \{DB_i.Name \mid i \in [1,n], Name \in \{Lieferant,\ Produkt,\ Mitarbeiter,\ Kunde,\ Kundentyp,\ Rechnung,\ Rechnungsposition\}\}$$

Die Dimensionstabellen werden entweder lokal im Warenhaus erzeugt (*DZeit*, *DFiliale*) oder als Vereinigung von Projektionen der zugehörigen Relationen aus den Filialen geladen, beispielsweise für die Dimension *DMitarbeiter* durch die relationale Anfrage

$$\bigcup_{i=1}^{n} \pi_{MID,Position,Gehalt}(DB_i.Mitarbeiter).$$

Anders ausgedrückt entsprechen *DZeit* und *DFiliale* „normalen“ Relationen, während *DRechnung, DProdukt, DKundentyp* und *DMitarbeiter* materialisierte Sichten über den Basisrelationen sind. In der Faktentabelle müssen Informationen aus den Basisrelationen *Kunde*, *Rechnung* und *Rechnungsposition* aller Filialdatenbanken integriert werden. Diese Integration kann in der relationalen Algebra durch die Sicht

$$Fakten = \pi_{RID,PID,KTyp,MID,Datum,FID,Anzahl,Zwischensumme}\Big(\bigcup_{i=1}^{n} (DB_i.Rechnung \bowtie DB_i.Rechnungsposition \bowtie DB_i.Kunde \bowtie \{\langle FID : i\rangle\})\Big)$$

ausgedrückt werden. Die *einmalige* Auswertung dieser Anfrage entspricht dem initialen Ladevorgang des Datenwarenhauses. Wir erhalten insgesamt eine Menge von materialisierten Sichten

$$W = \{Fakten,\ DRechnung,\ DProdukt,\ DKundentyp,\ DMitarbeiter\},$$

die in der Unternehmenszentrale gewartet werden müssen.

3.2.1 Wartung — Änderungsunabhängigkeit

Im laufenden Betrieb sollen ständig Änderungen an den Filialdatenbanken in die Relationen aus W eingebracht werden, was in der Regel in einer periodischen Abfolge durchgeführt wird. Wie sich im weiteren zeigen wird, spielt es nur eine untergeordnete Rolle, welche Periode konkret gewählt wird (also etwa nach jeder Transaktion, stündlich oder täglich), solange die Perioden zumindest die Änderungen vollständiger Datenbanktransaktionen umfassen. Entscheidend ist hier hauptsächlich, daß die Filialdatenbanken das Datenwarenhaus über Änderungen informieren, woraufhin dieses geeignet reagieren muß.

Wir untersuchen als nächstes für die einzelnen Basisrelationen einer Filialdatenbank $i, i \in [1, n]$, ob das Datenwarenhaus allein anhand der Informationen über eine Änderung und seines aktuellen Zustands den Folgezustand berechnen kann, also ob es bezüglich Änderungen an einzelnen Basisrelationen *unabhängig* bzw. *selbstwartbar* ist.

Die Basisrelation *Lieferant* ist im Warenhaus nur indirekt durch Fremdschlüssel repräsentiert. Unter der (im folgenden immer getroffenen) Annahme, daß Änderungen an den Basisrelationen die zugehörigen referentiellen Integritätsbedingungen nicht verletzen, ist das Warenhaus daher unabhängig gegenüber Änderungen an *Lieferant*. Zu den Basisrelationen *Produkt, Mitarbeiter* und *Kundentyp* liegen im Warenhaus mit den zugehörigen Dimensionen Projektionen vor, die den Schlüssel beinhalten. Daher können alle Änderungen durch bekannte inkrementelle Algorithmen in das Warenhaus fortgeschrieben werden [9].

Die verbleibenden drei Relationen *Kunde*, *Rechnung* und *Rechnungsposition* gehen in der Faktentabelle einen natürlichen Verbund ein, wodurch die Änderungsunabhängigkeit in zweierlei Hinsicht verletzt wird:

1. Der erste Grund ist eher theoretischer Natur: Man stelle sich etwa vor, daß dem Warenhaus zuerst ein in die Basisrelation *Rechnung* eingefügtes Tupel μ mitgeteilt wird. Da im Warenhaus noch kein Verbundpartner aus *Rechnungsposition* bekannt ist, wird von μ nur das Attribut *RID* in der Dimension *DRechnung* gespeichert, die restlichen Attribute werden verworfen. Wenn die Rechnungspositionen zu μ später im Warenhaus eintreffen, sind die Attribute *KID, MID, Datum* nicht mehr vorhanden, und der Verbund kann nicht berechnet werden. Allerdings läßt sich argumentieren, daß dem Warenhaus zumindest alle Änderungen einer Transaktion gleichzeitig gemeldet werden sollten, wodurch eine Rechnung immer zusammen mit ihren Positionen eintreffen wird. (Da sich die Summe einer Rechnung aus den Zwischensummen ihrer Positionen ergibt, werden in realen Anwendungen vermutlich alle Informationen gemeinsam eingetragen.) Dies führt direkt zum zweiten, schwerwiegenden Problem.

2. Selbst wenn eine neue Rechnung gemeinsam mit ihren Positionen an das Datenwarenhaus übermittelt wird, kann der für die Faktentabelle notwendige Verbund nicht vollständig berechnet werden, weil keine Informationen über das Attribut

KTyp vorliegen. Mit dem Tupel, das die Rechnung repräsentiert, wird zwar *KID* übertragen, aber es besteht im Warenhaus keine Möglichkeit, diese einem Typ der Dimension *DKundentyp* zuzuordnen.

Zur korrekten Aktualisierung des Datenwarenhauses werden demnach zusätzliche Informationen benötigt. Wir wollen hier die Möglichkeit von Rückfragen an die Filialdatenbanken ausschließen, da diese bei *jeder* Mitteilung über einen Rechnungsabschluß erforderlich wären, und statt dessen die attraktive Eigenschaft der Änderungsunabhängigkeit durch die Speicherung zusätzlicher Daten im Warenhaus garantieren.

In Anlehnung an das Vorgehen in [17] (das nicht direkt anwendbar ist, weil das Warenhaus mehrere materialisierte Sichten umfaßt) muß man nach zusätzlichen Sichten suchen, die gerade die für Änderungen notwendigen Informationen umfassen. Es ist leicht einzusehen, daß obige Probleme verschwinden und das Warenhaus damit änderungsunabhängig wird, wenn man dort die zusätzliche Sicht *Kunde(KID, KTyp)* unterhält.

Man beachte, daß dieses Vorgehen eine Ad-hoc-Lösung durch „scharfes Hinsehen“ liefert. Eine exakte Charakterisierung der Zusatzinformationen, die Änderungsunabhängigkeit garantieren, bleibt weiterhin ein offenes Problem, das im nächsten Abschnitt durch die Verwendung eines Komplements umgangen wird. Wie in Abschnitt 2 erwähnt wurde, sichert die Speicherung eines Komplements im Warenhaus neben der Anfrage- auch die Änderungsunabhängigkeit.

3.2.2 Auswertungen — Anfrageunabhängigkeit

Bei der Auswahl der im Sternschema des Datenwarenhauses zu erfassenden Filialdaten muß im allgemeinen ein Kompromiß gefunden werden zwischen der Datenmenge und einem vertretbaren Analyseaufwand. Weil das Datenvolumen aller Filialen zu groß sein wird, um vollständig mit Methoden des OLAP bearbeitet zu werden (etwa weil der resultierende Datenwürfel zu groß wird), versucht man, die bestimmenden Dimensionen der Anwendungswelt zu „erahnen“ und diese in der Faktentabelle zu referenzieren. Dabei werden einige Faktoren schon im Vorfeld ausgeschlossen, weil ihr Einfluß als zu gering erscheint, um den erhöhten Aufwand zu rechtfertigen. Die verbleibenden Faktoren sollten so beschaffen sein, daß die zugehörigen Daten effizient ausgewertet werden können, um mit hoher Wahrscheinlichkeit relevante Zusammenhänge aufzudecken.

Es läßt sich allerdings (wenn überhaupt) nur schwer vorhersagen, welche Informationen sich im Laufe einer Analyse aufgrund neuer Erkenntnisse als relevant erweisen werden. Man könnte sich beispielsweise vorstellen, daß der Analyst eine besonders erfolgreiche Filiale identifiziert und daraufhin *alle*, also auch nicht in den Dimensionen abgebildete Informationen zu deren Mitarbeitern sehen möchte, um die Erfolgsfaktoren genauer zu untersuchen. Weiterhin ist es denkbar, daß wichtige Lieferanten oder Kunden aufgrund der Analyseergebnisse direkt aus der Zentrale kontaktiert werden sollen, zum Beispiel zur Aushandlung unternehmensweit angepaßter Konditionen.

In derartigen Szenarien benötigen Analysten in der Zentrale Informationen aus den Filialdatenbanken, die nicht im Warenhaus gespeichert sind und daher durch Rückfragen beschafft werden müssen. Je nachdem, wie oft solche Situationen auftreten und wie teuer (sowohl zeitlich als auch finanziell) die Rückfragen sind, kann es sich lohnen, diese Zusatzinformationen ebenfalls im Warenhaus zu materialisieren und es dadurch *anfrageunabhängig* zu machen. Um das tägliche OLAP-Geschäft im Warenhaus nicht durch diese zusätzlichen, seltener benötigten Daten zu belasten, sollten sie jedoch nicht in die Faktentabelle einbezogen, sondern in getrennten Relationen gespeichert werden.

In [15] wurde gezeigt, daß ein Datenwarenhaus genau dann anfrageunabhängig ist, wenn diese zusätzlichen Relationen ein *Komplement* der Warenhaussichten bezüglich der Basisrelationen im Sinne von [3] bilden. Entsprechend Satz 2.1 wird ein Komplement von W bezüglich D gebildet von

$$C = \{C^b_{L_1}, C^b_{P_1}, C^b_{P_2}, C^b_{M_1}, C^b_{M_2}, C^b_{K_1}, C^b_{KT_2}, C^b_{R_1}, C^b_{R_2}, C^b_{RP_2} \mid b \in [1,n]\},$$

mit

$$\begin{aligned}
C^b_{L_1} &= DB_b.Lieferant \\
C^b_{P_1} &= \pi_{PID,\ PName}(\pi_{PID}(\sigma_{FID=b}(Fakten)) \bowtie DB_b.Produkt) \\
C^b_{P_2} &= DB_b.Produkt \setminus \\
&\qquad (\pi_{PID,LID,Kategorie,Preis}(\sigma_{FID=b}(Fakten) \bowtie DProdukt) \bowtie C^b_{P_1}) \\
C^b_{M_1} &= \pi_{MID,MName,PersDaten}(\pi_{MID}(\sigma_{FID=b}(Fakten)) \bowtie DB_b.Mitarbeiter) \\
C^b_{M_2} &= DB_b.Mitarbeiter \setminus \\
&\qquad (\pi_{MID,Position,Gehalt}(\sigma_{FID=b}(Fakten) \bowtie DMitarbeiter) \bowtie C^b_{M_1}) \\
C^b_{K_1} &= DB_b.Kunde \\
C^b_{KT_2} &= DB_b.Kundentyp \setminus \pi_{KTyp,Einzelkunde}(\sigma_{FID=b}(Fakten) \bowtie DKundentyp) \\
C^b_{R_1} &= \pi_{RID,Summe}(\pi_{RID}(Fakten) \bowtie DB_b.Rechnung) \\
C^b_{R_2} &= DB_b.Rechnung \setminus \pi_{RID,KID,MID,Datum}(\sigma_{FID=b}(Fakten)) \\
&= \emptyset \\
C^b_{RP_2} &= DB_b.Rechnungsposition \setminus \\
&\qquad \pi_{RID,PID,Anzahl,Zwischensumme}(\sigma_{FID=b}(Fakten)) \\
&= \emptyset
\end{aligned}$$

Komplementärsichten, die bereits laut Satz 2.1 leer sind, sind dabei nicht aufgeführt; ferner beachte man, daß $C^b_{R_2}$ und $C^b_{RP_2}$ aufgrund der Annahme, daß Rechnungen immer gemeinsam mit ihren Rechnungspositionen eingetragen werden, leer sind.

Als Beispiel zur Berechnung einer Basisrelation aus den Warenhaus- und Komplementärrelationen betrachten wir *Produkt*:

$$\begin{aligned}
&DB_b.Produkt = \\
&\quad (C^b_{P_1} \bowtie \pi_{PID,LID,Kategorie,Preis}(\sigma_{FID=b}(Fakten) \bowtie DProdukt)) \cup C^b_{P_2}
\end{aligned}$$

Basierend auf obigem Komplement C und den Ausdrücken zur Berechnung der Basisrelationen können alle Anfragen an operationale Datenbanken lokal im Warenhaus beantwortet werden, indem in der Anfrage referenzierte Basisrelationen durch ihre äquivalente Darstellung in Form von Warenhaussichten und Komplement ersetzt werden.

Falls die Speicheranforderungen des gesamten Komplements zu groß sind, ist es möglich, Komplementärsichten nur für einzelne, häufig benötigte Basisrelationen im Warenhaus zu materialisieren und für andere Basisrelationen Rückfragen an die operationalen Datenbanken vorzusehen.

4 Zusammenfassung und Ausblick

In dieser Arbeit haben wir den in [15] eingeführten Begriff der Unabhängigkeit eines Datenwarenhauses im Kontext von Sternschemata zur Herstellung von Selbstwartbarkeit verwendet. Damit wurde gezeigt, daß es sich bei diesem Begriff um einen praktikablen Ansatz handelt, wenn man unterstellt, daß Sternschemata mit relationaler Algebra formalisiert und die Berechnung eines Komplements des Sternschemas effizient durchführbar sind. Unsere Fallstudie zeigt einerseits, daß Sternschemata im allgemeinen nicht selbstwartbar sein werden, so daß man zu ihrer Wartung zusätzliche Information benötigt; zur Gewinnung dieser Information ist bisher kein allgemeines Vorgehen bekannt. Unser in Satz 2.1 aufgestelltes Berechnungsschema für Komplemente kann in dieser Situation Abhilfe schaffen, da durch die Hinzunahme eines Komplements zu einem Warenhaus dieses unabhängig wird; wie wir im Laufe der Fallstudie angedeutet haben, ist Unabhängigkeit wichtig sowohl für Wartung durch Selbstwartbarkeit als auch für Analysen durch Anfrageunabhängigkeit.

Andererseits deutet die hier entwickelte Fallstudie eine Richtung für eine zu entwickelnde „Entwurfsmethodologie“ für Datenwarenhäuser mit wünschenswerten Eigenschaften an, wobei offensichtlich noch etliche Fragen weiterer Untersuchungen bedürfen. Stellvertretend nennen wir Minimalität für unsere Art von Komplementen, den Zusammenhang zwischen Anfrage- und Änderungsunabhängigkeit oder die Bestimmung minimaler Zusatzinformation für Änderungsunabhängigkeit.

Danksagung. Wir danken D. Laurent (Université de Tours) und N. Spyratos (Université Paris-Sud Orsay) für zahlreiche Diskussionen im Rahmen unseres gemeinsamen PROCOPE-Projekts 312/pro-gg, die eine wichtige Voraussetzung für den vorliegenden Beitrag gebildet haben.

Literatur

[1] S. Agarwal, R. Agrawal, P. Deshpande, A. Gupta, J. Naughton, R. Ramakrishnan, S. Sarawagi, “On the Computation of Multidimensional Aggregates,” Proc. 22nd VLDB 1996, 506–521.

[2] R. Agrawal, A. Gupta, S. Sarawagi, “Modeling Multidimensional Databases,” Proc. 13th ICDE 1997, 232–243.

[3] F. Bancilhon, N. Spyratos, “Update Semantics of Relational Views,” ACM TODS 1981, 6 (4), 557–575.

[4] S. Chaudhuri, U. Dayal, “An Overview of Data Warehousing and OLAP Technologies,” SIGMOD Record 26 (1), 1997, 65–74.

[5] S. Conrad, *Föderierte Datenbanksysteme — Konzepte der Datenintegration*, Springer-Verlag, 1997.

[6] C. Fahrner, *Schematransformationen in Datenbanken*, DISDBIS 29, infix 1997.

[7] C. Fahrner, G. Vossen, “Transformation relationaler Datenbank-Schemas in objektorientierte Schemas gemäß ODMG-93,” BTW 1995, 111–129.

[8] A. Gupta, V. Harinarayan, D. Quass, “Aggregate-Query Processing in Data Warehousing Environments,” Proc. 21st VLDB 1995, 358–369.

[9] A. Gupta, H.V. Jagadish, I.S. Mumick, "Data Integration using Self-Maintainable Views," Proc. 5th EDBT 1996, LNCS 1057, 140–144.

[10] A. Gupta, I.S. Mumick, "Maintenance of Materialized Views: Problems, Techniques, and Applications," IEEE Data Engineering Bulletin 1995, 18 (2), 3–18.

[11] V. Harinarayan, A. Rajaraman, J.D. Ullman, "Implementing Data Cubes Efficiently," Proc. ACM SIGMOD 1996, 205–216.

[12] P. Honeyman, "Extension Joins", Proc. 6th VLDB 1980, 239–244.

[13] N. Huyn, "Multiple-View Self-Maintenance in Data Warehousing Environments", Proc. 23rd VLDB 1997, 26–35.

[14] W.H. Inmon, *Building the Data Warehouse*, John Wiley & Sons, 2nd ed., 1996.

[15] D. Laurent, J. Lechtenbörger, N. Spyratos, G. Vossen, "Complements for Data Warehouses," Proc. 15th IEEE ICDE 1999.

[16] I.S. Mumick, D. Quass, B.S. Mumick, "Maintenance of Data Cubes and Summary Tables in a Warehouse,", Proc. ACM SIGMOD 1997, 100–111.

[17] D. Quass, A. Gupta, I.S. Mumick, J. Widom, "Making Views Self-Maintainable for Data Warehousing," Proc. PDIS 1996.

[18] N. Roussopoulos, "Materialized Views and Data Warehouses," SIGMOD Record 27 (1), 1998, 21–26.

[19] H. Shu, "View Maintenance Using Conditional Tables," Proc. 5th DOOD 1997, Springer LNCS 1341, 67–84.

[20] G. Vossen, *Datenmodelle, Datenbanksprachen und Datenbankmanagement-Systeme*, R. Oldenbourg Verlag, 3. Auflage 1999.

[21] Y. Zhuge, H. Garcia-Molina, J.L. Wiener, "Multiple View Consistency for Data Warehousing," Proc. ICDE 1997, 289–300.