Kurzfassung der Dissertation: The $\mathbb{C}$OMET Temporal Data Warehouse

Christian Koncilia

Universität Klagenfurt
Institut für Informatik-Systeme
koncilia@isys.uni-klu.ac.at

Zusammenfassung Die vorliegende Dissertation zielte darauf ab ein Data Warehouse Modell zu definieren und prototypisch zu implementieren, welches den korrekten Umgang mit Änderungen in den Dimensionsstrukturen erlaubt und darüber hinaus Mehrperiodenanfragen auch über Strukturbrüche hinweg korrekt beantworten kann. Um dieses Ziel zu erreichen wurden bereits vorhandene Techniken aus den Bereichen temporaler Datenbanken, Data Warehousing und OLAP (On-Line Analytical Programming) erweitert und miteinander kombiniert.

1 Einleitung & Motivation

Data Warehouses bzw. Data Marts sind meist materialisierte Views über strukturierte oder semi-strukturierte Datenbestände aus unterschiedlichen Datenquellen [WB97]. Eines der wichtigsten Anwendungsgebiete von Data Warehouses besteht in der Analyse von unternehmensrelevanten Datenbeständen mittels On-Line Analytical Processing (OLAP) Technologien [CT98]. Dazu wird eine multidimensionalen Sicht auf die zu analysierenden Kennzahlen, wie z. B. Gewinn, Umsatz oder Absatz erzeugt.

Ein systemimmanentes Problem von Data Warehouses und OLAP-Systemen ist der Umgang mit Änderungen in den beschreibenden Strukturen. Dieses Problem lässt sich auf die implizite Annahme solcher Systeme, dass die einzelnen Dimensionen zueinander orthogonal sind, zurückführen. Gerade in analytischen Systemen ist aber eine korrekte Berücksichtigung solcher Änderungen notwendig, da im Unterschied zu herkömmlichen Produktionssystemen kein Snapshot zu einem bestimmten Zeitpunkt (siehe [Sno85]) sondern Daten für mehrere Perioden entlang der Dimension '*Zeit*' betrachtet werden. Soll zum Beispiel die ökonomische Entwicklung von europäischen Staaten über die letzten 20 Jahre analysiert werden, so ist es von essentieller Bedeutung, die Wiedervereinigung Deutschlands, die Trennung der Tschechoslowakei, etc. korrekt zu berücksichtigen.

In derzeit eingesetzten OLAP-Systemen lassen sich bei Strukturbrüchen (wie z. B. der Aufnahme eines neuen Produkts in das Produktportfolio, oder dem Auflösen einer Vertriebsniederlassung) keinerlei Beziehungen zwischen den unterschiedlichen Strukturversionen angeben. Bei der Datenanalyse kommt es dann unweigerlich zu Problemen bzgl. der Aussagekraft der Kennzahlen, sobald Analysen erstellt werden, die Daten aus Perioden vor und nach Strukturänderungen verwenden.

Wir wollen dies kurz an einem Beispiel aus dem medizinischen Sektor zeigen. In einem Data Warehouse zur Analyse medizinischer Daten wurden Diagnosen mittels des ICD Codes („International Statistical Classification of Diseases and Related Health Problems") repräsentiert. Diese Kodierungen änderten sich jedoch bei der Umstellung von ICD Version 9 (ICD-9) auf ICD Version 10 (ICD-10). So änderte sich z.B. die Kodierung für „Bösartige Neubildung des Magens" von *151* im ICD-9 zu *C16* im ICD-10. Andere Diagnosen wurden wiederum umgruppiert. So gehören die „Zerebralen transitorische ischämische Attacken und verwandte Syndrome" nun nicht mehr zu den „Krankheiten des Kreislaufsystems" sondern zu den „Krankheiten des Nervensystems". Bei anderen Kodierungen wurde die Granularität verändert, was zu 3.000 zusätzlichen Kodierungen im ICD-10 führte. Die Frage die sich nun stellt ist, wie man Anfragen der Art „Wie entwickelte sich die Anzahl der Leberkrebspatienten in den letzten 10 Jahren" korrekt beantworten kann, wenn sich vor drei Jahren die Kodierungen – und damit die Strukturen des Data Warehouses – geändert haben.

Um dieses Problem zu lösen, sind folgende Erweiterungen bisher bekannter Data Warehouse Modelle notwendig:

- **Temporalisierung**: Schema- und Instanzdaten müssen mit Zeitstempeln versehen werden. Diese Zeitstempel repräsentieren die Gültigkeitszeit[1] (engl. *Valid Time*) der Elemente, d.h. die Zeit zu denen das Element in der Realwelt den angegebenen Zustand, bzw. die angegebene Wertebelegung aufweist [JD98].
- **Strukturversionen**: aus der temporalen Speicherung von Schema- und Instanzdaten ergibt sich die Notwendigkeit mit unterschiedlichen Strukturversionen umzugehen.
- **Transformationsfunktionen**: Um Mehrperiodenanalysen zu ermöglichen, müssen Daten (Zellwerte) von einer Strukturversionen in eine andere mittels entsprechender Funktionen transformiert werden können.

Im Folgenden werden diese Erweiterungen kurz beschrieben werden. Aus Platzgründen wird auf Angabe und Besprechung aller formalen Definitionen und verwandter Arbeiten verzichtet.

2 Temporale Erweiterung eines Data Warehouse

Üblicherweise wird in einem Data Warehouse eine multidimensionale Sicht auf die zu analysierenden Daten aufgezogen. Eine solche multidimensionale Sicht

[1] Die Transaktionszeit (der Zeitpunkt zu dem eine Werteänderung dem Datenbanksystem bekannt gegeben wurde [JD98]) blieb in der Dissertation unberücksichtigt. In einem 2003 erschienenen Kurzbeitrag zur 15. International Conference on Advanced Information Systems Engineering wurde das formale Modell um die Transaktionszeit erweitert.

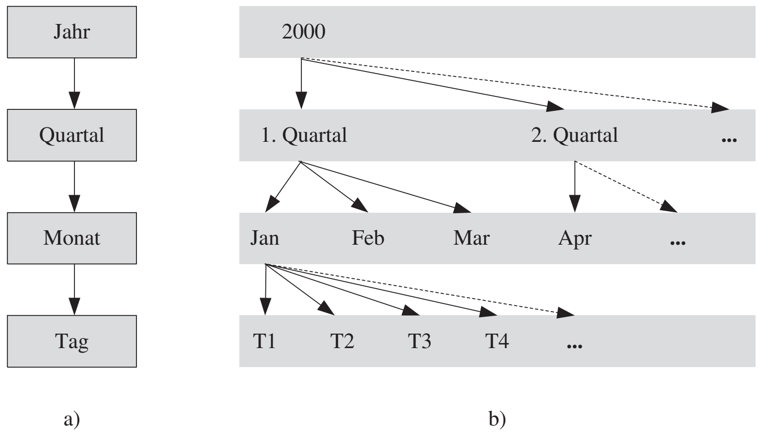

Abbildung 1. Ein Beispiel für Kategorien und zugehörige Dimensionselemente

auf Daten besteht im Wesentlichen aus einer Menge von Dimensionen die einen mehrdimensionalen Raum bzw. Datenwürfel aufspannen [VS99]. Zellen in diesem Raum beinhalten die zu analysierenden Werte und werden über die Dimensionen bzw. deren Ausprägungen referenziert. Üblicherweise wird ein solcher Datenwürfel durch eine Zeit-, eine Kennzahlen- und mehrere Strukturdimensionen aufgespannt.

Die Zeitdimension definiert dabei die Granularität der Gültigkeitszeit (der Zeitraum in dem ein Objekt in der Realwelt den angegebenen Zustand, bzw. die angegebene Wertebelegung aufweist [JD98]) – also das ,,Chronon[2]", bzw. den kleinsten nicht weiter zerlegbaren Zeitintervall (Monate, Tage, Stunden, ...).

Die Kennzahlendimension beschreibt die zu analysierenden Kennzahlen inkl. deren hierarchischem Zusammenhang. Sie beschreibt also ,,*Was*" analysiert wird. Typische Beispiele wären Absatz, Umsatz oder Kosten. Kennzahlen werden oft auch als Fakten bezeichnet.

Die Strukturdimensionen legen fest, ,,*Wie*" die definierten Kennzahlen analysiert werden können. Beispiele für solche Strukturdimensionen wären $Produkte$: $Produktgruppen \rightarrow Produkte$ oder $Gebiet$: $Staat \rightarrow Stadt \rightarrow Filiale$, etc.

Eine Dimension selbst besteht aus einer Menge von Kategorien (engl. *Categories*) denen wiederum eine Menge von Dimensionsdaten (engl. *Dimension Members*) zugeordnet ist. Dimensionsdaten entsprechen dabei der Extension der entsprechenden Dimensionen. So gehören – wie in Abbildung 1 gezeigt – zur Dimension *Zeit* z. B. die Kategorien *Jahr*, *Quartal*, *Monat* und *Tag*. Diesen wiederum sind die Dimensionsdaten *2000*, *1. Quartal*, *Jan* und *1. Jan. 2000*, die in der hierarchischen Beziehung $2000 \rightarrow 1.Quartal \rightarrow Jan \rightarrow 1.Jan.2000$ stehen, zugeordnet. Die hierarchischen Beziehungen definieren dabei den Konso-

[2] [JD98] definiert ein Chronon als ,,a non-decomposable time interval of some fixed, minimal duration"

lidierungspfad. So wird zum Beispiel durch die oben angegebene Dimension *Zeit* festgelegt, dass sich ein Monat aus der Summe[3] der einzelnen Tage ergibt.

Somit kann sowohl das Schema (Kategorien und deren hierarchische Beziehungen) als auch die Instanzen (Dimensionsdaten und deren hierarchische Beziehungen) jeder Dimension als gerichteter, azyklischer Graph aufgefasst werden. Dabei entspricht jedes Dimensionsdatum/jede Kategorie einem Knoten und jede hierarchische Beziehung zwischen Dimensionsdaten/Kategorien einer Kante. Ein multidimensionales System, welches die Gültigkeitszeit unterstützen soll, muss alle Knoten und Kanten eines solchen Graphen mit Zeitstempeln der Form $[T_s, T_e]$ versehen. T_s steht dabei für den Startzeitpunkt und T_e für den Endzeitpunkt. Steht der Endzeitpunkt noch nicht fest, so wird dieser durch ∞ repräsentiert. Weiters gilt, dass $T_e \geq T_s$. $[T_s, T_e]$ definiert also den Intervall von wann bis wann ein bestimmter Zustand gültig war.

Diese intuitive Beschreibung eines temporalen, multidimensionalen Systems wird in der Dissertation in Kapitel 3 durch eine formale Definition ergänzt.

3 Strukturversionen

Mittels der in [JD98] beschriebenen temporalen Projektion und temporalen Selektion können nun so genannte *Strukturversionen* (SV) ermittelt werden. Intuitiv kann eine Strukturversion definiert werden als eine multidimensionale Struktur die für einen bestimmten Zeitraum gültig ist, und in der alle für diesen Zeitraum definierten Elemente (Schema- und Instanzelemente) gültig sind. Dies bedeutet also, dass es innerhalb einer Strukturversion keine unterschiedlichen Versionen eines Elements geben kann, bzw. umgekehrt, dass für jede Änderung eines Elements automatisch eine neue Strukturversion generiert wird, so für den angegebenen Zeitintervall noch keine entsprechende Strukturversion existiert.

Eine formale Definition der Strukturversionen wird im Kapitel 3.2 der Dissertation gegeben.

Schema- und Instanzdaten können mit Hilfe der drei Funktionen `INSERT`, `DELETE` und `UPDATE` geändert werden. Im Unterschied zu den Basisoperationen in (relationalen) Datenbanken müssen diese Funktionen hier einen zusätzlichen Parameter berücksichtigen, welcher die Gültigkeitszeit des betroffenen Datums repräsentiert. Ein weiterer gravierender Unterschied zu nicht-temporalen Datenbanken ist, dass hier die Funktion `DELETE` ein Datum nicht löscht, sondern lediglich den Endzeitpunkt der entsprechenden Gültigkeitszeit setzt.

Durch Verwendung dieser drei Basisfunktionen können komplexere Modifikationen an Dimensionsdaten vorgenommen werden. Wir unterscheiden dabei zwischen den folgenden Änderungen:

1.) SPLIT: Ein Dimensionsdatum wird in n Dimensionsdaten aufgespaltet.
2.) MERGE: n Dimensionsdaten werden zu einem Dimensionsdatum zusammengefasst.

[3] Die Konsolidierung kann natürlich auch über andere Funktionen als die Addition erfolgen

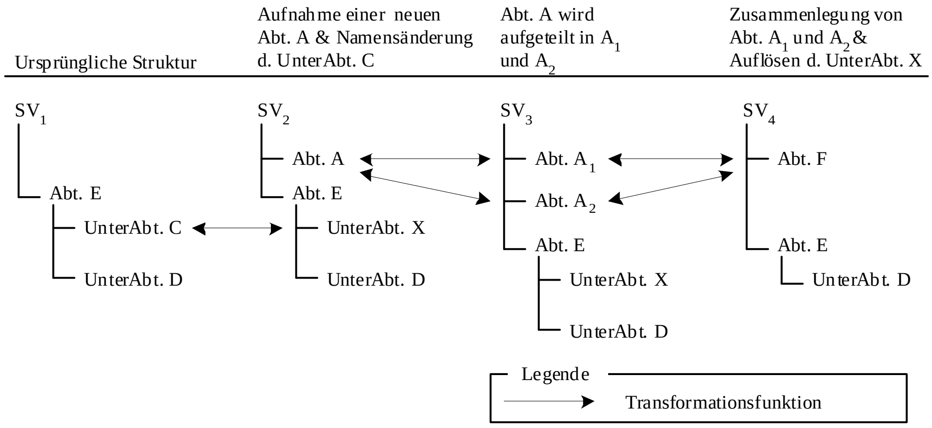

Abbildung 2. Ein Beispiel für Strukturänderungen und Transformationsfunktionen

3.) CHANGE: Ein oder mehrere Attribute eines Dimensionsdatums ändern sich.
4.) MOVE: Die hierarchische Zuordnung eines Dimensionsdatums ändert sich.
5.) NEW-MEMBER: Ein neues Dimensionsdatum wird eingefügt.
6.) DELETE-MEMBER: Ein existierendes Dimensionsdatum wird gelöscht.

4 Transformationsfunktionen

Bisher wurde kurz beschrieben, wie ein multidimensionales System um temporale Aspekte erweitert werden kann, und welche Funktionen notwendig sind um Änderungen in einem solchen System vornehmen zu können.

Im Folgenden möchte ich darauf eingehen, wie Beziehungen zwischen Dimensionsdaten unterschiedlicher Strukturversionen hinterlegt werden können. Solche Beziehungen werden z. B. hinterlegt, um dem System mitzuteilen, dass die Abteilung „*Recht & Finanzen*", welche in der Strukturversion SV_i gültig ist, nach einer Aufspaltung in die Abteilungen „*Recht*" und „*Finanzen*" in irgendeiner Beziehung mit diesen Abteilungen steht, welche in der Strukturversion SV_{i+1} gültig sind.

Durch die Verwendung von solchen Beziehungen kann ein Benutzer Daten von einer Strukturversion in eine andere Strukturversion umrechnen lassen. Er/Sie kann also z. B. die Umsätze einer alten Struktur nach der aktuell gültigen Vertriebsstruktur analysieren, oder aber auch eine Analyse der Umsätze der aktuellen Struktur nach einer alten Vertriebsstruktur durchführen.

Um solche Beziehung zu hinterlegen, werden so genannte *Transformationsfunktionen* eingeführt. Eine Transformationsfunktion ist eine generische Funktion, die jeweils ein Dimensionsdatum aus einer Strukturversion, unter Verwendung eines Gewichtungsfaktors, mit einem Dimensionsdatum einer anderen Strukturversion verbindet. Transformationsfunktionen sind weder injektiv noch surjektiv. Das heißt, ein Dimensionsdatum kann über mehrere Funktionen mit unterschiedlichen Dimensionsdaten einer anderen Strukturversion in Beziehung stehen (notwendig bei einer Split-Operation), und umgekehrt.

Transformationsfunktionen können dabei zeitlich unmittelbar benachbarte Strukturversionen miteinander in Verbindung setzen, also die Strukturversionen SV_n und SV_{n+1}. In der vorliegenden Dissertation wird gezeigt wie diese Transformationsfunktionen als Matrizen dargestellt werden können. Diese Matrizendarstellung ist für viele Fragestellungen ein gut geeignetes Beschreibungsinstrument. Unter anderem kann so gezeigt werden, wie durch einfache Matrizenmultiplikation Transformationsfunktionen zwischen zeitlich nicht benachbarten Strukturversionen ermittelt werden können. Dies ermöglicht Analysen entlang beider Zeitrichtungen, also die Analyse neuer Daten nach alten Strukturen, bzw. die Analyse alter Daten nach neuen Strukturen.

5 Implementierung

Aufbauend auf dem formalen Modell wurde das in UML verfasste $\mathbb{C}$OMET Metamodell in Kapitel 4 der Dissertation definiert. Neben einer Beschreibung des Metamodells wurden noch notwendige Integritätsbedingungen definiert. Außerdem wurde (in Kapitel 4.4) noch eine Methode zur performanteren Beantwortung von Anfragen, welche Daten aus mehreren Strukturversionen benötigen, vorgestellt. All dies bildete die Grundlage für die in den Kapiteln 6 und 7 beschriebenen möglichen Implementierungsarchitekturen, und die Implementierung einer dieser Architekturen.

In Kapitel 6 werden im Wesentlichen zwei verschiedene Architekturen einer möglichen Implementierung vorgestellt: ein direkter Ansatz (engl. *Direct Approach*) und ein indirekter Ansatz (engl. *Indirect Approach*).

Der direkte Ansatz hat den Vorteil, dass er dem Endbenutzer neben den eigentlichen Ergebnissen einer Anfrage noch zusätzliche Informationen bzgl. jener Strukturänderungen im Data Warehouse geben könnte, die einen Einfluss auf die Anfrageergebnisse haben. Der Nachteil des direkten Ansatzes ist, dass die notwendige Client Software (u.a. verantwortlich für das Stellen der Anfragen, die Präsentation der Ergebnisse, die Bereitstellung von OLAP Funktionen wie Drill-Down und Roll-Up, etc.) implementiert werden müsste.

Der zweite näher vorgestellte Ansatz – der so genannte indirekte Ansatz – ermöglicht es, bereits vorhandene Infrastrukturen zu nutzen. Dieser Ansatz setzt keine Implementierung einer Client-Software voraus, sondern kann am Markt verfügbare, nicht-temporale OLAP Standardsoftware zur Präsentation der Daten nutzen. Die Grundidee dabei ist, dass für jede gewünschte Strukturversion ein eigener Data Mart in der entsprechenden OLAP Standardsoftware generiert wird. Ein solcher Data Mart enthält nun die in einer bestimmten Strukturversion vorliegenden Strukturen und deren Zellwerte (die zu analysierenden Zahlen). Darüber hinaus enthält er jedoch auch die transformierten Zellwerte aus allen anderen Strukturversionen.

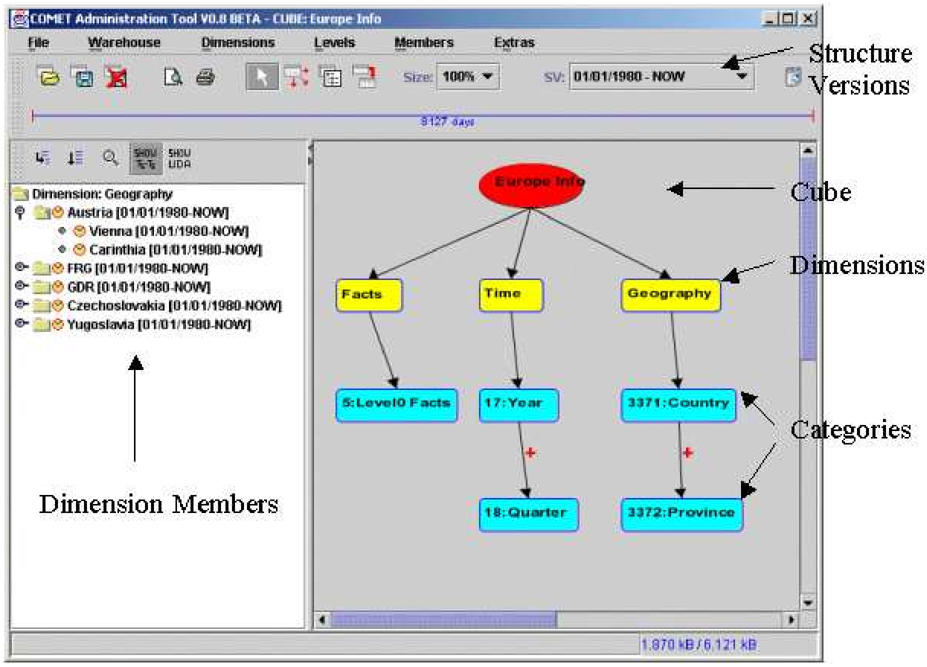

Abbildung 3. ℂOMET screenshot: Cube created with the ℂOMET Administration Tool

Im Rahmen der Dissertation wurde der indirekte Ansatz implementiert. Die Implementierung verwendet dabei zur temporalen Datenhaltung Oracle 8.1 und als Client-Software Hyperion Essbase. Die Implementierung der Administrationssoftware erfolgte in Java und besteht aus circa 30.000 Zeilen Java-Code. Die Administrationssoftware umfasst dabei Pakete zur Transformation der Daten und Generierung der Data Marts in Hyperion Essbase, und zur Wartung der Strukturen mittels einer grafischen Benutzeroberfläche (siehe Abbildung 3).

6 Fazit

In der vorliegenden Dissertation wurde ein Ansatz vorgestellt um Data Warehouses, bzw. multidimensionale Datenstrukturen um temporale Aspekte zu erweitern. Dieser Ansatz erlaubt einerseits die Historisierung von Änderungen in Dimensionsdaten und Dimensionshierarchien, und andererseits die korrekte Transformation von Daten zwischen unterschiedlichen Strukturversionen.

Das vorgestellte und prototypisch implementierte Modell ermöglicht also zusammenfassend:

- Einen korrekten Umgang mit Änderungen sowohl auf der Schema- als auch auf der Instanzebene.

- Einen korrekten Umgang mit Änderungen in den Maßeinheiten, z.B. die Umstellung von ATS auf EURO.
- Die Identifizierung von Strukturversionen, d.h. von Perioden ohne Änderungen in den Strukturen.
- Transformationsfunktionen werden zur Umrechnung von Daten zwischen unterschiedlichen Strukturversionen bereitgestellt.
- Die korrekte Beantwortung von Mehrperiodenanfragen an das System über Strukturbrüche hinweg.
- Analyse der im Data Warehouse gespeicherten Daten nach einer alten oder der aktuellen Strukturversion, wobei stets alle Daten aus sämtlichen Strukturversion in die ausgewählte Strukturversion umgerechnet werden.
- Jede Dimension kann beliebig vielen Datenwürfeln zugeordnet werden (es besteht also eine n:n Beziehung). Darüber hinaus kann jede Kategorie (d.h., jede Ebene im Schema wie z.B. die Ebene '*Staat*' der Dimension '*Regionen*') beliebig vielen Dimensionen, und jede Instanz beliebig vielen Kategorien zugeordnet werden. Dies stellt eine Erweiterung des bekannten Fact-Constellation Schemas dar.
- Der Ansatz erlaubt außerdem die korrekte Aggregation von nicht disjunkten Elementen.

Dies alles ermöglicht ein ,,sauberes" Design von Data Warehouses, welches ohne Hilfskonstruktionen zum Umgang mit Strukturänderungen auskommt. Des Weiteren erleichtert dieses System die korrekte Interpretation von Abfrageergebnissen da der Benutzer von der Pflicht entbunden wird, detailliertes Wissen über die historische Entwicklung der Strukturen zu besitzen.

Literatur

[CT98] L. Cabibbo and R. Torlone. A Logical Approach to Multidimensional Databases. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT'98)*, Valencia, Spain, 1998.

[EJS98] O. Etzion, S. Jajodia, and S. Sripada, editors. *Temporal Databases: Research and Practise.* Springer-Verlag (LNCS 1399), 1998.

[JD98] C. S. Jensen and C. E. Dyreson, editors. *A consensus Glossary of Temporal Database Concepts - Feb. 1998 Version*, pages 367–405. Springer-Verlag, 1998. In [EJS98].

[Sno85] R. Snodgrass. A Taxonomy of Time in Databases. *Proceedings of the 1985 ACM SIGMOD International Conference on Management of data*, 1985.

[VS99] P. Vassiliadis and T. Sellis. A Survey of Logical Models for OLAP Databases. In *SIGMOD Record 28*, 1999.

[WB97] M. Wu and A. Buchmann. Research Issues in Data Warehousing. In *Datenbanksysteme in Büro, Technik und Wissenschaft (BTW'97) GI-Fachtagung*, Ulm, Germany, 1997.