2 Die Integration heterogener Daten

„It is a wonder that the information processing community ever thought that *both* primitive and derived data would fit in a single database." [12]

„So wäre manches große Unternehmen sicherlich überrascht, wenn es wüßte, was es alles weiß." [13]

Gegenstand von Kapitel 2 sind Konzepte und Technologien, die eine Zusammenführung heterogener Daten unterstützen und damit dem Ziel dienen, diese Ressourcen integriert nutzen und auswerten zu können. Das einführende Kap. 2.1 stellt die hieraus erwachsenden Herausforderungen am Beispiel unterschiedlicher Formen der Heterogenität strukturierter Daten sowie von Nachteilen der Heterogenität am Beispiel von Unternehmensdaten dar. Kap. 2.2 behandelt das Konzept des Data Warehouse und geht nach einer Definition auf unterschiedliche Organisationsformen, Datenhaltung, Datenversorgung und Herausforderungen dieses Ansatzes ein. In Kap. 2.3 wird in einem Exkurs das Konzept des Online Analytical Processing (OLAP) vorgestellt, dass zur Modellierung multidimensionaler Daten dient und häufig in Verbindung mit dem Data Warehouse-Konzept genannt wird. Kap. 2.4 stellt überblicksartig alternative Ansätze zur Datenintegration vor. Ausgehend von einer Unterscheidung in virtuelle und materielle Vorgehensweisen werden die Konzepte der Mediatoren, der Migration und der föderierten Datenbanksysteme[14] behandelt. Ein abschließendes Fazit wird in Kap. 2.5 gezogen.

2.1 Einführung

Die Zusammenführung bisher isoliert vorhandener Datenressourcen - sei es zur Verbesserung der Entscheidungsqualität einzelner Unternehmen oder zur Schaffung einer geeigneteren wissenschaftlichen Ausgangsbasis spezifischer Forschungsdisziplinen - gewinnt zunehmend an Bedeutung. Entsprechend der vielfältigen Anwendungshorizonte können sich die zu integrierenden Datenquellen auf unterschiedlichen Ebenen befinden – innerhalb der einzelnen Abteilungen eines Unternehmens, der verschiedenen Projekte einer Forschungseinrichtung, verteilt auf unterschiedliche Forschungseinrichtungen einer Wissenschaftsdisziplin oder - wie im Falle des Potsdam-Instituts für Klimafolgenforschung - global verteilt auf eine Vielzahl von Forschungseinrichtungen und Datengeber unterschiedlicher Disziplinen. Durch den verfügbaren informationstechnologischen Stand können lokal oder auch global verteilte digitale Datenressourcen über Computernetze unabhängig von ihrem Standort zugänglich gemacht werden, so dass eine wesentliche technische Grundvoraussetzung für einen schnellen Zugriff auf verteilte Quellen potentiell gegeben ist. Allerdings sind die für konkrete Fragestellungen jeweils relevanten Datenquellen in den seltensten Fällen bereits im Hinblick auf eine spätere, flexible Zusammenführung mit anderen Ressourcen gestaltet worden, sondern zunächst isoliert anhand der jeweils spezifischen lokalen Anforderungen gewachsen. Auf diese Weise sind einzelne Datenräume in aller Regel von vielfältigen Heterogenitäten gekennzeichnet, die ihre integrierte Nutzung gemeinsam mit anderen Datenräumen erheblich erschweren.

2.1.1 Formen der Heterogenität am Beispiel strukturierter Daten

Datenbanken können mit ihren Konzepten zur Unterstützung bei der organisierten Verwaltung strukturierter Daten, ihrem rigiden und abfragbaren Schema sowie ihren wohldefinier-

[12] W.H. Inmon: *Building the Data Warehouse* [Inmon 1996, 19] (Hervorhebung im Original).
[13] B.-U. Kaiser: *Kritische Reflexion von Informationssystemen für das obere Management* [Kaiser 1998, 447] (Schreibweise im Original).
[14] Unter dem Begriff *Datenbanksystem* (DBS) werden Datenbankmanagementsystem (DBMS) und die zugehörigen Datenbanken zusammengefasst.

ten Zugriffschnittstellen als avancierte Ausgangsbasis für eine Bereitstellung verschiedenartiger Datenressourcen eingestuft werden. Bei einer oberflächlichen Betrachtungsweise vermag deshalb der - allerdings unzutreffende - Eindruck entstehen, dass bei einer strukturierten Organisation von Daten über Datenbanken von einer hinreichenden Homogenität verschiedener Datenräume ausgegangen werden kann. Dass dies nicht der Fall ist, belegen die verschiedenartigen Erscheinungsweisen von Heterogenität, die bei strukturierten Daten aus unterschiedlichen Quellen bereits dann auftreten können, wenn jeweils vergleichbare Ausschnitt der Welt in diesen Datenräumen repräsentiert werden. Nachfolgend werden einige ausgewählte Erscheinungsformen vorgestellt.

Für die Modellierung von Daten in Datenbanken stehen - bspw. mit dem relationalen Datenmodell, dem Netzwerkdatenmodell und dem objektorientierten Datenmodell - unterschiedliche Konzepte zur Verfügung. Da diese jeweils unterschiedliche Modellierungskonstrukte anbieten, entstehen bei der Modellierung vergleichbarer Weltausschnitte jeweils unterschiedlich strukturierte Datenbankschemata und bilden so eine der Quellen für Datenheterogenität. Ferner ist hier zu beachten, dass unterschiedliche Datenmodelle jeweils unterschiedlich reichhaltige Semantiken und Strukturen für die Modellierung bereitstellen; so stellt beispielsweise das objektorientierte Datenmodell Konzepte wie Vererbung bereit, die im relationalen Datenmodell nicht zur Verfügung stehen [Conrad 1997, 45f.] [Busse et al. 1999, 4f.].

Aber auch, wenn unterschiedliche Quellen das selbe Datenmodell verwenden, kann ein Sachverhalt durch Einsatz unterschiedlicher Modellierungskonzepte des Datenmodells auf verschiedenartige Weise in Struktur abgebildet werden [Kim, Seo 1991] [Conrad 1997, 46, 79ff.] [Busse et al. 1999, 5]. Dabei steigt die Zahl der Möglichkeiten, ein- und denselben Sachverhalt unterschiedlich darzustellen, mit der Zahl der verfügbaren Modellierungskonzepte eines Datenmodells. Bereits bei Verwendung der vergleichsweise eingeschränkten Konzepte des relationalen Datenmodells zur mehrfachen Abbildung des selben Sachverhaltes kann eine Vielzahl von Heterogenitäten auftreten. So ist es im relationalen Datenmodell bspw. möglich, den gleichen Sachverhalt entweder über Attributnamen oder Attributwerte zu modellieren[15]. Abb. 2.1 zeigt die resultierenden Tabellen für die Modellierung eines einfachen Sachverhaltes - Professoren und die von ihnen veranstalteten Seminare - einmal über Attributnamen (vgl. Abb. 2.1a) und einmal über Attributwerte (vgl. Abb. 2.1b).

(a)

Professor	Logik	Metaphysik	Ethik
Mayer	◆		
Müller	◆	◆	
Schmidt			◆

(b)

Professor	Seminar
Mayer	Logik
Müller	Logik
Müller	Metaphysik
Schmidt	Ethik

Abb. 2.1 - Unterschiedliche Modellierung des selben Sachverhaltes: (a) anhand von Attributnamen; (b) anhand von Attributwerten (in Anlehnung an [Busse et al. 1999, 5, Table 1]).

Heterogenität bei der Modellierung eines vergleichbaren Sachverhaltes durch zwei oder mehr Entwürfe - etwa für zwei Produktdatenbanken in unterschiedlichen Abteilungen eines Unternehmens - kann ferner dadurch entstehen, dass in einer Datenbank Attribute vorhanden sind, die in der anderen fehlen. Hierbei ist zu unterscheiden, ob die fehlenden Attribute implizit sind oder nicht. Fehlende *nichtimplizite* Attribute treten auf, wenn Attribute für die Aufgaben, für die eine Datenbank entworfen wurde, nicht erforderlich sind; fehlende, aber *implizite* Attribute sind solche Attribute, für die alle in der Datenbank abgelegten Datensätze den selben Wert haben würden und entsprechend aus der Semantik der Datenbank

[15] Nach Busse et al. existieren im relationalen Datenmodell insgesamt drei solcher Modellierungskonfliktarten: Relation vs. Attributname, Attributname vs. Attributwert und Relation vs. Attributwert [Busse et al. 1999, 5].

erschließbar sind [Conrad 1997, 79ff.] [Kim, Seo 1991] (vgl. Abb. 2.2).

(a)

Produkt_Typ	Produkt_ID	Farbe	Preis
A	1	blau	3000
A	2	rot	3000
B	3	blau	4000
C	4	grün	2000

(b)

Produkt_ID	Farbe	Preis
112	blau	5000
113	rot	2000
114	gelb	6000
115	rot	1000

Abb. 2.2 - Beispiel für ein fehlendes implizites Attribut: Während Tabelle (a) Produkte unterschiedlicher Produkttypen dokumentiert, verwaltet Tabelle (b) nur Produkte eines einzigen Produkttyps – das Attribut *Produkt_Typ* fehlt hier, ist jedoch implizit (eigenes Beispiel in Anlehnung an [Conrad 1997, 82]).

Einen weiteren Faktor bilden unterschiedliche Auffassungen des selben Sachverhaltes, die in unterschiedlichen Konzeptualisierungen münden können [Visser et al. 1997]. So bilden bspw. die Freiheitsgrade bei der Benennung von Attributen oder Tabellen eine weitere Quelle für Heterogenität strukturierter Daten. Hierbei können sowohl gleiche Bezeichnungen mit unterschiedlicher Bedeutung (*Homonyme*) oder unterschiedliche Bezeichnungen mit gleicher Bedeutung (*Synonyme*) auftreten [Kim, Seo 1991] [Conrad 1997, 80] [Busse et al. 1999, 5]. Die beiden Tabellen in Abb. 2.3 illustrieren die Verwendung homonymer und synonymer Attribute am Beispiel zweier Kundentabellen. In der Tabelle in Abb. 2.3a ist unter dem Attribut *Kontakt* die E-Mail-Adresse des Kunden speichert, in der Tabelle in Abb. 2.3b hingegen das Datum der letzten Kontaktaufnahme – die Namen beider Attribute sind homonym. Beide Tabellen speichern andererseits jeweils die Strasse der Kundenadresse in einem Attribut, verwenden für dieses jedoch unterschiedliche, synonyme Bezeichnungen (*Strasse* bzw. *Kunden_Str*).

(a)

Name	Kontakt	Strasse
Müller	mueller@abc.com	Hohestr.
Schmidt	schmidt@xy.de	Breitestr.

(b)

Name	Kontakt	Kunden_Str
Huber	21.03.2003	Lenaustr.
Hinze	10.12.2003	Goethestr.

Abb. 2.3 - Heterogene Benennung durch homonyme und synonyme Attribute (eigenes Beispiel).

Ferner können Werte, die jeweils die selbe Eigenschaft beschreiben, in individuellen Modellierungen unterschiedlich repräsentiert werden. So können zur Speicherung der selben Eigenschaft in einzelnen Quellen *unterschiedliche Datentypen* verwendet werden; ein Beispiel hierfür ist die Repräsentationen von Prüfungsergebnissen durch die numerischen Werte 1, 2, ..., 5 sowie durch die Zeichenketten *sehr gut*, *gut*, ..., *nicht bestanden*[16]. Auch bei Verwendung *gleicher Datentypen* können einander entsprechende Werte unterschiedlich repräsentiert werden; etwa ein- und dieselbe Straße durch Zeichenketten der Form *Breitestraße*, *Breitestrasse*, *Breitestr.*, *Breite Straße*, *Breite-Strasse*, *Breite-Str.* etc.[17] [Kim, Seo 1991] [Conrad 1997, 83]. Die nachgerade unerschöpfliche Vielzahl von Möglichkeiten verdeutlichen auch die folgenden Beispiele für mögliche Codierungen der Eigenschaft *männlich* bzw. *weiblich*: *männlich*/*weiblich*, *m*/*w*, *male*/*female*, *m*/*f*, *a*/*b*, 0/1, 1/0 etc.

Wertere präsentationen können ferner *unterschiedliche Maßeinheiten* zugrundelegen und bspw. Preise[18] in Euro, DM oder Dollar, Gewichte in Kilogramm oder Gramm oder Temperaturen in Celsius oder Fahrenheit abgespeichert werden. Zudem können Werte in jeweils *unterschiedlichen Genauigkeiten* vorliegen. Hierbei können sowohl unterschiedliche Datentypen vorliegen (Angabe eines Gewichtes über eine der Zeichenketten *heavy*, *middle*, *light* bzw. *ultra-light* oder über einen numerischen Wert zur Bezeichnung des Gewichtes in Tonnen)[19] oder numerische Werte jeweils mit einer unterschiedlichen Anzahl von

[16] Beispiel aus [Conrad 1997, 83].
[17] Beispiel angelehnt an [Conrad 1997, 83].
[18] Preise können zudem mit oder ohne Mehrwertssteuer interpretiert werden (vgl. [Conrad 1997, 46]).
[19] Beispiel aus [Kim, Seo 1991].

Nachkommastellen repräsentiert sein, etwa durch Speicherung eines Gewichtswertes in der Form 0,34567 (kg) bzw. 0,346 (kg)[20]. Conrad zufolge ist bei Vorliegen von Werten unterschiedlicher Genauigkeiten aus verschiedenen Quellen meist nicht klar, wann man Werte als gleich bzw. semantisch äquivalent bezeichnen kann [Kim, Seo 1991] [Conrad 1997, 46f.] [Conrad 2002, Kap. 9].

Datenheterogenität kann ferner durch inkorrekte oder veraltete Daten auftreten. *Veraltete Daten* können in einer Datenbank bspw. dadurch entstehen, dass eine Aktualisierung dort entweder vergessen wurde oder für die lokalen Anwendungen nicht erforderlich ist. Ursachen für *inkorrekte Daten* sind zumeist Tippfehler bei der interaktiven Eingabe oder Programmierfehler in entsprechenden Applikationen. Nach Conrad treten fehlerhafte Abänderungen von Daten in der Praxis relativ häufig auf und sind nur sehr bedingt über Integritätsbedingungen einer Datenbank abzufangen [Kim, Seo 1991] [Conrad 1997, 79ff.] [Conrad 2002, Folie 9-19].

Unter *syntaktischer Heterogenität* schließlich werden von Busse et al. sowohl die unterschiedlichen Eigenschaften der Computer, über die der Zugriff auf die Daten erfolgen soll (technische Heterogenität), sowie der Schnittstellen der jeweiligen individuellen Datenhaltungssysteme (Schnittstellenheterogenität) zusammengefasst. *Technische Heterogenität* ist gegeben, wenn sich die zu adressierenden Daten auf Rechnern mit unterschiedlicher Hardware oder unterschiedlichen Betriebssystemen befinden; ebenso durch Vorliegen verschiedenartiger Anmeldungsverfahren, Sicherheitsstrategien, Kommunikationsprotokolle oder Verbindungsarten für den Zugriff auf diese Systeme. Die Schnittstellen zum Zugriff auf einzelne Datenhaltungssysteme (bspw. individuelle DBMS) sind nicht einheitlich; die hierdurch auftretenden Formen von Heterogenität werden als *Schnittstellenheterogenität* bezeichnet. Hierunter fallen sowohl unterschiedliche Arten von Anfragesprachen (etwa SQL für relationale und OQL für objektorientierte DBMS), unterschiedliche Ausdrucksstärken von Anfragesprachen (etwa wenn bestimmte logische Konstrukte wie Negation oder Disjunktion nicht von allen Systemen unterstützt werden) sowie unterschiedliche Restriktionen individueller Datenhaltungssysteme bezüglich jeweils gültiger Anfragen [Busse et al. 1999].

Die vorgestellten Beispiele stellen nur einen kleinen Ausschnitt der unterschiedlichen Formen von Heterogenität dar. Dennoch wird deutlich, dass Datenheterogenität vielfache Ausprägungen besitzt, die je nach Anwendungskontext und Komplexität mehr oder minder aufwendig behoben werden muss, bevor Daten aus unterschiedlichen Quellen gemeinsam genutzt werden können. [Conrad 1997, 46f.] verweist darauf, dass insbesondere *semantische Heterogenität* schwer zu entdecken und behandeln sei. Ihm zufolge ist diese dann gegeben, wenn „[...] *ein nicht übereinstimmendes Verständnis über die Bedeutung und Interpretation von gleichen oder zusammengehörigen Daten sowie über die beabsichtigte Verwendung dieser Daten*"[21] vorliegt. Dieses Problem sei im allgemeinen noch wenig erforscht, so dass es bisher nur wenige Ansätze zu einer allgemeinen Lösung gebe.

In diesem Zusammenhang sei darauf hingewiesen, dass sich in der Literatur diverse Ansätze zur Klassifikation der verschiedenen Erscheinungsweisen von Heterogenität bzw. der durch diese ausgelösten Konflikte finden, die sich ihrerseits allerdings ebenfalls nicht eben durch Homogenität auszeichnen. So systematisieren [Kim, Seo 1991] die Konflikte, die zwischen relationalen Datenbankschemata auftreten können, und unterscheiden auf der obersten Ebene zwischen *Schemakonflikten* und *Datenkonflikten*, die sie jeweils detailliert weiter untergliedern. [Conrad 1997, 79ff.] legt hingegen vier Klassen von Integrationskonflikten zugrunde (*semantische Konflikte*, *Beschreibungskonflikte*, *Heterogenitätskonflikte*

[20] Beispiel aus [Conrad 2002, Folie 9-20].
[21] [Conrad 1997, 46].

und *strukturelle Konflikte*) und merkt an, dass eine Ausweitung der detaillierten Betrachtungsweise von Kim und Seo auf verschiedene Datenmodelle schnell zu einer Vielzahl von Konfliktarten und zu entsprechender Unübersichtlichkeit führen würde.

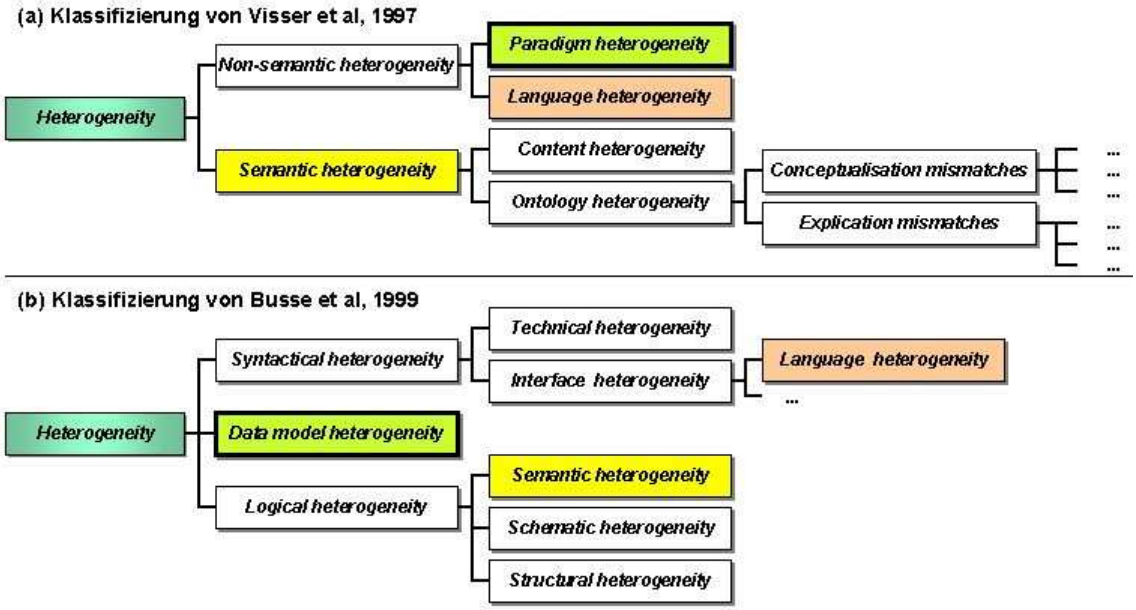

Abb. 2.4 - Heterogene Klassifizierungen von Heterogenität: (a) Aufteilung bei Visser et al, 1997; (b) Aufteilung bei Busse et al., 1999.

Abb. 2.4 veranschaulicht einige der Unterschiede, die zwischen verschiedenen Klassifizierungsansätzen auftreten. So nehmen [Visser et al. 1997] eine detaillierte Klassifizierung ontologischer Datenkonflikte vor und unterscheiden zur Einordnung zwischen vier Formen von Heterogenität, die sie den zwei Hauptformen *non-semantic heterogeneity* sowie *semantic heterogeneity* zuordnen (vgl. Abb. 2.4a). [Busse et al. 1999] wählen in ihrer Klassifizierung hingegen für die oberste Ebene mit *syntactical heterogeneity, data model heterogeneity* sowie *logical heterogeneity* drei Arten von Heterogenität, die teilweise weiter untergliedert werden (vgl. Abb. 2.4b). Ein Vergleich beider Klassifizierungen zeigt, dass zum einen die Begriffe *semantic heterogeneity* sowie *language heterogeneity* jeweils unterschiedlichen Stufen der Hierarchie zugeordnet werden; ferner werden mit *paradigm heterogeneity* bzw. *data model heterogeneity* jeweils andere Begriffe für die Beschreibung des selben Phänomens - der durch unterschiedliche Datenmodelle ausgelösten Heterogenität - gewählt.

2.1.2 Nachteile der Heterogenität am Beispiel von Unternehmensdaten

Die Entwicklung von Ansätzen zur Überwindung von Datenheterogenitäten ist nicht zuletzt vor dem Hintergrund des wachsenden Bedarfs von Unternehmen zu sehen, die vielfältigen dort anfallenden Daten einer integrierten Auswertung zu unterziehen. Die Dynamisierung von Märkten und Wettbewerbssituationen fordert Unternehmen zunehmend präzise und schnelle Entscheidungen ab [Behme 1996, 28]; dabei kommt der effizienten Nutzung von Informations- und Kommunikationssystemen eine hohe strategische Bedeutung zu [Gabriel 1998]. Diese stetig wachsende Bedeutung des Faktors Information für moderne Unternehmen sowie ein permanentes Anschwellen vorhandener Daten bei gleichzeitigem Informationsdefizit [Behme, Mucksch 1996, 9] spiegelt sich in einer Vielzahl von Ansätzen zur Computerunterstützung des Managements[22], die seit den 60er Jahren des 20. Jahrhunderts

[22] Entsprechende Entwicklungen, die hier nicht weiter ausgeführt werden sollen, sind mit Begriffen wie Management Support Systems (MIS), Decision Support Systems (DSS) und Executive Information Systems (EIS) verbunden. Einen Überblick über derartige Ansätze geben beispielsweise [Glu-

hervorgebracht wurden, jedoch bis in die Mitte der 90er Jahre nur mäßigen Erfolg erringen konnten (vgl. bspw. [Schinzer et al. 1999, 5ff.]).

Eine wesentliche Ursache für die Schwierigkeiten, aus den operationalen Daten[23] eines Unternehmens durch Analyse relevante Informationen gewinnen zu können, bildet die Heterogenität dieser Daten. Unternehmensdaten liegen in den seltensten Fällen in einer zentralen Datenbank vor, sondern verteilen sich auf jeweils eigene Systeme[24] bspw. für Logistik, Finanzen, Produktion, Personal oder Rechnungswesen; ein Großunternehmen kann so durchaus 30 bis 40 transaktionsorientierte Systeme im Einsatz haben [Schinzer et al. 1999, 24]. Solche umfassenden Systeme entstehen in der Regel nicht im Hinblick auf eine integrierte Auswertung der in ihnen verarbeiteten Daten, sondern entwickeln sich - getrieben von den operationalen Anforderungen - in Form einer stetig komplexer und heterogener werdenden „*naturally evolving architecture*" aus Daten und Applikationen [Inmon 1996, 1ff.]. Die Durchführung von Analysen auf den operationalen Daten eines Unternehmens steht damit vor einer Vielzahl schwerwiegender Probleme, die kurz umrissen werden sollen.

- **Mangelnde Entscheidungsrelevanz**

Die Eignung operationaler Daten als Basis einer Analyse ist Inmon zufolge durch einen Mangel an Glaubwürdigkeit (*lack of credibility of data*) deutlich limitiert [Inmon 1996, 8ff.]. Probleme sind unter anderem[25] auf die Auswirkung unterschiedlicher Zugriffszeitpunkte auf die jeweiligen Analyseergebnisse zurückzuführen: Da operationale Daten durch die auf ihnen ausgeführten Transaktionen beständig verändert werden, kann bereits eine geringe zeitliche Abweichung beim analytischen Zugriff auf diese zu unterschiedlichen Ergebnissen mit u.U. erheblichen Differenzen führen; ferner wird die Stimmigkeit systemübergreifender Auswertungen durch die heterogene Verwendung betriebswirtschaftlicher Begriffe und Methoden deutlich beeinträchtigt [Gluchowski 1998, 186]. Ebenso kann nicht ausgeschlossen werden, dass verschiedene Extraktionsprogramme aufgrund algorithmischer Unterschiede unterschiedliche Ergebnisse[26] produzieren [Inmon 1996, 11].

- **Gleichbleibend hoher Integrationsaufwand**

Die Heterogenität der operationalen Daten erfordert erheblichen Aufwand, wenn Daten aus allen operativen Systemen lokalisiert, extrahiert und zusammengestellt werden müssen. So veranschlagt Inmon für die Generierung eines unternehmensweiten Reports durchaus einen Zeitrahmen von 3 bis 5 Jahren (!); darüber hinaus besteht die Gefahr, dass ein analog hoher Aufwand aufgrund neuer oder veränderter Anforderungen auch bei der Durchführung zukünftiger Analysen anfällt [Inmon 1996, 12ff.].

- **Mangelnde analytische Eignung**

Neben der Uneinheitlichkeit der operationalen Daten stellt ihre Ausrichtung auf das Tagesgeschäft ein weiteres wesentliches Problem dar. Während operationale Daten meist ausschließlich den aktuellen Informationsstand des Unternehmens spiegeln [Kemper, Finger 1998, 63], erfordert die Analyse von Unternehmensdaten auch die Einbeziehung histori-

---

chowski et al. 1997], [Chamoni, Zeschau 1996] und [Schinzer et al. 1999].

[23] Im Folgenden werden die Begriffe *operativ* und *operational* synonym verwendet.

[24] Auch bei Unternehmen, die Standardanwendungssoftwarepakete wie SAP R/3 verwenden, liegt nach [Schinzer et al. 1999, 24] ein Grossteil der internen Daten auf heterogenen Systemen vor,

[25] [Inmon 1996, 8ff.] nennt neben einer fehlenden zeitlichen Basis und algorithmischen Abweichungen als zusätzliche Faktoren ferner unterschiedliche Extraktionsebenen, das Problem einer nicht rekapitulierbaren Vermengung interner und externer Daten sowie das Fehlen einer gemeinsamen Ausgangsbasis für Analysen.

[26] Kritisch wird dies insbesondere bei mangelnder Dokumentation der Extraktionssoftware; hier kann es geschehen, dass Abweichungen in Ergebnissen kaum mehr nachvollzogen werden können [Gluchowski 1998, 186].

scher Daten und damit eines weitaus größeren Zeithorizontes [Inmon 1996, 15ff.].

- **Grenzen des Zugriffs**

Ein direkter analytischer Zugriff auf operationale Daten setzt zudem eine dialogorientierte, parallele Abfrage aller relevanten operationalen Systeme voraus. Bei vielen Unternehmen befinden sich neben relationalen Datenbankmanagementsystemen allerdings auch heute noch hierarchische oder dateiorientierte Systeme im Einsatz, bei denen eine interaktive Abfrage nicht vorgesehen ist oder nur mit hohem, nicht vertretbarem Aufwand realisiert werden kann. Ferner ist ein akzeptables Antwortverhalten von analytischen Anfragen an operative Systeme, die hierfür nicht optimiert sind, unwahrscheinlich; darüber hinaus können rechenintensive Anfragen die Performance der operativen Systeme beeinträchtigen [Kemper, Finger 1998, 62].

[Inmon 1996, 17ff.] stuft entsprechend die operationalen Systeme eines Unternehmens aufgrund ihres Mangels an Integration sowie ihres geringen Zeithorizontes als inadäquate Infrastruktur für die Bereitstellung der erforderlichen Analysen von Unternehmensdaten ein. Das von ihm entwickelte Konzept des sog. Data Warehouse*,* das die Behebung dieser Defizite anstrebt und zunehmende Verbreitung findet, wird nachfolgend vorgestellt.

2.2 Das Konzept des Data Warehouse

Das Data Warehouse-Konzept zählt zu den Vertretern der sog. materiellen Datenintegration (vgl. Kap. 2.4.1) und dient zu einer redundanten Speicherung von Daten zu Zwecken ihrer nachfolgenden Auswertung. Die Daten werden dabei von ihrer ursprünglichen Verwendung losgelöst und in für Analysen besser geeigneter Form gespeichert. Ein Data Warehouse stellt dabei kein einzelnes Produkt[27] dar [Schinzer et al. 1999, 16], sondern kann sowohl als Konzept wie als Technologie verstanden werden [Holten et al. 2001, 3]. Die nachfolgende Darstellung erfolgt in gewisser Ausführlichkeit, da das Data Warehouse zu den einflussreichen Ansätzen zur Datenintegration zu rechnen ist und häufige Anwendung etwa in Wirtschaft und Verwaltung findet (so finden sich in der Literatur bspw. Berichte über den Einsatz von Data Warehouse-Konzepten bei der Stadt Köln [Christmann et al. 1996], bei der Lufthansa [Bahr 1996], bei BAYER [Kaiser 1996], der Deutsche Post AG [Parteina, Chakrovertty 2001] oder der Douglas Holding [Bertram-Kretzberg 2001]). Ferner lassen sich in diesem Kontext die Vorteile veranschaulichen, die eine redundante Zusammenführung und Aufbereitung von Daten für eine nachfolgende Analyse bieten, ebenso wie die Herausforderungen, die bei der Bereitstellung einer solchen integrierten Datenbasis entstehen können. Schließlich werden Data Warehouse-Methoden auch am Potsdam-Institut für Klimafolgenforschung eingesetzt, um eine integrierte Bereitstellung von multidisziplinären Wissenschaftsdaten aus unterschiedlichen und heterogenen Datenbanken des Institutes zu ermöglichen, so dass die nachfolgenden Ausführungen auch zu einem besseren Verständnis der im Rahmen dieser Arbeit durchgeführten Schnittstellenentwicklung beitragen.

2.2.1 Definition des Data Warehouse

Der Kerngedanke eines *Data Warehouse* (dt. etwa: Daten-Lagerhaus) besteht in der Verbesserung der Informationsqualität eines Unternehmens durch redundante Bereitstellung der analyserelevanten Daten. Die Daten werden aus den heterogenen operativen Systemen extrahiert, aufbereitet, integriert, verdichtet und schließlich für einen rein lesenden

[27] Ein Überblick über Anbieter von Data Warehouse-Werkzeugen findet sich bspw. in [Schinzer 2001], [Schinzer et al. 1999] sowie [Schinzer, Bange 1998]. Ferner finden sich Berichte zu Produkten spezifischer Anbieter; vgl. bspw. zu Produkten von IBM [Pirk, Wolf 2001], von NCR [Bertram 2001], von SAP [Kothen et al. 2001] [Klein 2001] sowie von Oracle [Wittenborg, Rother 2001].

Zugriff streng von den operationalen Systemen getrennt vorgehalten [Schinzer et al. 1999, 15f.]. Die klassische Definition des Data Warehouse stammt von W.H. Inmon, der als der ‚Vater' dieses Konzeptes gilt:

- „*A data warehouse is a subject oriented, integrated, nonvolatile, and time variant collection of data in support of management's decisions.*" [Inmon 1996, 33]

Nachfolgend wird zunächst die Bedeutung der von Inmon geforderten Eigenschaften Themenorientierung (*subject orientation*), Integration (*integration*), Beständigkeit (*nonvolatility*) sowie Zeitbezug (*time variancy*) skizziert.

- **Themenorientierung**

Während die operationalen Daten eines Unternehmens auf effiziente Abwicklung des Tagesgeschäftes ausgelegt und kaum für die Entscheidungsunterstützung geeignet sind [Chamoni, Gluchowski 1998, 14], stellt das Data Warehouse die *thematische* Zusammenführung von Daten mit dem Ziel ihrer Analyse in den Vordergrund. Themen ergeben sich dabei aus den Sichten des Managements auf die jeweilige Unternehmensstruktur sowie die Geschäftsprozesse und können sich bspw. auf Unterabteilungen, Absatzgebiete, Waren- oder Kundenstrukturen beziehen [Holten et al. 2001, 5]. Es ist also keineswegs eine vollständige Kopie sämtlicher in einem Unternehmen vorhandenen Daten angestrebt; vielmehr werden operationale Daten, die nur für die Abwicklung des Tagesgeschäftes benötigt, aber für die Entscheidungsfindung als irrelevant eingestuft werden, *nicht* in das Data Warehouse integriert [Chamoni, Gluchowski 1998, 14].

- **Integration**

Integration bildet nach Inmon den wichtigsten aller Aspekte des Data Warehouse [Inmon 1996, 33f.]. Wesentlicher Beitrag eines Data Warehouse ist also ein konsistenter Datenbestand, auch wenn die eingehenden Datenquellen sehr heterogen sind [Chamoni, Gluchowski 1998, 14]. Die Überwindung der auftretenden Formen von Inkonsistenzen soll dabei bei der Überführung der operationalen Daten in das Data Warehouse vorgenommen werden [Inmon 1996, 34f.] und beinhaltet u.a. einheitliche Namensgebungen, Attributkodierungen oder Maßeinheiten [Holten et al. 2001, 5f.].

- **Beständigkeit**

Das Konzept des Data Warehouse sieht eine kumulative Anreicherung mit Daten vor, die als eine Serie von jeweils zu einem bestimmten Zeitpunkt erstellten Momentaufnahmen aufgefasst werden können [Inmon 1996, 36]. Dabei gilt, dass Daten, die einmal in das Data Warehouse integriert wurden, in der Regel nicht mehr verändert werden [Schinzer et al. 1999, 16]. Anders als die operationalen Daten, die durch die auf ihnen ausgeführten Transaktionen beständigen Veränderungen unterliegen, soll ein Data Warehouse eine Kollektion von Daten darstellen, die langfristig unverändert bereitgehalten werden, um so die vom Management benötigten Analysen über längere Zeiträume hinweg zu ermöglichen [Holten et al. 2001, 6]. Entsprechend fallen die jeweils relevanten Zeithorizonte für operationale Daten und Data Warehouse deutlich unterschiedlich aus. Während der durch ihre Aufgaben vorgegebene Zeitbezug von Daten in operationalen Systemen von Inmon mit etwa 60 bis 90 Tagen veranschlagt wird[28], können die Daten im Data Warehouse einen Zeitraum von 5 bis 10 Jahren umspannen [Inmon 1996, 36]. Solche langen Zeithorizonte sind bspw. für Trendanalysen über „historischen" Daten eines Unternehmens erforderlich [Mucksch 1998, 125].

[28] Nach [Schinzer et al. 1999, 21] enthalten die operativen Datenbestände hingegen selten Zeiträume, die über das letzte Geschäftsjahr hinausgehen.

- **Zeitbezug**

Die Forderung nach explizitem Zeitbezug von Daten im Data Warehouse ist eng mit der für diese geforderten Beständigkeit sowie dem Ziel verknüpft, auf ihnen flexible Analysen durchführen zu können [Holten et al. 2001, 6]. Während operationale Daten abhängig von ihrer Struktur Zeitbezug enthalten *können*, wird dieser für Daten im Data Warehouse *gefordert* [Inmon 1996, 36], so dass die Zeit jeweils als bewertbare Bezugsgröße für die Auswertung der enthaltenen Daten herangezogen werden kann [Schinzer et al. 1999, 14].

2.2.2 Organisationsformen des Data Warehouse

Zum Aufbau eines Data Warehouse stehen verschiedene Organisationsformen zur Verfügung [Eicker 2001, 66ff.] [Schinzer et al. 1999, 20f.]. Wird eine strikt von den operativen Systemen getrennte Datenhaltung aufgebaut, lassen sich zentrale und dezentrale Realisierungen sowie zwei Mischformen unterscheiden (vgl. Abb. 2.5).

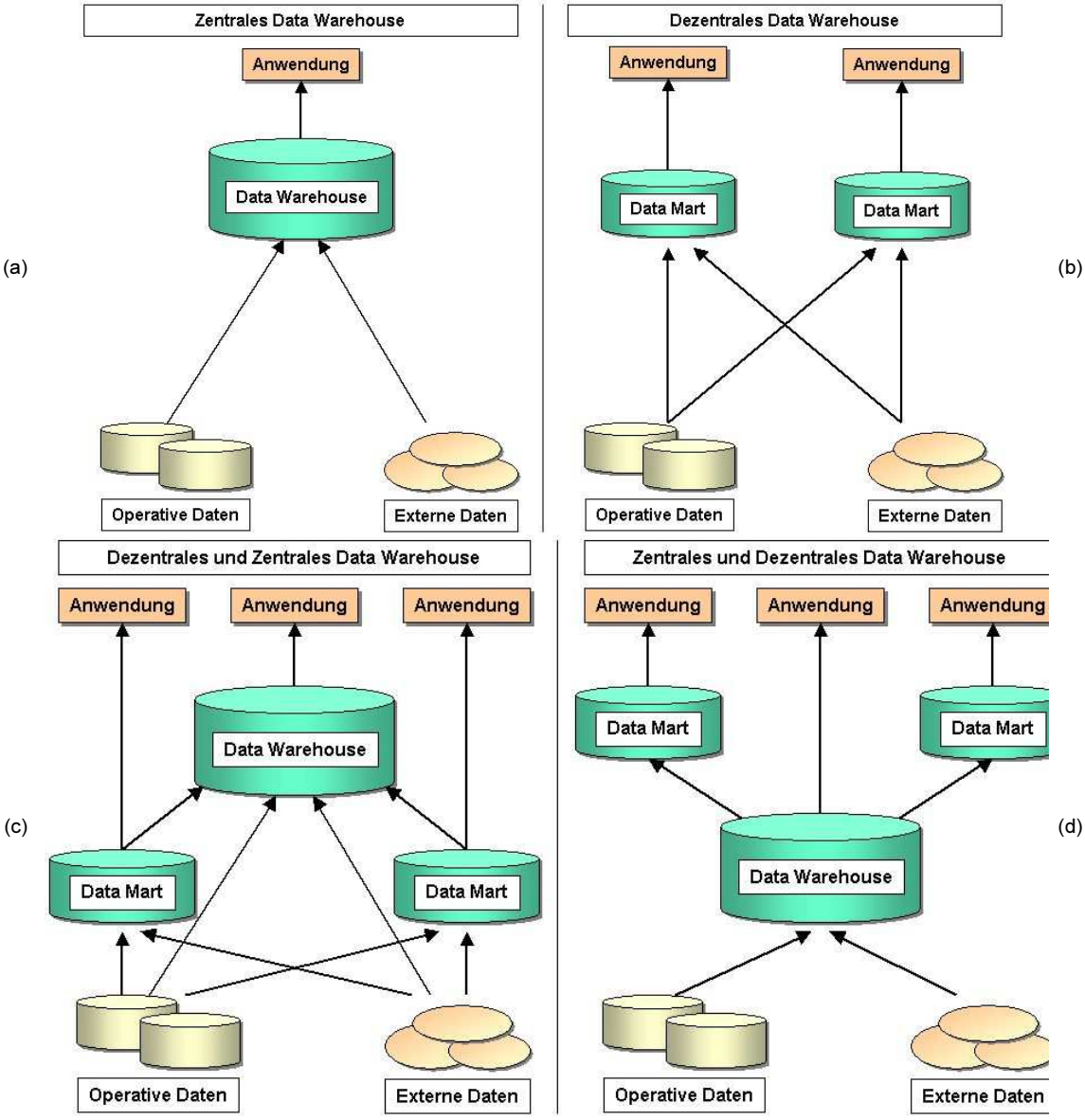

Abb. 2.5 - Unterschiedliche Organisationsformen des Data Warehouse[29]:
(a) zentral; (b) dezentral; (c) und (d) Mischformen.

Ferner können Auswertungswerkzeuge eingesetzt werden, die direkt auf den operativen

[29] Abbildung erstellt in Anlehnung an [Eicker 2001, 67, Abb. 2.4].

Daten arbeiten; in diesem Fall spricht man von einem „virtuellen" Data Warehouse. Nachfolgend werden die unterschiedlichen Organisationsformen kurz gegeneinander abgegrenzt.

Zentral
- Bei einem *zentralen Data Warehouse* (vgl. Abb. 2.5a) werden alle Daten unter der Kontrolle eines Datenbankmanagementsystems gespeichert, wobei bei entsprechender Unterstützung durch dieses auch eine physisch verteilte Haltung möglich ist.

Dezentral
- Bei einem *dezentralen Data Warehouse* (vgl. Abb. 2.5b) werden die Daten nicht in einer zentralen Datenbank gehalten, sondern nach Themen oder Organisationseinheiten in mehreren Einheiten (sog. Data Marts) abgespeichert

Dezentral und zentral
- Bei einem *dezentralen und zentralen Data Warehouse* sind die Data Marts dem Data Warehouse vorgeordnet (vgl. Abb. 2.5c). Die Data Marts werden mit Daten aus den operativen Systemen aufgebaut; das Data Warehouse wird aus den Data Marts befüllt und enthält weitere Daten, die nicht in den Data Marts gespeichert werden.

Zentral und dezentral
- Bei einem *zentralen und dezentralen Data Warehouse* sind hingegen die Data Marts dem Data Warehouse nachgeordnet (vgl. Abb. 2.5d). Das Data Warehouse wird mit Daten aus den operativen Systemen aufgebaut; die Data Marts werden nur über das Data Warehouse befüllt.

Virtuell
- Eine Alternative zum Aufbau einer von den operativen Systemen getrennten Datenbasis besteht im sog. *virtuellen Data Warehouse*. Hier existiert keine gesonderte Verwaltung und Sammlung von Daten; es wird über entsprechende Software ein direkter lesender Zugriff der analytischen Applikationen auf die operativen Datensysteme realisiert.

Durch eine integrierte physische Datenbasis neben den operativen Systemen wird die Basis für die Vorteile bereitgestellt, die mit dem Data Warehouse-Konzept angestrebt werden – einen integrierten, aufbereiteten Datenbestand, die Auswertung historischer Daten, die Möglichkeit, Daten transformiert und aggregiert vorzuhalten, die Möglichkeit zur Nachnutzung bereits erstellter Anfragen und Ergebnisse sowie eine Vermeidung der Belastung operativer Systeme bei der Durchführung von Analysen. Für die Aufspaltung der zentralen Datenbasis in Data Marts können mehrere Gründe vorliegen. Faktoren für eine Verteilung der zentralen Datenbasis bilden bspw. die gleichzeitige Bereitstellung von Daten an mehreren Orten bei örtlich verteilten Unternehmen oder spezielle Datenanforderungen einzelner Fachabteilungen sowie der hohe Zeitaufwand für Auswertungen auf sehr großen Datenmengen. Schließlich kann in großen Unternehmen auch eine schrittweise, von einzelnen Abteilungen ausgehende Realisierung von Data Warehouse-Projekten zum Entstehen verteilter Lösungen führen [Schinzer et al. 1999, 21f.] [Mucksch 1998, 125] [Eicker 2001, 66f.].

Data Marts können dabei als bewusst redundant gehaltenen Kopien von Ausschnitten des Data Warehouse zur Steigerung der Performance angelegt werden. Nach Mucksch können durch gezielte Analyse des Geschäftsprozesses Kerninformationen aus dem Data Warehouse extrahiert und in Data Marts abgelegt werden, die dann mit 20 Prozent der Daten 80 Prozent der Anfragen abdecken [Mucksch 1998, 126]. Bei einer echten Verteilung des Data

Warehouse entsteht hingegen ein deutlich höheren Verwaltungsaufwand sowohl durch die notwendige Synchronisation der Daten wie durch das Entstehen dezentraler Administrationsaufgaben, die angemessenes Fachwissen in den Abteilungen, in denen ein Data Mart aufgebaut wird, erfordern [Schinzer et al. 1999, 22f.]. Hier kommt der Verwendung geeigneter Metadaten besondere Bedeutung zu, da diese zur Sicherung der Konsistenz der verteilt gehaltenen Daten beitragen können [Inmon 1996, 226]. Nach [Eicker 2001, 68] ist der Verzicht auf eine zentrale Datenbasis langfristig zumeist nicht sinnvoll, da so unternehmensweite Analysen deutlich erschwert werden. Er verweist ferner darauf, dass in der Praxis die gewählte Lösung von der Machtverteilung im Unternehmen abhänge, da Abteilungen häufig auf individuelle Lösungen über vorgelagerte Data Marts mit kontrollierter Datenweitergabe an ein zentrales Data Warehouse drängen, um die Kontrolle über den Zugriff auf „ihre" Daten zu behalten. Jeder der skizzierten Ansätze zur Realisierung eines Data Warehouse besitzt spezifische Vor- und Nachteile (vgl. Tab. 2.1), die im Einzelfall allerdings jeweils durch gezielte organisatorische und technische Maßnahmen beeinflusst werden können [Eicker 2001, 69].

Kriterium / gewählter Ansatz für die DW-Basis	**zentral**	**dezentral**	**zentral mit vorgelagerten Data Marts**	**zentral mit nachgelagerten Data Marts**	**virtuell**
allgemeiner Realisierungsaufwand	hoch	hoch, aber verteilt auf mehrere Projekte	sehr hoch, aber verteilt auf mehrere Projekte	sehr hoch, aber verteilt auf mehrere Projekte	gering
Komplexität der DW-Entwicklung	hoch	mittel	mittel	mittel	gering
Unterstützung der sukzessiven Verbreitung des DW-Ansatzes im Unternehmen	gering	sehr gut	sehr gut	sehr gut	sehr gut
Komplexität des DW-Schemas aus Endbenutzersicht	sehr hoch	vergleichsweise gering	stets problemadäquat	stets problemadäquat	abhängig von der Komplexität der zugehörigen operativen Datensysteme
Aufwand für die Entwicklung unternehmensweiter Analysen	vergleichsweise minimal	hoch	vergleichsweise minimal	vergleichsweise minimal	in den meisten Fällen zu hoch
Aufwand für die Entwicklung themen-/ organisationseinheitsspezifischer Analysen	mittel	gering	gering	gering	mittel
Performance unternehmensweiter Analysen	gut	relativ schlecht	gut	gut	sehr schlecht
Performance themen-/ organisationseinheitsspezifischer Analysen	gut	sehr gut	sehr gut	sehr gut	i.a. schlecht
Kontrolle von Organisationseinheiten über die Weitergabe entscheidungsrelevanter Daten	gering	sehr hoch	hoch	gering	sehr hoch

Tab. 2.1 - Vor- und Nachteile der verschiedenen Organisationsformen des Data Warehouse[30].

Nach [Eicker 2001, 68] wird in der Praxis oft vom Aufbau eines getrennten Data Warehouse Abstand genommen, da der Aufwand für Entwicklung und Betrieb als zu hoch eingestuft wird. Bei einer „virtuellen" Umsetzung ohne getrennte physische Datenbasis bleiben so gut wie alle der unter 2.1.2 beschriebenen Nachteile, die bei analytischen Anfragen auf operativen Daten auftreten, bestehen; ein virtuelles Data Warehouse kann jedoch aufgrund der Möglichkeit einer vergleichsweise schnellen und kostengünstigen Umsetzung eine Alternative für einen ersten Einstieg in eine Data Warehouse-Umsetzung oder für relativ

[30] Tabelle übernommen aus [Eicker 2001, 69, Tabelle 2].

kleine Zielgruppen, die auf sehr aktuelle Informationen angewiesen sind, darstellen [Schinzer et al. 1999, 20f.].

2.2.3 Datenhaltung im Data Warehouse

Obwohl in der Literatur keine völlige Einigkeit über die Komponenten besteht, die zu einem Data Warehouse gerechnet werden, lassen sich mit der eigentlichen Data Warehouse-Datenbasis, dem Archivierungssystem, der Metadatenbank sowie den Transformationsprogrammen zur Dateneinspeisung (vgl. hierzu Kap. 2.2.4) vier elementare Komponenten eines idealtypischen Data Warehouse [Mucksch 1998, 127ff.] ausmachen (vgl. Abb. 2.6). Die Datenbasis eines Data Warehouse enthält aktuelle und historische Daten der relevanten Unternehmensbereiche in unterschiedlichen Verdichtungsstufen [Mucksch 1998, 128]; das hier anfallende Datenvolumen kann Größenordnungen im Terabyte-Bereich[31] erreichen [Holten et al. 2001, 10]. Das Datenbanksystem eines Data Warehouse bildet damit einen seiner zentralen Bestandteile.

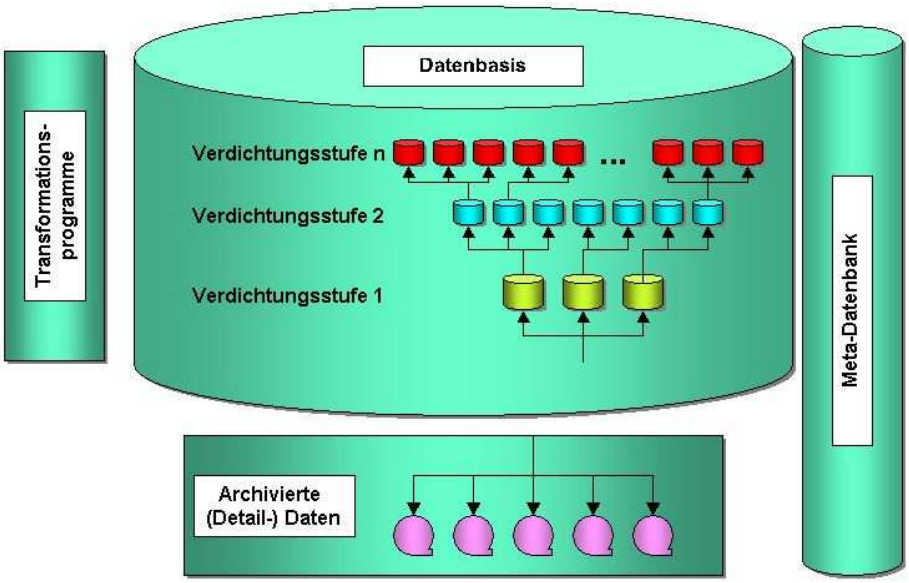

Abb. 2.6 - Schematischer Aufbau eines Data Warehouse[32].

Zur Speicherung der Daten sind mit gewissen Einschränkungen grundsätzlich alle relationalen Datenbankmanagementsysteme[33], die auch im operativen Bereich Verwendung finden, einsetzbar [Mucksch 1998, 130]; diese bilden nach Holten et al. auch die in der Praxis am häufigsten verwendete Technologie[34] für die physische Realisierung der Data Warehouse-Datenbank [Holten et al. 2001, 10]. Da für die Auswertung der Daten oft eine flexible multidimensionale Betrachtung angestrebt wird (vgl. Kap. 2.3, OLAP), kommen in diesem Bereich jedoch auch spezielle multidimensionale Datenbanksysteme zum Einsatz (vgl. bspw. [Schinzer et al. 1999, 33]).

---

[31] Bei der Deutschen Post AG fallen etwa allein im Bereich Paketversand jährlich etwa 1,5 Terabyte an Liefer- und Logistikdaten an [Parteina, Chakrovertty 2001, 270]. Das Data Warehouse von Walmart wird von [Fayyad 1998] - mit einem Datenvolumen von 11 Terabyte für 1996 - als „*the biggest data warehouse in the world*" eingestuft; nach einer jüngeren Quelle enthält es rund 25 Terabyte an Bestandsdaten [Holten et al. 2001, 10].

[32] Abbildung erstellt in Anlehnung an [Mucksch 1998, 128, Abb. 2].

[33] Um eine Verbesserung der Anfrageperformance durch eine geringere Zahl der für Auswertungen erforderlichen Datenbankzugriffe zu erreichen, wird dabei eine redundante Speicherung durch Denormalisierung bewusst in Kauf genommen [Mucksch 1998, 130] (vgl. die in Kap. 2.3.4 vorgestellten Modellierungsschemata).

[34] Zum Einsatz objektorientierter Datenbanksysteme für Data Warehouses vgl. bspw. [Ohlendorf 1996].

- **Granularität**

Eine entscheidende Rolle beim Design einer Data Warehouse-Datenbank kommt der Auswahl geeigneter Formen von Granularität und Partitionierung zu [Inmon 1996, 45ff.]. Beide Aspekte wirken auf das Antwortzeitverhalten und stellen somit einen wesentlichen Faktor für Nutzerzufriedenheit und Akzeptanz eines Data Warehouse dar [Keppel et al. 2001, 103]. Der Begriff *Granularität* beschreibt dabei den Grad der Verdichtung, mit dem Daten im Data Warehouse abgespeichert werden. Dabei ist Granularität umgekehrt proportional zum Detailgrad der Daten – je stärker die Daten bspw. durch Aggregation verdichtet werden, desto weniger Details sie also enthalten, desto höher ist ihre Granularität. Von der gewählten Granularität hängen in direkter Weise sowohl die Menge[35] der im Data Warehouse zu speichernden Daten ab, als auch die Art der Fragen, die mit diesen Daten beantwortet werden können (vgl. Abb. 2.7); Inmon stuft sie entsprechend als wichtigsten Einzelaspekt im Design eines Data Warehouse ein [Inmon 1996, 45].

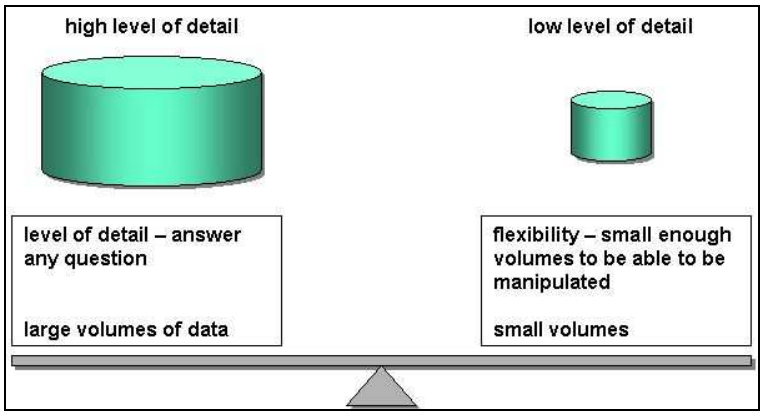

Abb. 2.7 - Die Granularität eines Data Warehouse als Balanceakt zwischen verfügbarem Detailgrad und Handhabbarkeit[36].

Da in der Praxis in der Regel ein Kompromiss zwischen verarbeitbarer Datenmenge und erforderlichem Detailgrad von Anfragen an das Data Warehouse gefunden werden muss, bietet sich die Realisierung mehrerer Verdichtungsstufen an [Inmon 1996, 51ff.]. So können bspw. dynamisch unterschiedliche Detaillevel abhängig von der Zeitnähe der Daten erstellt werden, etwa durch detailliertes Vorhalten von Daten des aktuellen und des vergangenen Monats. Nach Ablauf eines Monats können dann die Daten des jeweils älteren Monats detailliert archiviert und im Data Warehouse nur noch in verdichteter Form bereitgestellt werden [Mucksch 1998, 129]. Die Wahl der jeweils geeigneten Granularisierung einer konkreten Data Warehouse-Implementierung hängt dabei in hohem Maße von den Anforderungen der Anwender ab und sollte daher nicht vollständig a priori definiert, sondern in einem inkrementellen Prozess unter Einbeziehung der Nutzer über die Zeit entwickelt werden [Inmon 1996, 159].

- **Partitionierung**

Die geeignete *Partitionierung* des Data Warehouse-Datenbestandes, also seine Aufteilung in kleinere, physisch selbständige Einheiten mit redundanzfreien Datenbeständen, kann die Verarbeitungseffizienz entscheidend prägen, muss aber unter Abwägung des möglicherweise entstehenden erhöhten Aufwandes[37] entworfen werden [Mucksch 1998, 129]. Eine

[35] Die Granularität besitzt damit direkte Auswirkungen auf Speicherbedarf, erreichbare Verarbeitungsgeschwindigkeit und Flexibilität des Data Warehouse [Mucksch 1998, 128].
[36] Abbildung erstellt in Anlehnung an [Inmon 1996, 50, Fig. 2.14].
[37] Kriterien sind hier gesteigerte Anforderungen an die Datenmodellierung, bei der Übernahme der Daten aus den operativen Systemen oder bei Auswertungen, die auf mehrere Partitionen zugreifen müssen (vgl. [Mucksch 1998, 129]).

Abspeicherung von Daten in zu großen Einheiten wirkt sich negativ auf die Flexibilität[38] eines Data Warehouse aus; entsprechend stellt sich nach [Inmon 1996, 57ff.] nicht die Frage, *ob* eine Data Warehouse-Datenbank partitioniert werden soll, sondern *wie* eine geeignete Aufteilung erfolgen soll. Die Aufteilung selber kann sich dabei an einer Vielzahl unterschiedlicher Kriterien orientieren. Im betriebswirtschaftlichen Kontext kann etwa zwischen *horizontaler Partitionierung* (Aufteilung der Unternehmensdaten auf die verschiedenen Tochter- und das Mutterunternehmen bzw. auf verschiedene Zeiträume) und *vertikaler Partitionierung* (Aufteilung anhand unternehmensbestimmender Sachverhalte) unterschieden werden [Mucksch 1998, 129]. Inmon zufolge kommt Partitionierungen entlang des Zeitbezuges im Data Warehouse dabei so gut wie immer eine zentrale Bedeutung zu [Inmon 1996, 59].

- **Archivierungssystem**

Das Archivierungssystem eines Data Warehouse dient sowohl der Datensicherung wie der Datenarchivierung. Die *Sicherung* des Datenbestandes, für die alle aus den operativen Systemen bekannten Techniken einsetzbar sind, bildet die Basis für eine Wiederherstellung des Data Warehouse, wie sie bspw. nach Programm- oder Systemfehlern erforderlich sein kann. Hierzu sollten zumindest die Daten der untersten Verdichtungsstufe gesichert werden; wenn eine schnelle Wiederherstellung im Vordergrund steht, sollten zudem auch sämtliche Verdichtungsstufen archiviert werden. Die eigentliche *Datenarchivierung* dient der Reduzierung des Datenvolumens im Data Warehouse durch Auslagerung von Daten und damit der Performancesteigerung; so können bspw. Daten der untersten Verdichtungsstufen ausgelagert und offline gehalten werden [Mucksch 1998, 133].

- **Metadaten**

Metadaten dienen zur Dokumentation der Zusammenhänge im Data Warehouse und bilden die Grundlage für den Austausch von Daten zwischen seinen einzelnen Komponenten sowie zur effizienten Nutzung durch Applikationen und Anwender [Schinzer et al. 1999, 25ff.]. So benötigen Endanwender, die zunehmend Aufgaben wahrnehmen, die früher nur von EDV-Spezialisten ausgeführt wurden [Behme, Mucksch 1996, 10ff.], nicht nur Zugang zu den Daten eines Data Warehouse, sondern darüber hinausgehende Informationen, um deren Relevanz für ihre jeweiligen Aufgaben beurteilen zu können [Mucksch 1998, 134]. Zu den in der Metadatenbank eines Data Warehouse gespeicherten Daten zählen unter anderem Metadaten über das dem Data Warehouse zugrundeliegende Datenmodell, semantische und DV-technische Beschreibungen aller gespeicherter Daten, die Herkunft der Daten, Informationen über den gesamten Transformationsprozess sowie Informationen über alle Verdichtungsstufen. Sinnvolle Erweiterungen können bspw. ein Thesaurus mit Synonymen und ein alphabetisch geordnetes Glossar aller verwendeten Bezeichnungen und Abkürzungen darstellen [Mucksch 1998, 135f.].

Obwohl die Metadatenbank eines Data Warehouse ein Schlüsselelement für seine Akzeptanz durch die Entscheidungsträger [Mucksch 1998, 136] darstellt, besteht in der Praxis oft erheblicher Aufwand für die Erzeugung und Bereitstellung von Metadaten. Nach Schinzer et al. erfordert insbesondere die Realisierung von Data Warehouse-Lösungen mit Komponenten verschiedener Hersteller Programmierarbeiten zum Austausch der jeweiligen herstellerspezifischen Metadaten mit sehr hohem Zeit- und Kostenaufwand[39] [Schinzer et al. 1999, 26].

---

[38] Hiervon sind unter anderem Möglichkeiten zur flexiblen Reorganisation, Indizierung oder Datensicherung betroffen [Inmon 1996, 59].

[39] In der Folge beschränken sich viele Anwender auf die Werkzeuge eines einzigen Anbieters, auch wenn für einzelne Aufgaben geeignetere Werkzeuge anderer Hersteller verfügbar sind [Schinzer et al. 1999, 26].

2.2.4 Datenversorgung

Die Qualität der über das Data Warehouse bereitgestellten Datenbasis ist von wesentlicher Bedeutung für die Akzeptanz durch die Anwender und damit für den Erfolg eines solchen Projektes [Keppel et al. 2001, 103f.]. Die Transformation operativer Daten in eine zur Analyse geeignete Form bildet dabei den Engpass (Bottleneck) bei der Erstellung und Nutzung eines Data Warehouse [Müller 1998, 81]; weitere Herausforderungen bestehen in der Etablierung geeigneter Mechanismen zur laufenden Aktualisierung des Data Warehouse sowie in der Einbeziehung externer Daten. Aufgrund der vielfältigen Heterogenitäten der operativen Datenbasis entfällt der entschieden größte Anteil des Aufwandes bei der Realisierung eines Data Warehouse mit bis zu 80 Prozent auf Design und Realisierung der Schnittstelle zur Dateneinspeisung aus den operativen Systemen [Inmon 1996, 281]. Nach [Schinzer et al. 1999, 28] ist der hierfür erforderliche Zeit- und Kostenaufwand von Auftraggebern oft nicht nachvollziehbar und kann zu schweren Projektkrisen führen, die sogar mit dem Abbruch der Bemühungen enden können, so dass vermehrt unterstützende Werkzeuge entwickelt werden. Die Entwicklung von Standardlösungen für die Versorgung des Data Warehouse mit Daten allerdings erscheint nur begrenzt erfolgversprechend, da diese die vielen individuellen Probleme, insbesondere bei der Erkennung und Auflösung semantischer Heterogenitäten, kaum hinreichend berücksichtigen können [Müller 1998, 99].

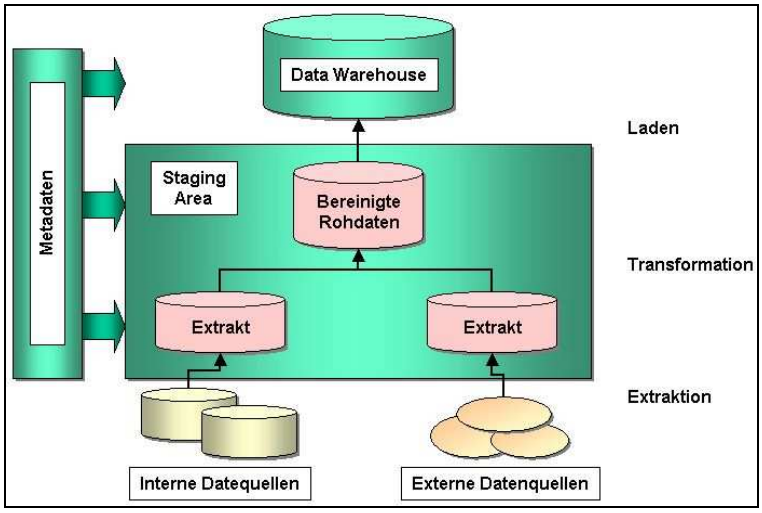

Abb. 2.8 - Überblick über den ETL-Prozess[40].

Die zur Einspeisung von Daten aus den operativen Systemen in das Data Warehouse durchzuführenden Schritte werden oft mit Akronym ETL (für Extraktion, Transformation und Laden, *Extraction-Transformation-Loading,* vgl. Abb. 2.8) beschrieben (vgl. bspw. [Holten et al. 2001]). [Kemper, Finger 1998] fassen die Extraktion von Daten aus den operativen Systemen sowie ihre Aufbereitung unter dem Begriff *Transformation* zusammen und unterscheiden zwischen den Subprozessen Filterung, Harmonisierung, Verdichtung und Anreicherung, die nachfolgend beschrieben werden.

- **Filterung**

Der Prozess der *Filterung* umfasst die Extraktion der relevanten Daten aus den operativen Systemen sowie die Bereinigung syntaktischer und inhaltlicher Defekte. Die *Extraktion* der Basisdaten aus den operativen Systemen setzt zunächst einen umfassenden Prozess zur Klärung der Zugriffsmöglichkeiten sowie die Erarbeitung eventuell erforderlicher, neuer Schnittstellen voraus. Ferner können durch Analyse der in der operativen Datenbanken realisierten Konzepte Rückschlüsse auf die Qualität der Datenquellen gewonnen werden

[40] Abbildung erstellt in Anlehnung an [Schinzer et al. 1999, 30, Abb. 3/2].

[Kemper, Finger 1998, 65ff.]. Die extrahierten Daten werden für die nachfolgend erforderlichen Prozesse der Fehlerbereinigung, Verdichtung und Anreicherung in einer sog. *Staging Area* temporär zwischengespeichert [Schinzer et al. 1999, 29]. An die Extraktion der relevanten operativen Daten schließt sich ihre *Bereinigung* von syntaktischen und inhaltlichen Fehlern an[41]. Einige Arten von Datenfehlern können während der Extraktion automatisch erkannt werden, erfordern jedoch unterschiedliche Arten der Behandlung. Eine automatische Korrektur kann während des Extraktionsvorganges bei einigen systematischen Fehlern - wie nichtinterpretierbaren Steuerzeichen oder uneinheitlichen Zeichensätzen - erfolgen[42]. Anomalien, die zwar automatisch erkannt, aber nicht mittels einer bekannten Regel korrigierbar sind - wie das Auftreten von Ausreißerwerten - werden in einer speziellen Log-Datei dokumentiert; ihre Auswertung und entsprechende Datenkorrekturen sind dann durch Fachspezialisten manuell durchzuführen [Kemper, Finger 1998, 68f.]. Die Erkennung und Korrektur von Fehlern mit Einzelfallcharakter hingegen kann nur manuell durch Analysen von Fachspezialisten geleistet werden [Kemper, Finger 1998, 77, Anm. 2].

- **Harmonisierung**

Der sich an die Filterung anschließende Prozess der *Harmonisierung* dient der betriebswirtschaftlichen Abstimmung der Daten und beinhaltet ihre themenbezogene Gruppierung bspw. nach Kunden, Produkten oder Organisationseinheiten. Zuvor erfordert allerdings die notwendige Beseitigung der vorhandenen Heterogenitäten unterschiedlich komplexe Prozesse. So können unterschiedliche Codierungen, Homonyme und Synonyme meist verhältnismäßig einfach korrigiert werden; die Beseitigung vorhandener Schlüsseldisharmonien ist hingegen in der Praxis oft nicht ohne weiteres lösbar[43]. Einen wichtigen Aspekt der Harmonisierung bildet zudem die Vereinheitlichung betriebswirtschaftlicher Begriffe und deren datentechnische Abgrenzung. Da in diesem Bereich innerhalb eines Unternehmens oft durchaus unterschiedliche Auffassungen und Definitionen auftreten können, ist zur Klärung ein kontroverser Diskussionsprozess mit den Beteiligten erforderlich [Kemper, Finger 1998, 69ff.].

- **Verdichtung**

Nach Abschluss von Filterung und Harmonisierung liegt ein bereinigter und konsistenter Datenbestand auf der niedrigsten Granularitätsebene vor, der nun geeigneten *Verdichtungsprozessen* bspw. entlang der Zeitachse, nach Produktgruppen, Regionen etc. unterzogen wird. Da typische analytische Fragestellungen nicht das kleinste Granularitäts-Niveau betreffen, ist es aus Performancegründen sinnvoll, benötigte Aggregationen im Vorfeld zu berechnen, um zeitintensive Berechnungen während der Anfragen zu vermeiden [Kemper, Finger 1998, 72] [Schinzer et al. 1999, 30]. Zudem sind bestimmte betriebswirtschaftliche Kennzahlen erst berechenbar, wenn Aufsummierungen stattgefunden haben[44]. [Kemper, Finger 1998, 73] weisen darauf hin, dass diese Datenvorbereitung im Hinblick auf erwartete Fragestellungen eine gewisse Abkehr von applikationsneutraler Datenmodellie-

[41] Ein positiver Nebeneffekt kann eine Verbesserung der Datenbasis in den operativen Systemen durch anschließende Durchführung entsprechender Korrekturen auf dieser sein [Kemper, Finger 1998, 69].

[42] Dies kann unter Verwendung entsprechender Abbildungstabellen (Mapping-Tabellen) geschehen [Kemper, Finger 1998, 69].

[43] So erfordert etwa die Einführung eines fehlenden gemeinsamen Primärschlüssels zumeist ein grundsätzliches Re-Design der operativen Systeme. Manuelle Lösungen können bspw. in der Zusammenführung der gefilterten Daten über eine Schlüsselgenerierung durch den Einsatz von Mapping-Tabellen für bspw. die hundert relevantesten Einträge bestehen, sind jedoch für Massendaten nicht geeignet [Kemper, Finger 1998, 69ff.].

[44] [Kemper, Finger 1998, 72] nennen als Beispiel die Berechnung einer Ist-Plan-Abweichung für den Verkauf, die Daten nicht in detaillierter Form von Einzelartikeln auf Tagesbasis, sondern auf dem Niveau von Monaten und Produktgruppen benötigt.

rung bedeutet und als verstärkte Hinwendung zu einer Applikationsorientierung der Daten, bei der die Anwendungslogik teilweise in den Datenstrukturen implementiert wird, gewertet werden kann.

- **Anreicherung**

Den abschließenden Prozess der Transformation bildet die *Anreicherung* der harmonisierten Daten durch die Bildung und Speicherung betriebswirtschaftlicher Kenngrößen wie der Plan/Ist-Abweichung oder des Deckungsbetrags. Die Erweiterung des Data Warehouse-Bestandes um solche vorberechneten Werte trägt neben einer besseren Abfrageperformance zur konsistenten Bereitstellung von Kenngrößen sowie zur Etablierung eines abgestimmten betriebswirtschaftlichen Instrumentariums bei [Kemper, Finger 1998, 74f.] [Schinzer et al. 1999, 30].

- **Aktualisierung des Datenbestandes**

Bei der Datenbeschaffung für ein Data Warehouse ist zwischen der initialen Befüllung und seiner laufenden Aktualisierung zu unterscheiden. Dabei stellt die *erstmalige Befüllung* mit Daten, möglicherweise unter Einbeziehung vorhandener, offline gehaltener Archivdaten[45], aufgrund ihres einmaligen Charakters das kleinere Problem dar; hier können wenig automatisierte Verfahren sowie eine kurzzeitige Beeinträchtigung der operativen Systeme in Kauf genommen werden. Die *laufende Aktualisierung* der Datenbestände hingegen erfordert aufgrund ihres wiederholten Auftretens automatisierte Prozesse, die zugleich die operativen Systeme nicht behindern dürfen und ist entsprechend als problematischer einzustufen [Inmon 1996, 76ff.] [Müller 1998, 82ff.]. Eine Möglichkeit zum Datenexport bildet dabei das Erstellen einer vollständigen Kopie aller Datensätze (Bulk Copy) aus den operativen Systemen und ihre anschließende Weiterverarbeitung. Diese Vorgehensweise kann bei der initialen Befüllung angewendet werden und eignet sich dann für Aktualisierungsläufe, wenn die Ressourcenbelastung niedrig genug ist, um die Entwicklung ausgefeilter Methoden als nicht zweckmäßig erscheinen zu lassen. Die Alternative bildet die gezielte Extraktion neuer oder aktualisierter Datensätze, der allerdings die Identifikation dieser Datensätze vorausgehen muss [Müller 1998, 86f.]. Ziel einer Aktualisierung ist die Anreicherung des Data Warehouse um relevante operative Daten, für die sich seit der letzten Übernahme Veränderungen ergeben haben. Entsprechende Konzepte können dabei unterschieden werden in sog. Push-Techniken, bei denen die operativen Systeme geänderte Daten in das Data Warehouse übertragen, in sog. Pull-Techniken, bei denen das Data Warehouse selber die Daten holt, und in die Verwendung separater Komponenten (Middleware) zur Steuerung der Datengewinnung [Müller 1998, 85].

Für die Identifikation derjenigen Datensätze, die für eine Aktualisierung in Frage kommen, stehen mehrere Verfahren zur Verfügung, von denen die meisten allerdings mit deutlichen Nachteilen einhergehen, so dass sich bei realen Implementierungen durchaus Mischformen ergeben [Müller 1998, 88]. So besteht der zeitstempelgesteuerte Ansatz in einer Auswertung vorhandener Zeitfelder in den operativen Daten. Falls die operativen Datenmodelle jedoch zunächst um entsprechende Felder erweitert werden müssen, besteht ein erhebliches Risiko, dass tiefgreifende Änderungen erforderlich werden, die sich zu einem fehler- und kostenträchtiges Projekt auswachsen und auf entsprechende Akzeptanzprobleme stoßen können [Müller 1998, 88f.] [Inmon 1996, 77]. Eine weitere Alternative bildet die Abänderung der auf den operativen Daten arbeitenden Applikationen, so dass diese Extraktionsdaten direkt an das Data Warehouse übergeben. Auch dieser Ansatz bringt erhebliche

[45] Nach [Inmon 1996, 77] schätzen allerdings viele Organisationen die Nutzung alter Daten als nicht kosteneffektiv ein und verzichten entsprechend auf die Integration ihrer Archivdaten in das Data Warehouse.

Nachteile mit sich: Der Aufwand zur Änderung bestehender Applikationen ist als sehr hoch einzuschätzen; ältere Applikationen sind zudem oft nicht sehr stabil und schlecht wartbar und können überdies nicht immer ist eine verändert oder erweitert werden. Ein weiteres wesentliches Problem bildet die Konsistenzsicherung der so entstehenden dezentralen Verarbeitungslogik [Müller 1998, 90f.] [Inmon 1996, 78].

Ebenfalls problematisch erscheint eine Verwendung systemeigener Log-Dateien wie DBMS-Logfiles oder Audit trails mit dem Ziel**,** durch ihre Analyse veränderte Datensätze zu entdecken. DBMS-Logfiles dokumentieren alle Änderungen an den Datenbeständen eines DBMS, um bspw. die automatische Wiederherstellung eines konsistenten Zustands nach fehlgeschlagenen Transaktionen (Recovery) zu ermöglichen. Allerdings sind sie speziell für das jeweilige Recovery-Werkzeug eines DBMS ausgelegt, enthalten viele weitere Informationen und sind insbesondere proprietär, so dass eine entsprechende Auswertung bei einem Wechsel des DBMS neu definiert werden muss. Audit trails dienen zur Protokollierung spezieller, vom Datenbankadministrator festgelegter Ereignisse. Ihre Auswertung stößt auf vergleichbare Probleme wie die von DBMS-Logfiles und ist auf die vordefinierten Auditing-Funktionen beschränkt [Müller 1998, 94] [Inmon 1996, 77f.].

Ein kaum geeigneter Ansatz zur Identifikation veränderter operativer Daten besteht im Vergleich von Datenbankschnappschüssen (Snap Shots). Ein Schnappschuss ist eine Kopie von Datenbank-Tabellen, die zu einem bestimmten Zeitpunkt angelegt wird und nur für Lesezugriffe zur Verfügung steht; entsprechend können durch den Vergleich von zwei zu unterschiedlichen Zeitpunkten angefertigten Schnappschüssen die in der Zwischenzeit ausgeführten Transaktionen entdeckt werden. Die erheblichen Nachteile dieses Ansatzes liegen zum einen in seinem hohen Ressourcenverbrauch, da das Durchsuchen des gesamten Datenbestandes erforderlich ist, um die relevanten Datensätze aufzufinden; zudem müssen hierzu mindestens zwei Kopien auf Speichermedien mit schnellem Zugriff gehalten werden. Insbesondere müssen diejenigen Tabellen der operativen Daten, die durch referentielle Integritätsregeln verknüpft sind, während der Erstellung von Snap Shots mit einer Schreibsperre versehen werden, was *de facto* einer - inakzeptablen - Stillegung des operativen Tagesgeschäfts gleichkommt. Inmon merkt an, dass dieser Ansatz primär dazu geeignet sei, andere Personen davon zu überzeugen, dass es einen besseren Weg geben müsse [Müller 1998, 95] [Inmon 1996, 78f.].

Vor dem Hintergrund der Nachteile der beschriebenen Ansätze scheint eine gezielte Definition von Tabellen oder Dateien, die für alle Änderungen an den für das Data Warehouse relevanten operativen Daten einen Protokolldatensatz aufnehmen, den am ehesten geeignete Ansatz zur Identifikation relevanter Aktualisierungsdaten darzustellen. Zu berücksichtigen ist allerdings, dass dieser Ansatz das Vorliegen der operativen Datenbestände in modernen DBMS erfordert und einen erhöhten Ressourcenverbrauch bei den operativen Transaktionen mit sich bringen kann [Müller 1998, 91ff.].

- **Aktualisierungszyklen**

Die Konzeption des Data Warehouse sieht *nicht* vor, dass sich jede Änderung in den operativen Daten sofort in einer Aktualisierung des Data Warehouse niederschlagen soll. Da die Anwendungsschwerpunkte von Analysen über längere Zeiträume gebildet werden, reichen hierfür oft Daten aus, die zuletzt vor Stunden, Tage oder Wochen aktualisiert wurden. Auf diesem Weg kann zudem ausgeschlossen werden, dass zwei hintereinander generierte Anfragen an das Data Warehouse zu unterschiedlichen Ergebnissen führen [Chamoni, Gluchowski 1998, 14]. Eine möglichst zeitnahe Nachführung des Data Warehouse bindet durch Prozessorlast und Netzverkehr erhebliche Ressourcen, verringert damit unter Umständen die Leistungsfähigkeit der operativen Systeme und verursacht Kosten; so dass üblicherweise Zeiten geringer Auslastung wie Nächte oder Wochenenden gewählt werden

[Müller 1998, 84f.]. Zu beachten ist allerdings, dass bspw. Sicherungs- und Archivierungsläufe ebenfalls in diesen Schwachlastzeiten stattfinden und nicht behindert werden dürfen [Müller 1998, 101, Anm. 5]. Inmon empfiehlt eine Verzögerung von mindestens 24 Stunden, um der Versuchung zu begegnen, das Data Warehouse in eine operationale Umgebung zu verwandeln [Inmon 1996, 281f.]. Nach [Schinzer et al. 1999, 37] betrachten die meisten Anwender Aktualisierungszyklen von einem oder mehreren Tagen als ausreichend; die Autoren sehen jedoch einen Trend zu höherer Aktualität der Data Warehouse-Daten.

2.2.5 Herausforderungen

Abschließend soll anhand fehlender Standardisierung und erforderlichem Aufwand, der Einbeziehung externer Daten sowie Datenschutzaspekten auf einige der bestehenden Herausforderungen verwiesen werden, die mit dem Konzept des Data Warehouse verbunden sind.

- **Mangelnde Standardisierung und Aufwand**

Chamoni und Gluchowski verweisen darauf, dass noch lange nicht alle Fragen im Data Warehouse-Umfeld beantwortet sind. Als besonders kritisch für die Investitionssicherheit werten sie den Umstand, dass nicht alle benötigten Standards, bspw. für Schnittstellen und Speichertechnologie, definiert sind [Chamoni, Gluchowski 1998, 17f.]. Müller stuft die Entwicklung von Standardlösungen zur Versorgung des Data Warehouse mit Daten als derzeit nur begrenzt erfolgversprechend ein, da die vielen individuellen Probleme, die durch die bestehenden operativen Systeme verursacht werden, nicht berücksichtigt werden können; er folgert, dass insbesondere für das Entdecken und Auflösen semantischer Heterogenitäten jeweils einzelfallbezogene Entwicklungsarbeit erforderlich bleiben wird [Müller 1998, 99]. Ferner entsteht durch den Aufbau eines Data Warehouse ein zusätzliches Datenhaltungssystem neben den vorhandenen operativen Systemen, das neben hohen Investitionskosten für Hard- und Software auch einen nicht geringen Personalaufwand erfordert; für die Systemintegration werden hier 10 Prozent, für die Systemadministration 38 Prozent des Gesamtbudgets eines zentralen Data Warehouse veranschlagt [Schinzer et al. 1999, 22].

- **Einbeziehung externer Daten**

Neben den internen operativen Daten kann potentiell eine Vielzahl externer Daten herangezogen werden, um die Entscheidungsqualität eines Unternehmens zu verbessern; hierzu zählen bspw. Daten über Produkte, Service und Preise der Konkurrenz, Daten über Währungsschwankungen, Aktienkurse und Rohstoffpreise, demographische Daten etc. Der von Unternehmen durch Einbeziehung solcher externer Daten erhoffte Nutzen besteht dabei sowohl im rechtzeitigen Erkennen von Chancen wie der Vermeidung potentieller Risiken [Schinzer et al. 1999, 24]. Behme und Kruppa sehen das World Wide Web als primäre Datenquelle für externe Informationen; das systematisches Auffinden von Inhalten und Hinzufügen zum Data Warehouse wird dabei als Web Farming bezeichnet. Die Autoren verweisen allerdings darauf, dass die im World Wide Web verfügbaren Daten in einer großen Anzahl überaus heterogener Datenquellen vorliegen; ein Data Warehouse kann hier Metadaten über externe Web-Quellen bereitstellen und so als Mittler zu diesen fungieren [Behme, Kruppa 1998]. [Schinzer et al. 1999, 25] zufolge zählen gebührenpflichtiger Online-Datenbanken zu den wichtigsten externen Datenquellen, während „Internetdaten" bisher nur mit hohem manuellem Aufwand erschließbar seien. Nach Inmon sind bei der Einbeziehung externer Daten mehrere Herausforderungen zu berücksichtigen. Zum einen können relevante Daten aus einer Vielzahl von Quellen einzubeziehen sein; ferner können im Gegensatz zu internen Daten weniger Vorhersagen über die jeweilige Frequenz der Verfügbarkeit getroffen werden. Schließlich muss entsprechender Aufwand zur Integration dieser Daten betrieben werden [Inmon 1996, 264f.]. Fischer stuft externe Daten gar als

„*Achillesferse von Data-Warehouse-Projekten*" ein und sieht in ihnen „*sehr gefährliche Katalysatoren*" für Barrieren, an denen entsprechende Projekte scheitern können [Fischer 2001, 107].

- **Datenschutz**

Data Warehousing kann nicht zuletzt auch zur langfristigen und zur Auswertung optimierten Archivierung von *personenbezogenen* Daten herangezogen werden. In diesem Zusammenhang wurde das Data Warehouse-Konzept insbesondere in Kombination mit Technologien wie Data Mining (vgl. Kap. 3), die das automatische Auffinden zuvor unbekannter Muster in großen Datenmengen erlauben, von den Datenschutzbeauftragten von Bund und Ländern bereits im Jahr 2000 kritisch eingestuft [KDBA 2000]. Dieser Aspekt wird im Anschluss an die Vorstellung des Gebietes des Data Mining in Kap. 3.5 noch einmal aufgenommen.

2.3 Exkurs – Modellierung multidimensionaler Daten durch OLAP

Das Konzept des Online Analytical Processing (OLAP) wurde, ebenso wie das Data Warehouse-Konzept, Mitte der 90er Jahre des 20. Jahrhunderts entwickelt, um eine effizientere Analyse von Unternehmensdaten zu ermöglichen. Beide Konzepte werden oft gemeinsam eingesetzt, sind jedoch von ihrem Ansatz her deutlich zu unterscheiden. Während das Data Warehouse zur redundanten Bereitstellung analyserelevanter Daten dient, kann OLAP dazu verwendet werden, diese Daten zu organisieren. OLAP erlaubt die Speicherung multidimensionaler Daten in Form hochdimensionaler Würfel und unterstützt so eine flexible Betrachtung dieser Daten aus verschiedenen Perspektiven, in verschiedenen Hierarchieebenen, Ausschnitten etc. OLAP kann sowohl über spezielle multidimensionale Datenbanksysteme realisiert oder in relationalen Datenbanksystemen modelliert werden (vgl. Kap. 2.3.3 und 2.3.4).

2.3.1 Motivation – eine dimensionenbezogene Datensicht

Bei der Durchführung betriebswirtschaftlicher Analysen ist die flexible Betrachtung großer Mengen von Ausgangsdaten unter verschiedenen Blickwinkeln wie Produkt, Filiale, Umsatz, Kunde, Quartal etc. erforderlich. Die Bereitstellung solcher nutzerdefiniert wählbarer Blickwinkel, die auch als Dimensionen bezeichnet werden, verspricht eine Vereinfachung der Analysetätigkeit und eine erhöhte Akzeptanz durch die Anwender, die sich so auf eine ihrer Denkstruktur mehr entsprechende Weise den Daten nähern können [Schinzer et al. 1999, 39]. Relationale Datenbankmanagementsysteme (RDBMS), die den heutigen Stand der Technik für die Verwaltung operationaler Daten darstellen [Eicker 2001, 70], basieren auf dem relationalem Datenmodell und dem Konzept des *Online Transaction Processing* (OLTP), das für transaktionsorientierte, operative Aufgaben optimiert ist. Hingegen sind sie weniger für die Analyse multidimensionaler Datenstrukturen aus unterschiedlichen Perspektiven geeignet [Holthuis 1996, 167] [Gärtner 1996, 147ff.] [Schinzer et al. 1999, 38]. Vor diesem Hintergrund stellte E.F. Codd 1993 mit OLAP (für *Online Analytical Processing*) ein Konzept vor, dass sich bewusst gegen OLTP abgrenzt und die flexible Betrachtung großer Datenmengen aus verschiedenen Dimensionen in den Vordergrund stellt [Codd et al. 1993].

OLAP erlaubt die Organisation von Daten in Form von vieldimensionalen Datenwürfeln (Hypercubes), wobei jede Achse des Würfels einer Dimension, also einer Sicht auf die Daten, entspricht [Holten et al. 2001, 11] [Schinzer et al. 1999, 39ff.] [Schelp 1998] [Bulos 1998]. Während OLAP die Realisierung höherdimensionaler Würfel unterstützt, werden in der Literatur aus Gründen der Anschaulichkeit für graphische Darstellungen von Hyper-

cubes in der Regel dreidimensionale Datenwürfel gewählt[46]; Abb. 2.9 zeigt einen dreidimensionalen Datenwürfel für Verkaufsumsätze mit den drei Dimensionen Zeit, Region und Produkt.

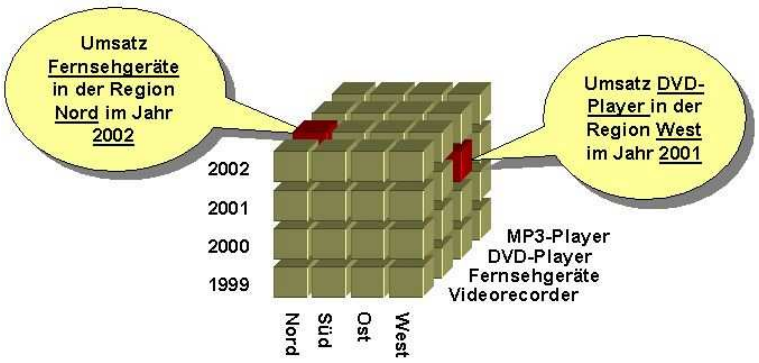

Abb. 2.9 - Schematische Darstellung eines dreidimensionalen OLAP-Datenwürfels.

2.3.2 Vergleich von OLAP und OLTP

OLAP-Datenbestände, die für eine nachfolgende Analyse vorgehalten werden, unterscheiden sich in vielfacher Hinsicht von nach dem OLTP-Konzept modellierten operationalen Daten. Wesentliche Unterschiede sind [Schinzer et al. 1999, 46f.]:

Anwendungsziel
- Während OLTP-Datenbestände für die transaktionsorientierte operative Datenverarbeitung genutzt werden, dienen OLAP-Datenbestände der Unterstützung einer dynamischen Datenanalyse.

Redundanzen
- Die Transaktionsorientiertheit von OLTP-Datenbeständen erfordert einen hohen Grad von Redundanzfreiheit, der durch die Anwendung der Normalisierungsregeln des relationalen Datenmodells erreicht wird; bei OLAP-Datenbeständen wird hingegen eine redundante Datenspeicherung durch Denormalisierung bzw. Vermeidung von Normalisierung bewusst in Kauf genommen.

Datenumfang
- Während OLTP-Datenbestände typischerweise operative Daten eines Unternehmens mit vergleichsweise geringem Zeithorizont enthalten, dienen OLAP-Datenbestände zur Analyse. Sie können auch historische Daten mit einem sehr großen Zeithorizont und damit ein um ein Vielfaches größeres Datenvolumen[47] enthalten.

Zugriffsformen
- Während OLTP-Datenbestände vielfältigen Veränderungen durch das operationale Tagesgeschäft unterliegen und entsprechend sowohl lesende wie schreibende Zugriffe unterstützen müssen, wird auf OLAP-Datenbestände nur lesend zugegriffen.

Datenmenge pro Zugriff
- Während die für einen einzelnen Zugriff benötigte Datenmenge bei OLTP-Datenbeständen in der Regel relativ klein ist, erfordern OLAP-Zugriffe, die typischerweise komplexe Verknüpfungen und Selektionen realisieren, in der Regel die Einbeziehung eines we-

[46] Vgl. bspw. [Holthuis 1996, 173, Abb. 4] [Chamoni, Zeschau 1996, 72, Abb. 7] [Schelp 1998, 266, Abb. 2] [Schinzer et al. 1999, 40, Abb. 4/1] [Holten et al. 2001, 11, Abb. 3].
[47] So liegen nach [Scalzo 2003, 3, Tab. 1-1] typische Datenbankgrößen bei OLTP zwischen 10 und 400 Gigabyte, bei OLAP hingegen zwischen 100 und 800 Gigabyte und bei Data Warehouses im Bereich von 800 bis 80.000 Gigabyte.

sentlich höheren Volumens pro Zugriff.

Abfragekomplexität ▸ Analog zur einzubeziehenden Datenmenge fällt auch die Komplexität einzelner Abfragen auf OLAP-Datenbeständen deutlich höher aus als bei typischer Abfragen auf OLTP-Datenbestände.

Arbeitsweise ▸ Viele Zugriffe auf OLTP-Datenbestände finden wiederholt statt und können über vordefinierte Abfragestatements automatisiert werden; die Arbeit mit OLAP-Datenbeständen hingegen erfolgt von ihrer Konzeption her als spontaner, iterativ ablaufender und entsprechend unplanbarer Prozess.

2.3.3 Multidimensionales OLAP (MOLAP)

Während das OLAP-Konzept stets eine logische Multidimensionalität anbietet, unterscheiden sich die verfügbaren Werkzeuge hinsichtlich der physischen Realisierung der Datenspeicherung. Sie können in multidimensionale und relationale OLAP-Lösungen (vgl. Kap. 2.3.4) aufgeteilt werden. *Multidimensionales OLAP* (MOLAP) wird über spezielle Datenbanksysteme mit einer für multidimensionale Strukturen optimierten Speicherstruktur realisiert, die eine direkte Adressierung der einzelnen Zellen erlaubt. Die physische Modellierung erfolgt dabei entweder über einen einzelnen Hypercube, der den gesamten Datenbestand in einem einzigen Würfel abbildet, oder über die Unterteilung der Daten in mehrere, jeweils möglichst dicht besetzte Würfel (Multicubes)[48]. Die in der Regel proprietären, also nicht standardisierten MOLAP-Systeme erlauben einen extrem schnellen Datenzugriff und die flexible Darstellung verschiedener Blickwinkel auf die gespeicherten Daten [Schinzer et al. 1999, 47, 52f.] [Eicker 2001, 70f.]. Die Speicherung von Daten in Form eines multidimensionalen Würfels bedeutet allerdings auch einen mit der Anzahl der abzulegenden Dimensionen exponentiell ansteigenden Speicherbedarf. Insbesondere die Ablage dünnbesetzter Daten in einem Hypercube resultiert in extrem großen Datenwürfeln mit nur wenigen belegten Zellen[49]. In der Praxis muss dabei durchaus mit dem Auftreten dünnbesetzter Daten gerechnet werden – so sind etwa im Bereich von betriebswirtschaftlicher Planung und Kontrolle nur Besetzungsgrade in einer Größenordnung zwischen einem und zehn Prozent zu erwarten [Chamoni 1998a, 241]. Lösungsansätze für dünnbesetzte Daten bestehen in ihrer Aufteilung auf jeweils dichter besetzte Multicubes sowie in Komprimierungsverfahren für Hypercubes [Schinzer et al. 1999, 53].

2.3.4 Relationales OLAP (ROLAP)

Im Gegensatz zu den spezialisierten MOLAP-Systemen basiert *relationales OLAP* (ROLAP) auf klassischen relationalen Datenbankmanagementsystemen. Die Multidimensionalität wird hier durch entsprechende Modellierungstechniken auf zweidimensionale Tabellen abgebildet. Für die Realisierung von ROLAP, das auch als virtuelles OLAP bezeichnet wird, existieren mit den nachfolgend skizzierten Schemata Star-Schema, Snowflake-Schema und Galaxy-Schema unterschiedliche Modellierungsformen, die jeweils auf einer zumindest partiellen Denormalisierung basieren.

- **Star-Schema**

Das *Star-Schema* (vgl. bspw. [Schinzer et al. 1999, 48ff.] [Hahne 1998, 110ff.] [Holten et al. 2001, 20ff.]) stellt das einfachste Datenmodell für die Modellierung multidimensionaler Da-

[48] Für einen Vergleich von Hypercube- und Multicube-Ansatz siehe [Pendse 2001].
[49] Angestrebt ist aus Gründen der Performance ein Halten der gesamten Struktur im Hauptspeicher [Chamoni 1998a, 243].

tenstrukturen in relationalen Datenbankmanagementsystemen dar. Der Name des Schemas ist dabei auf die sternförmige Anordnung von Dimensionstabellen um eine Faktentabelle zurückzuführen (vgl. Abb. 2.10).

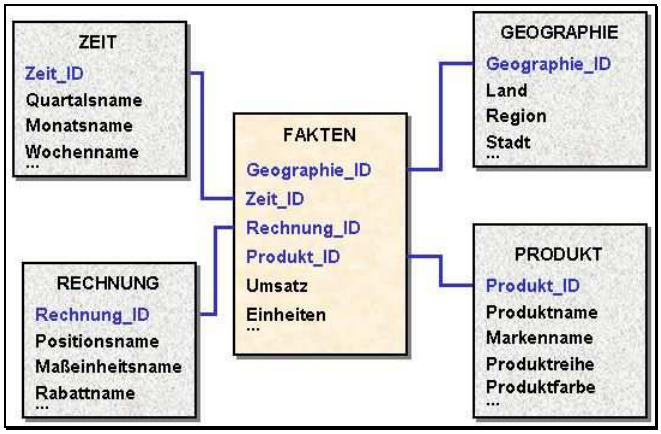

Abb. 2.10 - Star-Schema mit den vier Dimensionen Zeit, Rechnung, Geographie und Produkt[50].

Die *Faktentabelle* bildet die zentrale Tabelle des Star-Schemas. Sie enthält einen Datensatz für jeden relevanten Eintrag und kann entsprechend groß werden. Jeder Datensatz in der Faktentabelle besteht aus Attributen zur Speicherung der eigentlichen Werte (bspw. Stückzahlen, Umsatz etc.) sowie einer Reihe von Schlüssel-Attributen, die die Verbindung zu den Dimensionstabellen herstellen. *Dimensionstabellen* spannen hingegen die jeweiligen Dimensionen auf, unter denen die eigentlichen Werte betrachtet werden sollen, bspw. Zeitbezug, Produktdetails, geographische Zuordnungen etc. Jede Dimensionstabelle enthält das entsprechende Schlüsselattribut, über das sie mit der Faktentabelle verbunden ist, sowie weitere Attribute, die die Ausprägung dieser Dimension spezifizieren (bspw. Produktname, Produktgruppe, Produktfarbe etc). Dabei können innerhalb einer Dimensionstabelle auch Aggregationen wie Woche, Monat, Quartal aufgebaut werden. Nach [Hahne 1998, 117f.] kann für eine Steigerung der Abfragegeschwindigkeit ein Star-Schema auch um in zusätzlichen Faktentabellen abgelegte Aggregationen bereichert werden (sog. *Fact-Constellation-Schema*). Schinzer et al. zufolge bieten Einfachheit und Übersichtlichkeit des Star-Schemas einerseits eine gute Kommunikationsbasis für die Diskussion mit den Anwendern; die redundante Speicherung von Daten in den Dimensionstabellen bringt auf der anderen Seite allerdings erhöhten Speicherbedarf mit sich, der sich nachteilig auf die Abfrageperformance auswirken kann [Schinzer et al. 1999, 50].

- **Snowflake-Schema**

Das *Snowflake-Schema* (vgl. Abb. 2.11) vermeidet im Gegensatz zum Star-Schema die Redundanzen in den Dimensionstabellen. Während die Faktentabelle im Vergleich zum Star-Schema unverändert bleibt, wird bei den Dimensionstabellen eine Normalisierung durchgeführt: Eine Dimension wird nun nicht mehr in einer einzigen Tabelle gehalten, sondern auf mehrere Dimensionstabellen aufgeteilt, die wiederum über Schlüssel verknüpft werden [Schinzer et al. 1999, 50f.] [Holten et al. 2001, 22f.]. Die so erreichte Redundanzvermeidung führt zu entsprechender Verringerung des notwendigen Speicherbedarfs. Allerdings muss hier der benötigte Speicherplatz für die zusätzlich hinzukommenden Schlüssel mit einbezogen werden, die zudem den erforderlichen Organisationsaufwand erhöhen. Insbesondere kann die erhöhte Verknüpfungstiefe zu verringerter Abfrageperformance führen; einen möglichen Ausweg stellt hier die Kompromisslösung dar, nur besonders große Dimensionstabellen aufzuspalten (sog. *Partitial Snowflake Schema*) [Schinzer et al. 1999, 50f.].

[50] Abbildung erstellt in Anlehnung an [Schinzer et al. 1999, 49, Abb. 4/4].

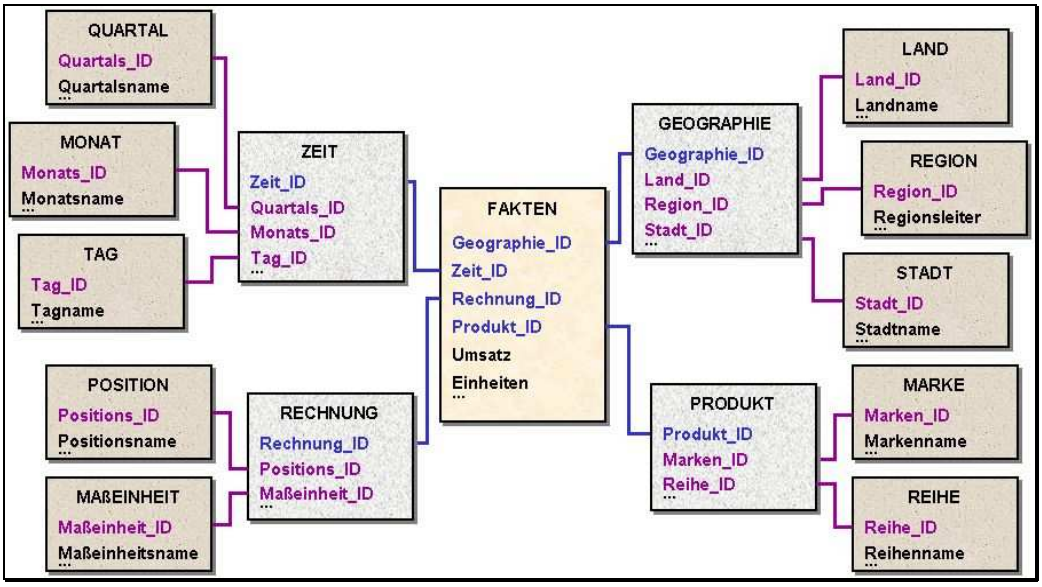

Abb. 2.11 - Snowflake-Schema[51].

- **Galaxy-Schema**

Sowohl Star-Schema wie Snowflake-Schema verwenden zur Speicherung sämtlicher „Fakten" jeweils eine einzige Faktentabelle. Wenn Fakten mit jeweils unterschiedlichen Dimensionen abzulegen sind, führt diese Art der Modellierung jedoch zu einer hohen Anzahl von Schlüsselattributen, die nicht durchgehend belegt sind und so zu einer erhöhten Abfragekomplexität beitragen. Dieser Nachteil kann durch Verwendung des *Galayxy-Schema* (vgl. Abb. 2.12) umgangen werden, das mehrere voneinander unabhängige Faktentabellen verwendet. Fakten mit gleicher Dimensionierung werden dabei jeweils in einer eigenen Faktentabelle abgelegt und mit den entsprechenden Dimensionstabellen verknüpft [Hahne 1998, 115f.] [Schinzer et al. 1999, 51f.].

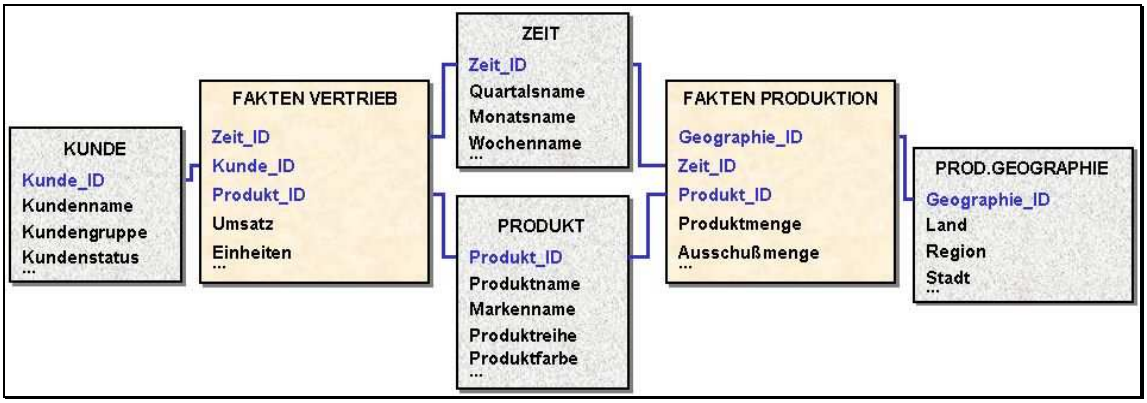

Abb. 2.12 - Galaxy-Schema[52].

Als Argumente für eine Realisierung von OLAP über relationale Datenbankmanagementsysteme werden u.a. die Ausgereiftheit dieser Systeme, das zu diesen in den Unternehmen bereits vorhandene Know-How sowie die Vermeidung von Kosten für ein zusätzliches multidimensionales Datenbanksystem angeführt [Gärtner 1996, 149f.] [Eicker 2001, 70]. Allerdings können Eicker zufolge auf relationalen Datenbankmanagementsystemen basierende Realisierungen die Performance multidimensionaler Datenbanksysteme „*nach heutigem Wissenstand nicht erreichen*" [Eicker 2001, 71].

[51] Abbildung erstellt in Anlehnung an [Schinzer et al. 1999, 51, Abb. 4/5].
[52] Abbildung erstellt in Anlehnung an [Schinzer et al. 1999, 52, Abb. 4/6].

2.3.5 OLAP-Funktionalität

OLAP-Datenwürfel können vom Anwender verschiedenen Operationen unterworfen werden, um die enthaltenen Daten jeweils aus unterschiedlichen Perspektiven betrachten zu können (vgl. bspw. [Holten et al. 2001, 11f.]). So wird der Begriff *Rotation* für ein Drehen oder Kippen des Datenwürfels verwendet, durch das jeweils ein anderer Blickwinkel auf die Daten entsteht (vgl. Abb. 2.13a). Unter *Ranging* wird hingegen die Einschränkung von Dimensionen des Würfels verstanden. Da so stets ein Teilwürfel entsteht, wird diese Operation auch als *Dicing*[53] bezeichnet (vgl. Abb. 2.13b).

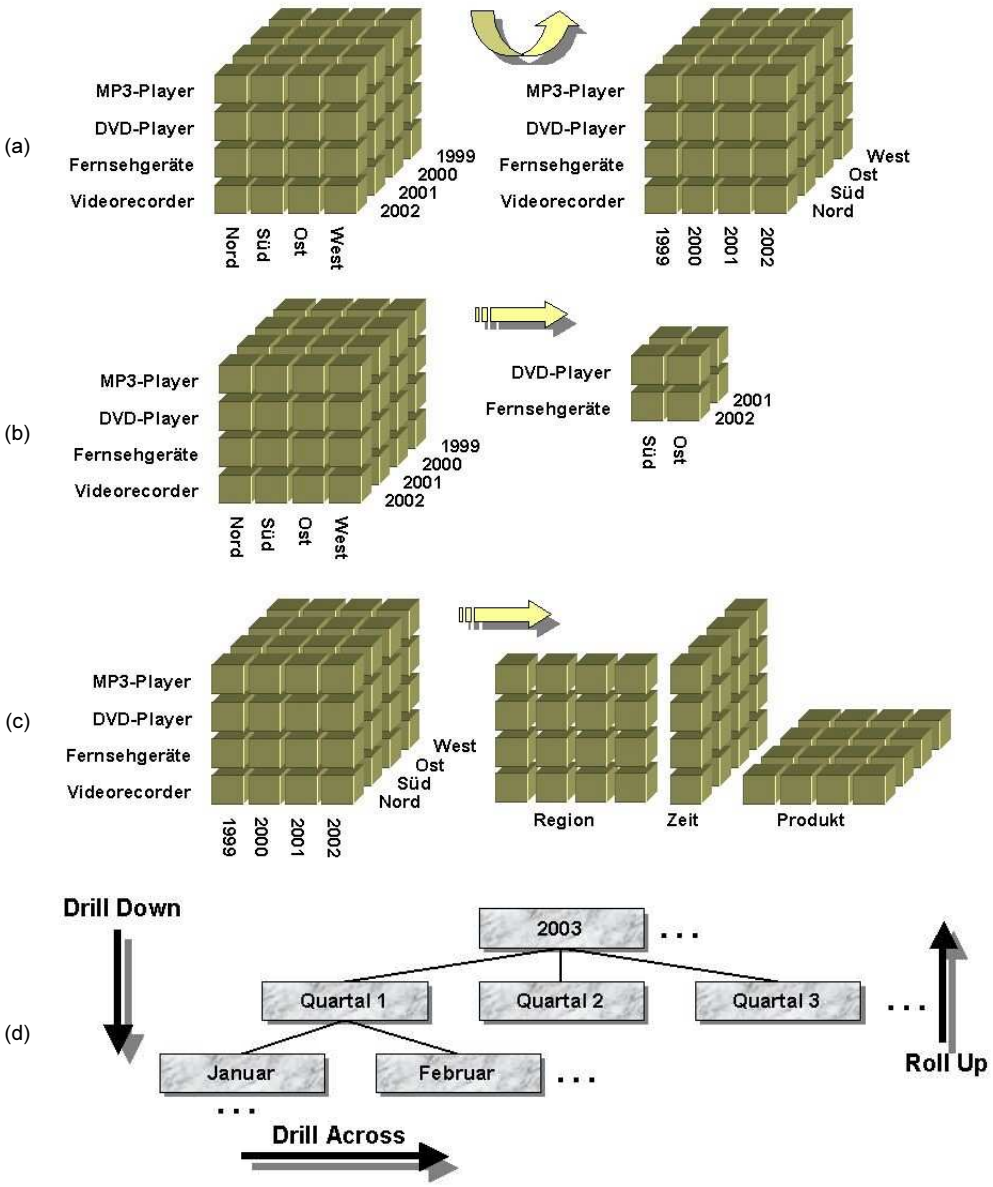

Abb. 2.13 - OLAP-Funktionalität[54]: (a) Rotation; (b) Dimensionsreduktion durch Ranging / Dicing; (c) Isolation einzelner Scheiben aus dem Datenwürfel durch Slicing; (d) Navigation über verschiedenen Aggregationsebenen durch Drill Down, Roll Up und Drill Across.

Unter *Slicing* wiederum versteht man das Herausschneiden einer „Scheibe" aus dem Datenwürfel, etwa die Verkaufsumsätze in der Region Süd, die Verkaufsumsätze für das Jahr

[53] [Schinzer et al. 1999, 41] verwenden den Begriff Dice hingegen für das Drehen bzw. Kippen des Würfels.
[54] Abbildung erstellt in Anlehnung an [Schinzer et al. 1999, 40ff., Abb. 4/1 bis 4/3].

2001 oder die Verkaufsumsätze für MP3-Player (vgl. Abb. 2.13c). OLAP erlaubt ferner die Betrachtung der Daten im Datenwürfel auf unterschiedlich detaillierten Aggregationsebenen, die auch als Konsolidierungsebenen bezeichnet werden. Mit Drill Down, Roll Up sowie Drill Across werden dabei verschiedene Arten der Navigation über diese Ebenen unterschieden (vgl. Abb. 2.13d). Als *Drill Down* wird der Übergang von einer weniger detaillierten Ebene zu einer Ebene mit höherem Detailgrad bezeichnet, also beispielsweise von Jahren zu Quartalen oder von Quartalen zu einzelnen Monaten. Der umgekehrte Weg, also der Übergang zu einer weniger detaillierten, aggregierten Ebene - also etwa von Quartalen zu Jahren - wird *Roll up* genannt. Die Bewegung innerhalb einer Aggregationsebene, also etwa von einem Quartal zum nächsten, bezeichnet man als *Drill Across* [Schinzer et al. 1999, 42f.].

2.4 Alternative Ansätze zur Datenintegration

2.4.1 Virtuelle oder materielle Datenintegration

Das Data Warehouse-Konzept stellt nicht den einzigen aktuellen Ansatz zur Integration heterogener Daten dar. In den vergangenen Jahren wurden verschiedene weitere Konzepte zur Überwindung der Heterogenität individueller Datenquellen hervorgebracht, die jeweils nach Art der Herangehensweisen in eine materielle oder virtuelle Integration zu unterscheiden sind [Domenig, Dittrich 1999] [Dittrich, Domenig 1999] [Busse et al. 1999].

Beim Konzept der *materiellen Integration* werden die getrennten Datenquellen zunächst physisch zu einer neuen Quelle zusammengefasst, gegen die dann Zugriffe erfolgen. Zu diesen Ansätzen zählt das beschriebene Konzept des Data Warehouse mit seiner Zusammenfassung und redundanten Vorhaltung relevanter Datenbestände neben den Originalsystemen für einen lesenden Zugriff. Ebenfalls eine materielle Integration erfolgt durch die Migration vorher getrennter Datenbestände in ein einzelnes einheitliches Datenbanksystem, auf das dann lesend und schreibend zugegriffen werden kann. Anders als beim Data Warehouse werden hier die einzelnen Quellen nach der Migration nicht mehr verwendet. Als *virtuelle Integration* werden hingegen Konzepte bezeichnet, bei denen anzusprechenden Datenbestände in ihren jeweiligen Quellen verbleiben und erst während eines Zugriffs - eben „virtuell" - integriert werden. Hierzu zählen[55] Konzepte wie Mediatoren, die einen reinen Lesezugriff auf verschiedene Arten von Datenquellen realisieren, sowie föderierte Datenbanksysteme[56], die mehrere getrennt betriebene Datenbanksysteme für Lese- und Schreibzugriffe so zusammenkoppeln, dass die Einzelsysteme bis zu einem gewissen Grad autonom weiterbetrieben werden können.

Konzept	Integration	Zugriffe
Data Warehouse	Materiell	lesend
Mediatoren	Virtuell	lesend
Migration	Materiell	lesend + schreibend
Föderierte Datenbanksysteme	Virtuell	lesend + schreibend

Tab. 2.2 - Virtuelle und materielle Integrationskonzepte im Vergleich.

Tab. 2.2 gibt einen Überblick über die verschiedenen Integrationskonzepte und die jeweils

[55] Nicht eingegangen wird hier auf Suchmaschinen bzw. Meta-Suchmaschinen (d.h. Suchmaschinen, die ihrerseits auf andere Suchmaschinen zugreifen) für das World Wide Web, die [Domenig, Dittrich 1999] [Dittrich, Domenig 1999] ebenfalls zu den Systemen für eine virtuelle Integration zählen.

[56] [Dittrich, Domenig 1999] verweisen darauf, dass föderierte Datenbanksysteme auch einen Teil der Daten materiell integrieren können, rechnen sie jedoch primär zu den virtuellen Integrationsansätzen, da der Hauptteil der Daten in den Originalsystemen verbleibt. [Busse et al. 1999, 17f., Tab. 3] stufen föderierte Datenbanksysteme ebenfalls als Konzept einer virtuellen Integration ein.

unterstützten Zugriffsformen. Beide Herangehensweisen bieten jeweils Vor- und Nachteile: Materielle Integration erlaubt eine höhere Anfrageperformance sowie Kontrolle über die integrierten Daten, bringt andererseits jedoch vermehrten Speicherbedarf mit sich und erfordert komplexe Prozesse der Datenaktualisierung [Busse et al. 1999, 16]. Nachfolgend werden drei weitere Ansätze zur Datenintegration überblicksartig vorgestellt. Zunächst werden mit Mediatoren (Kap. 2.4.2) und Migration (Kap. 2.4.3) zwei von ihrer Idee her einander besonders entgegengesetzte Integrationskonzepte dargestellt; daran anschließend wird auf das Konzept der föderierten Datenbanksysteme eingegangen (Kap. 2.4.4).

2.4.2 Vermittler im Sinne der Anfrage – Mediatoren

Zu den Vertretern des virtuellen Integrationsansatzes zählt das Konzept der *Mediation*[57], das von Gio Wiederhold [Wiederhold 1992a] eingeführt wurde und die Realisierung von reinen Lesezugriffen auf heterogene Quellen verfolgt. Die Datenquellen sind dabei nicht auf Datenbanken beschränkt, sondern können strukturierte, unstrukturierte und semistrukturierte Datenquellen einbeziehen und bspw. auch Suchmaschinen einbinden (vgl. [Domenig, Dittrich 2001]). Eine zentrale Rolle kommt in diesem Ansatz dem Begriff des *Mediators* zu; ein Mediator wird von Wiederhold definiert als ein

- „[…] *software module that exploits encoded knowledge about some sets or subsets of data to create information for a higher layer of applications.*" [Wiederhold 1992a]

Ein einzelner Mediator soll dabei nicht zu komplex ausfallen:

- „[…] *it should be small and simple so that it can be maintained by one expert or at most by a coherent group of experts.*" [Wiederhold 1992a]

Mediatoren stellen verschiedene sog. *value-added services* zur Verfügung. Ihre Aufgaben bestehen insbesondere in der Lokalisierung von verschiedenen heterogenen Quellen, im Zugriff auf diese Quellen, der Transformation der so erhaltenen Daten in ein allgemeines Datenmodell, der Integration der transformierten Daten sowie die Verdichtung der integrierten Daten zur Steigerung der Informationsdichte des Ergebnisses. Die Basisarchitektur von Mediatorensystemen (vgl. Abb. 2.14) besteht aus drei Schichten, wobei eine vermittelnde Zwischenschicht (Mediation Layer) zwischen Anwendungen (Application Layer) und heterogener Datenschicht (Foundation Layer) eingezogen wird und für die Integration der Daten verantwortlich ist [Wiederhold 1994] [Wiederhold 1995].

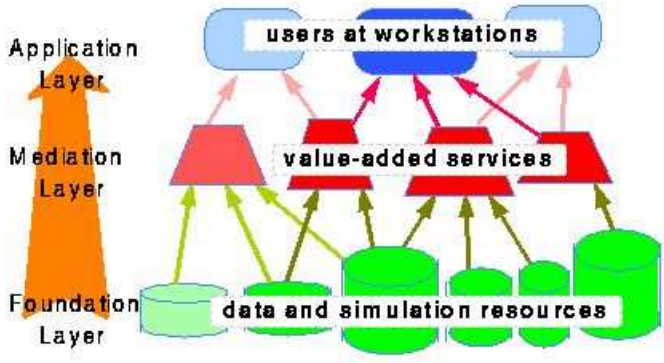

Abb. 2.14 - Die Basisarchitektur von Mediatorensystemen[58]: Application Layer, Mediation Layer und Foundation Layer.

Der Zugriff der Mediatoren auf die einzelnen heterogenen Datenquellen wird dabei durch sog. *Wrapper* erleichtert, die die einzelnen Quellen einkapseln und den Mediatoren einen

[57] Der Begriff *Mediation* bedeutet Vermittlung oder vermittelndes Dazwischentreten.
[58] Abbildung übernommen aus [Wiederhold 1995, Fig. 1].

jeweils gleichartigen Zugriff auf die Quellen erlauben. Wrapper werden basierend auf den Anfragemöglichkeiten der jeweiligen Quellen implementiert; falls eine Quelle diese nicht zur Verfügung stellt, können sie durch einen entsprechenden Wrapper bereitgestellt werden (vgl. [Domenig, Dittrich 1999]). Die Einbindung von unstrukturierten Quellen kann durch die Entwicklung von sog. *Extractors* unterstützt werden (vgl. [Chawathe et al. 1994]); eine automatisierte Identifikation und Einbindung neuer Quellen wird durch eine spezielle Art von Mediatoren, den sog. *Facilitators*, angestrebt [Wiederhold, Genesereth 1997].

Varianten des Mediatoren-Ansatzes wurden in einer Reihe von Projekten umgesetzt; prominente Implementierungen[59] sind u.a. The Stanford-IBM Manager of Multiple Information Sources (TSIMMIS)[60] [Chawathe et al. 1994], Information Manifold von AT&T [Levy et al. 1996a] [Levy et al. 1996b] oder Garlic[61] vom IBM-Research Center Almaden [Tork Roth, Schwarz 1997] [Haas et al. 1999]; zu den jüngeren Arbeiten zählt bspw. SINGAPORE [Domenig, Dittrich 2001].

2.4.3 Integration durch Migration

Einen vollständig anderer Ansatz besteht in der Realisierung einer materiellen Integration durch Migration der Daten aus lokalen heterogenen Systemen in ein einzelnes Datenbanksystem[62] mit dem Ziel, nach Abschluss der Migration nur noch dieses weiterzuverwenden (vgl. Abb. 2.15). Angestrebt ist also ein integrierter Datenbestand, für den nachfolgend volle Datenbankfunktionalität - also nicht nur Abfragen, sondern auch schreibende Zugriffe - zur Verfügung gestellt werden kann. Mit diesem Konzept sind allerdings potentiell wesentliche Nachteile verbunden. So kann sich die Migration der Daten aufgrund der notwendigen Transformationen als sehr aufwendiger Prozess erweisen. Für den Datenbestand müssen neue Zugriffskontrollstrategien erarbeitet werden; insbesondere ist die Veränderung der bereits existierenden Applikationen notwendig, damit diese nun mit der neuen Datenbasis arbeiten können. Zu berücksichtigen ist ferner, dass es während einer Migration zu einem zwischenzeitlichen Ausfall einzelner Systeme kommen kann. Dennoch kann dieser Ansatz einen gangbaren Weg zur Datenintegration darstellen; bspw. wenn vollständige Datenbankfunktionalität für den Zugriff auf die Daten erforderlich ist und bestehende Applikationen nicht mehr oder nicht mehr in ihrer bisherigen Form verwendet werden sollen [Domenig, Dittrich 1999] [Dittrich, Domenig 1999] [Conrad 1997, 41].

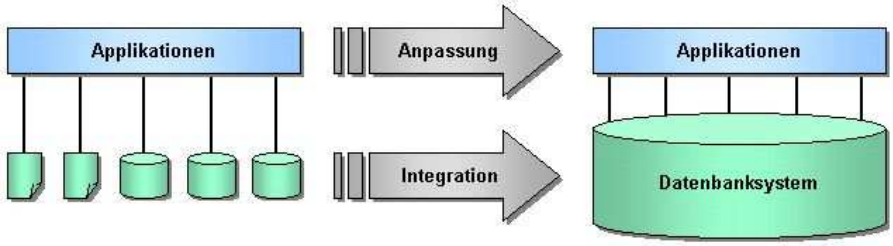

Abb. 2.15 - Integration von Daten durch Migration in ein Datenbanksystem.

Wie problematisch der Versuch werden kann, komplexe Einzelsysteme und ihre Anwendungen in ein System zu integrieren, illustriert das Beispiel des World-Wide Military Command and Control System (WWM CCS). Das WWM CCS wurde seit 1966 entwickelt, um

[59] Für einen Vergleich einiger dieser Systeme siehe [Busse et al. 1999, 25ff.].
[60] http://www-db.stanford.edu/tsimmis/
[61] http://www.almaden.ibm.com/cs/garlic/
[62] Domenig und Dittrich verwenden hierfür den Begriff „'universal' DBMS", also „universelles" Datenbankmanagementsystem. Dabei ist zu beachten, dass es sich *nicht* um eine besondere Art von Datenbanktechnologie handelt, sondern um die Überführung verschiedener Quellen in ein Datenbanksystem, das dann anstelle der Einzelquellen „universell" für die gegebenen Anforderungen genutzt wird.

den Problemen zu begegnen, die sich aus dem Nebeneinanderbestehen von elf verschiedenen Command and Control Systems (CCS)[63] ergaben. Erst nach rund zehn Jahren wurde die Integration erreicht; das so entstandene System erwies sich in der Folge aufgrund seiner Komplexität und der vielfältigen Abhängigkeiten zwischen System und Applikationen als nahezu ungeeignet für nachfolgende Verbesserungen [Wiederhold 1992b].

2.4.4 Teilverzicht auf Autonomie – Föderierte Datenbanksysteme

Die Bereitstellung voller Datenbankfunktionalität unter Umgehung der Nachteile einer Migration der Daten wird mit dem Konzept der *föderierten Datenbanksysteme* (*Federated Database Systems*, FDBS) angestrebt. Ein föderiertes Datenbanksystem (vgl. Abb. 2.16) ist dabei nach [Conrad 1997, 7f.] ein aus mehreren (teil-)autonomen Datenbanksystemen bestehendes Gesamtsystem, das gegebenenfalls eine zusätzliche Schicht bereitstellt, die neuen globalen Anwendungen den Zugriff auf die an der Föderation beteiligten Systeme ermöglicht. Die Kernidee eines föderierten Datenbanksystems besteht im Zusammenschluss vorhandener Datenbanksysteme zu einem Verbund, der es den teilnehmenden Systemen, die als *Komponentendatenbanksysteme* oder kurz als *Komponentensysteme* bezeichnet werden, erlaubt, in wesentlichen Zügen autonom weiterzubestehen. Ein wesentlicher Vorteil dieses Ansatzes liegt im Vergleich zur Migration mithin darin, dass bestehende lokale Anwendungen ohne Anpassungsaufwand weiter verwendet werden können. Um übergreifende Funktionalität bereitzustellen, verzichten die Komponentensysteme auf einen Teil ihrer Autonomie; in diesem Sinne besteht also eine gewisse Analogie zu einer politischen Föderation [Conrad 1997, 4ff.].

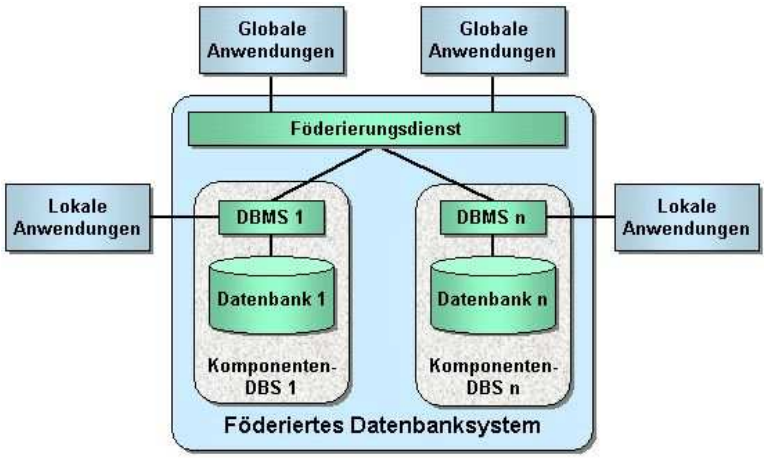

Abb. 2.16 - Allgemeine Architektur föderierter Datenbanksysteme[64].

Die bei einer Föderation zur Disposition stehende Autonomie der Komponentensysteme kann weiter untergliedert werden. Der Begriff der *Entwurfsautonomie* bezieht sich auf die lokalen Datenbankschemata der Komponentensysteme. Wenn diese nicht durch eine übergeordnete Föderierungsebene verändert werden dürfen, ist eine vollständige Entwurfsautonomie gegeben. *Kommunikationsautonomie* ist hingegen gegeben, wenn die Komponentensysteme - bzw. ihre jeweiligen Datenbankadministratoren - selber entscheiden, ob und wann sie in eine Föderation eintreten oder diese wieder verlassen sowie mit welchen anderen Systemen der Föderation sie kommunizieren. *Ausführungsautonomie* schließlich bezeichnet die Freiheit der Komponentensysteme, zu entscheiden, welche Operationen - bspw. Anwendungsprogramme, Anfragen und Änderungen - sie ausführen und zu welchem Zeitpunkt sie dies tun [Conrad 1997, 47ff.].

---

[63] Ein Command and Control System stellt im wesentlichen ein Inventar militärischer Ressourcen zur Verfügung [Wiederhold 1992b].
[64] Abbildung erstellt in Anlehnung an [Conrad 1997, 49, Abb. 3.6].

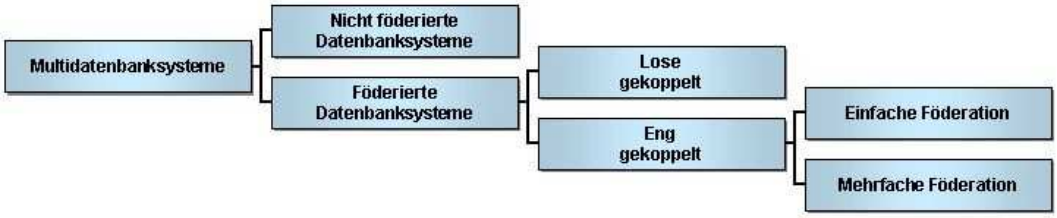

Abb. 2.17 - Unterteilung von Multidatenbanksystemen[65].

Eine genauere Abgrenzung von föderierten Datenbanksystemen erfolgt anhand mehrerer Kriterien (vgl. [Conrad 1997, 40ff.] [Sheth, Larson 1990] sowie Abb. 2.17). So spricht man von einem *Multidatenbanksystem*, wenn ein Verbund mehrerer Datenbanksysteme vorliegt. Je nach Grad der Autonomie der Komponentensysteme in einem Multidatenbanksystem ist dann eine Föderation gegeben oder nicht – wird den Komponentensystemen von einer übergeordneten Ebene jegliche Autonomie entzogen, handelt es sich um ein *nicht föderiertes* Multidatenbanksystem; behalten sie hingegen einen gewissen Grad an Autonomie, liegt ein *föderiertes* Datenbanksystem vor[66]. Föderierte Datenbanksysteme können weiter in lose gekoppelte sowie eng gekoppelte Systeme aufgeteilt werden[67]. *Lose gekoppelte föderierte Datenbanksysteme* besitzen keine übergeordnete Koordinations- und Integrationsinstanz; für die Auswahl der einzubeziehenden Komponentensysteme, den Zugriff auf diese sowie die Integration der Daten sind die lokalen Administratoren und / oder die Benutzer verantwortlich[68]. Bei einer *engen Kopplung* wird hingegen über einen globalen Administrator festgelegt, auf welche Daten der Komponentensysteme zugegriffen werden kann und wie dies geschehen soll. Eng gekoppelte föderierte Datenbanksysteme werden schließlich weiter danach unterteilt, ob eine einfache Föderation oder eine mehrfache Föderation vorliegt. Bei einer *einfachen Föderation* existiert nur ein einziges Schema zur integrierten Beschreibung sämtlicher in die Föderation eingehenden Daten (sog. föderiertes Schema, vgl. u.); bei einer *mehrfachen Föderation* existieren hingegen mehrere föderierte Schemata.

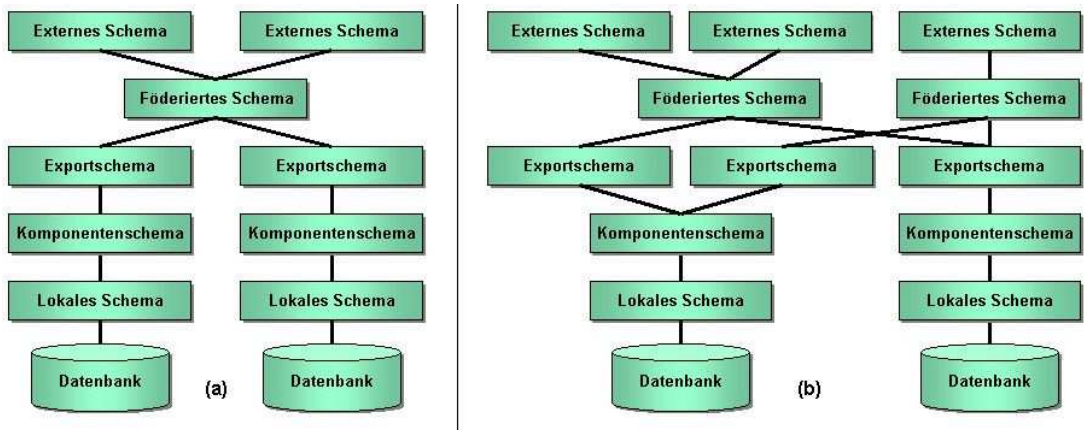

Abb. 2.18 - 5-Ebenen-Schema-Architektur[69]: (a) einfache Föderation; (b) mehrfache Föderation.

[65] Abbildung erstellt in Anlehnung an [Conrad 1997, 42, Abb. 3.3]. Vgl. auch [Sheth, Larson 1990].
[66] [Conrad 1997, 42] verweist darauf, dass diese auf Shet und Larson zurückgehende Aufteilung in nicht föderierte und föderierte Datenbanksysteme „*sehr grob*" sei und nicht übergreifend verwendet werde. Sie könne entsprechend nur als „*Anhaltspunkt für eine begriffliche Einordnung*" konkreter Systeme oder Systemarchitekturen dienen.
[67] Lose gekoppelte Systeme werden dabei nicht von allen Autoren zu den föderierten Datenbanksystemen gerechnet (vgl. [Busse et al. 1999, 17ff.]).
[68] Hierzu werden sog. Multidatenbanksprachen wie MSQL oder SchemaSQL verwendet, um auf mehrere Datenbanksysteme zugleich zuzugreifen (vgl. [Conrad 1997, 228ff.]).
[69] Abbildung erstellt in Anlehnung an [Conrad 1997, 58ff., Abb. 3.9 und Abb. 3.12]. Vgl. auch [Sheth,

Für die Realisierung von föderierten Datenbanksystemen werden unterschiedliche Architekturen[70] vorgeschlagen. Lose Kopplung kann sowohl durch die Import-/Export-Schema-Architektur [Heimbigner, McLeod 1985] wie die Multidatenbank-Architektur [Litwin et al. 1990] realisiert werden. Als „klassische"[71] Architektur für eng gekoppelte[72] föderierte Datenbanksysteme gilt die 5-Ebenen-Schema-Architektur [Sheth, Larson 1990], auf deren Bestandteile (vgl. Abb. 2.18) beispielhaft kurz eingegangen wird. Die 5-Ebenen-Schema-Architektur entscheidet zwischen fünf verschiedenen Arten von Schemata mit jeweils folgender Bedeutung (vgl. [Conrad 1997, 57ff.]):

- **Lokale Schemata**: Ein *lokales Schema* entspricht jeweils dem konzeptuellen Schema eines Komponentensystems, d.h. es beschreibt die Gesamtheit aller von diesem System verwalteten Daten. Die verschiedenen lokalen Schemata können dabei in unterschiedlichen Datenmodellen vorliegen.
- **Komponenten-Schemata**: Die unterschiedlichen Datenmodelle der lokalen Schemata werden in jeweils in ein *Komponenten-Schema* transformiert; hierbei bleiben alle Informationen des lokalen Schemas erhalten, es wird jedoch ein für alle Komponenten-Schemata einheitliches Datenmodell verwendet, das auch im föderierten Schema (vgl. u.) verwendet wird. Ein Komponentenschema dient zur Beseitigung der Datenmodellheterogenität der einzelnen Komponentensysteme, es kann als in ein anderes Datenmodell transformiertes lokales Schema aufgefasst werden.
- **Export-Schemata**: Ein *Export-Schema* beschreibt diejenigen Ausschnitte eines Komponenten-Schemas, die der Föderation zur Verfügung gestellt werden; dabei wird das selbe Datenmodell wie im Komponenten-Schema verwendet.
- **Föderierte Schemata**: Ein *föderiertes Schema* fasst mehrere Export-Schemata zu einem global verfügbaren Schema zusammen. Eine einfache Föderation besitzt nur ein einziges föderiertes Schema (vgl. Abb. 2.18a); liegen mehrere unterschiedliche föderierte Schemata vor, spricht man von einer mehrfachen Föderation (vgl. Abb. 2.18b). Bei der Zusammenführung der Exportschemata müssen dabei vielfältige Integrationsprobleme gelöst werden.
- **Externe Schemata**: Die *externen Schemata* schließlich stellen bestimmte Ausschnitte der föderierten Schemata bereit. Sie stellen auf einzelne Nutzer oder Nutzergruppen zugeschnittene Schnittstellen dar, die die jeweils relevanten Daten in geeigneter Form anbieten.

Zu den Herausforderungen bei der Realisierung föderierter Datenbanksysteme (vgl. [Conrad 1997, 287ff.] zählt neben weiteren Aspekten[73] die Integration heterogener Datenbank-

Larson 1990].

[70] Für eine ausführliche Diskussion der unterschiedlichen Architekturen siehe [Conrad 1997, 50ff.].

[71] Vgl. [Busse et al. 1999, 19].

[72] Die 5-Ebenen-Schema-Architektur erlaubt allerdings auch die Realisierung lose gekoppelter Systeme [Conrad 1997, 68f.].

[73] Hierunter fallen die Sicherstellung der Datenkonsistenz, die Berücksichtigung von lokalen Integritätsbedingungen oder die Einbeziehung von sog. Stored Procedures (ausführbare Programme, die in relationalen Datenbanksystemen abgelegt werden) bzw. von Methoden in objektorientierten Datenbanksystemen [Conrad 1997, 288].

schemata in ein föderationsübergreifendes Schema. Strategien zur Integration der Schemata [Conrad 1997, 74ff.] können zunächst danach unterschieden werden, ob sie in einem Schritt (one-shot) oder in mehreren Schritten erfolgen; sukzessive Integrationsstrategien können wiederum danach unterteilt werden, ob sie jeweils zwei (binäre Integrationsstrategien) oder mehrere Schemata (n-äre Integrationsstrategien) auf einmal integrieren. Binäre Integrationsstrategien können weiter danach unterschieden werden, ob sie balanciert oder gewichtet erfolgen. Bei balancierten binären Integrationsstrategien wird immer ein lokales Schema mit einem andern integriert, die so entstehenden Teilschemata werden dann ebenfalls paarweise zusammengefasst. Es dürfen dabei jeweils nur Schemata zusammengefasst werden, die auf der selben Ebene im Integrationsbaum stehen (vgl. Abb. 2.19a). Eine nichtbalancierte Integrationsstrategie wird als gewichtet bezeichnet, da hier einzelne lokale Schemata bevorzugt und damit mit weniger Veränderungen integriert werden (vgl. Abb. 2.19b und 2.19c). Sogenannte n-äre Integrationsstrategien erlauben hingegen in jedem Schritt die Zusammenfassung einer beliebigen Anzahl von Schemata. Die Zusammenfassung aller Schemata in einem Schritt stellt dabei einen Spezialfall der n-ären Integration dar (vgl. Abb. 2.19d); ein anderes Vorgehen besteht in der iterativen Zusammenfassung jeweils mehrerer lokaler oder teilintegrierter Schemata (vgl. Abb. 2.19e).

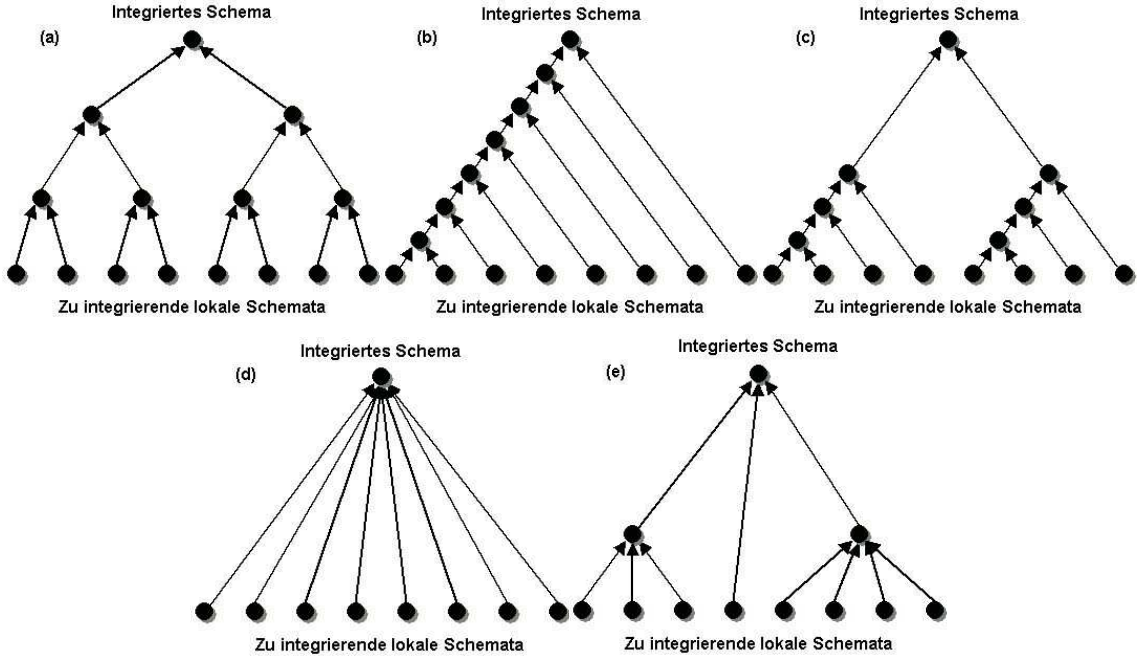

Abb. 2.19 - Verschiedene Strategien zur Schemaintegration[74]: (a) bis (c) stellen binäre Strategien dar: (a) binär balanciert; (b) und (c) binär gewichtet. (d) und (e) zeigen zwei n-äre Integrationsstrategien: (d) one-shot; (e) iterativ.

Der eigentliche Integrationsprozess kann nach [Spaccapietra et al. 1992] in zwei Stufen aufgeteilt werden. Die erste Stufe, von den Autoren als Investigationsphase (Investigation Phase) bezeichnet, dient zur Auffindung von Übereinstimmungen und Unterschieden in den einzelnen Schemata; sie kann entweder als manueller Prozess oder durch Einsatz unterstützender Werkzeuge durchgeführt werden. Anschließend wird als zweite Stufe die Integrationsphase (Integration Phase) halbautomatisch durchgeführt; hier wird unter Berücksichtigung der Korrespondenzen zwischen den Schemata und Verwendung von Integrationsregeln ein integriertes Schema erstellt. Für die Durchführung der Integration stehen unterschiedliche Herangehensweisen zur Verfügung[75]; allerdings existiert Conrad zufolge

[74] Abbildung erstellt in Anlehnung an [Conrad 1997, 75f., Abb. 4.2 bis 4.4].
[75] Vgl. hierzu ausführlich [Conrad 1997, 84ff.].

noch kein vollständiger Schemaintegrationsansatz, der alle Datenmodellierungskonzepte umfassend berücksichtigen kann [Conrad 1997, 288]. Aufgrund ihrer Abhängigkeit vom Schemaintegrationsprozess besitzen eng gekoppelte föderierte Datenbanksystemen zudem eine weitgehend statische Architektur, die nachträgliche Erweiterungen oder Veränderungen lokaler Datenbankschemata erschwert [Conrad 1997, 47] [Busse et al. 1999, 19f.] [Domenig, Dittrich 1999].

2.5 Fazit

Die gemeinsame Nutzung bisher isolierter Datenressourcen ist mit dem zentralen Problem der Heterogenität dieser Quellen konfrontiert, die bereits bei einer Beschränkung auf über Datenbanken bereitgestellte Ressourcen in vielfältigen Formen auftreten kann. Zur Unterstützung bei der Überwindung der Heterogenitäten wurden in den letzten Jahren vielfältige Ansätze hervorgebracht, die nach der Art der durchgeführten Integration in virtuelle und materielle Konzepte mit jeweils spezifischen Vor- und Nachteilen unterschieden werden können. Herauszuheben ist dabei das Konzept des Data Warehouse, das aus den vielfältigen Herausforderungen einer integrierten Analyse heterogener Unternehmensdaten entstand und in diesem Umfeld zunehmend Verbreitung findet. Als Vorteil dieses Konzeptes ist zu werten, dass es einen Weiterbetrieb der operationalen Systeme und der darauf basierenden Anwendungen erlaubt und zugleich relevante Daten aus diesen Systemen für eine effiziente Auswertung bereitstellen kann.

Aber auch die anderen vorgestellten Ansätze - Mediatoren, Migration und föderierte Datenbanksysteme - besitzen jeweils ihre Berechtigung und können je nach Anforderungen des Anwendungsgebietes die am besten geeignete Lösung zur Datenintegration darstellen. Keines dieser Konzepte kann jedoch eine vollständig automatisierte Integration beliebiger Datenquellen leisten; der hohe Aufwand, der für die Bereitstellung und Aktualisierung eines integrierten Datenbestandes auch bei Nutzung von Konzepten wie Data Warehouse investiert werden muss, gibt deutliche Hinweise auf die hierbei zu bewältigenden vielfältigen Herausforderungen.

Die vorgestellten Technologiekonzepte können hier je nach Anwendungsgebiet die Integration von Daten aus verschiedenen Quellen effizient unterstützen; eine Zusammenführung wird dabei jedoch stets dann an Grenzen stoßen, sobald inhaltliche Fragen berührt werden, die nur anhand entsprechender fachlicher Klärungsprozesse gelöst werden können. Dass hierbei schnell nichttriviale Fragestellungen aufgeworfen werden, veranschaulichen zwei Beispiele aus den Bereichen Gesundheitswesen und Sozioökonomie:

- **Aufwendungen verschiedener Länder für das Gesundheitswesen**

Die Organisation for Economic Co-operation and Development (OECD)[76] weist bereits 1985 auf einen Mangel an international einheitlichen Richtlinien zur Beschreibung der Ausgaben im Gesundheitswesen hin [OECD 1985]. Schneider et al. stellen fest, dass auch 1998 die Aufwendungen verschiedener Länder für das Gesundheitswesen nur eingeschränkt miteinander verglichen werden können. Es bestehen dabei zum Teil erhebliche Unterschiede darin, welche Bereiche jeweils zum Gesundheitswesen gerechnet werden. So zählt etwa das Statistische Bundesamt der Bundesrepublik Deutschland zu den diesbezüglichen Kosten auch Krankengeld sowie Berufs- und Erwerbsunfähigkeitsrenten, während dies in anderen Ländern nicht üblich ist. Zusätzlich sind in den jeweiligen Daten, die verschiedene europäische Länder an die OECD weiterleiten, Ausgaben für den Pflegebereich unvollständig oder gar nicht in den gesamten Gesundheitsausgaben berücksichtigt [Schneider et al. 1998].

[76] http://www.oecd.org/home/

- **Primärenergiebedarf einzelner Länder**

Bei der Zusammenführung von sozioökonomischen Daten der United Nation Statistic Division (UNSD)[77] sowie der International Energy Agency (IEA)[78], einer Unterabteilung der OECD, stellte das Potsdam-Institut für Klimafolgenforschung im Jahr 2003 z.T. erhebliche Abweichungen bspw. bezüglich des jeweils dokumentierten Primärenergiebedarfs einzelner Länder fest[79]. Auf eine diesbezügliche E-Mail-Anfrage des Institutes antwortete die UNSD:

> „[...] *Our mission here is to collect official energy statistics of the countries of the world and publish the data – we do not really have the capacity to compare data published at different places so I could not offer an answer to your question.*"

In Fällen wie diesen ist durch technische Möglichkeiten einer Zusammenführung von Daten - sei sie nun virtuell oder materiell - noch wenig gewonnen. Der Umgang mit solchen oder vergleichbaren Diskrepanzen in Daten unterschiedlicher Quellen und ihre geeignete Auflösung setzt sorgfältige Analysen durch entsprechende Experten voraus – eine Aufgabe, die heute von Computern unterstützt, jedoch keinesfalls von diesen autonom ausgeführt werden kann.

[77] http://unstats.un.org/unsd/default.htm
[78] http://www.iea.org
[79] Potsdam-Institut für Klimafolgenforschung, Dept. Global Change and Social Systems, interner Praktikumsbericht, 2003, unveröffentlicht.