

Operative Metadaten sind u. a.:

- *Systemstatistiken* für die Ressourcenplanung und Optimierung, d. h. Anfragemuster oder nutzer- bzw. gruppenspezifisches Nutzungsverhalten,
- Informationen über *Scheduling*, *Logging* und *Jobausführung* des DWH,
- Regeln und Funktionen für *Nachladen* und *Archivierung*.

Den **domänenspezifischen Metadaten** werden u. a. zugerechnet:

- *Informationsmodelle* und *konzeptuelle Schemata* (→ 2), die der implementierungsunabhängigen Dokumentation dienen,
- Organisations- bzw. branchenspezifische Begriffswerke (*Vokabulare*, *Terminologien* und *Taxonomien*),
- *Abbildungen* zwischen diesen drei Begriffswerken und den korrespondierenden Elementen im DWH,
- Informationen über Organisationsstrukturen und Geschäftsprozesse,
- *Konzeptionelle Beschreibungen* von Berichten, Anfragen, Kennzahlen,
- Angaben über die *Datenqualität*.

Das Metadaten-Repository kommuniziert mit den anderen DWS-Komponenten, die entweder Metadaten anfordern (z. B. Schemabeschreibungen) oder aber ihrerseits erzeugte Metadaten im Repository ablegen (z. B. Zugriffsstatistiken).

Typischerweise existieren in der Praxis neben einem zentralen Repository bei den einzelnen Werkzeugen lokale Datenhaltungskomponenten, in denen Metainformationen abgelegt werden.

14.2 Multidimensionale Datenmodelle

Für die Einführung statischer und dynamischer Konzepte multidimensionaler Datenmodelle soll das folgende Beispiel-Szenario dienen: In einem Unternehmen werden die Verkaufszahlen von Produkten pro Tag und Filiale analysiert; relevante Zeiteinheiten neben dem Tag sind Woche, Monat, Quartal und Jahr. Die Produkte sollen einerseits zu Produktgruppen, andererseits zu Marken und Herstellern zusammengefasst werden; Filialen lassen sich immer einer Stadt zuordnen, diese einer Region und diese wiederum einem Land.

14.2.1 Statische Aspekte

Grundbegriffe

Hauptcharakteristikum multidimensionaler Datenmodelle (/14.4/, /14.10/) ist die Klassifikation in **quantifizierende und qualifizierende Daten**. **Fakten**

sind dabei Datenobjekte, die sowohl *quantifizierende* als auch *qualifizierende* Eigenschaften besitzen. Die *quantifizierenden* Eigenschaften beinhalten für die Organisation *relevante Daten*, die während einer Datenanalyse weitergehend untersucht werden können. *Qualifizierende* Eigenschaften dienen der *näheren Beschreibung der quantifizierenden Eigenschaften*, wodurch diese eine Bedeutung erhalten.

Ein Fakt setzt sich aus einem oder mehreren *Faktattributen* (synonym *Kennzahlen* oder *Maßzahlen*) zusammen, die *meist numerisch* sind und den *quantifizierenden Aspekt bestimmen*. Dagegen beschreiben *Dimensionen* den *qualifizierenden Aspekt von Fakten*. Sie beantworten dabei typischerweise Fragen wie „Wo, Wann, Warum, ... ist der Fakt aufgetreten?“.

- *Beispiel:* In der Verkaufswelt ist Verkaufszahl ein Fakt, die konkrete Anzahl verkaufter Produkte ist eine Kennzahl, und das Produkt (Was wurde verkauft?), die Filiale (Wo wurde es verkauft?) und der Tag (Wann wurde es verkauft?) stellen Dimensionen dar.

Die *Anzahl der Dimensionen* eines Faktis wird als seine *Dimensionalität* bezeichnet. Die *um einen Fakt angeordneten Dimensionen* spannen einen (multidimensionalen) Datenraum auf, der *Datenwürfel* (data cube) genannt wird. Bei bis zu dreidimensionalen Fällen wie im Beispiel lässt sich dieser Würfel grafisch gut veranschaulichen (→ Bild 14.3).

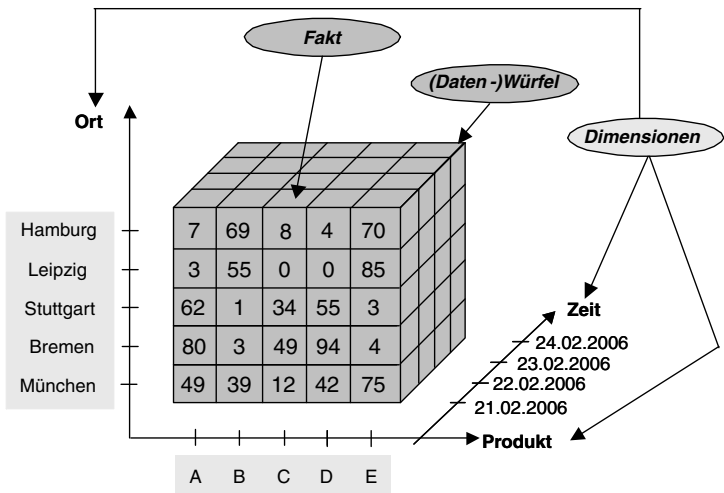


Bild 14.3 Statische Aspekte multidimensionaler Datenmodelle

Hierarchie und Verdichtung

Weil multidimensionale Datenschemata einen analyseorientierten Charakter besitzen, werden die Daten auf den Dimensionen im Hinblick auf diese Analysen zusammengefasst. Eine solche Zusammenfassung wird als **Hierarchieebene** (auch *Aggregations-* oder *Verdichtungsebene*) bezeichnet. Eine Menge **aufeinander aufbauender Hierarchieebenen** heißt **Dimensionshierarchie** (auch: *Hierarchie*, *Verdichtungspfad*). Das **Zusammenfassen von Daten entlang einer Hierarchie** nennt man **Verdichtung** (auch: *Gruppierung*, *Aggregation*). **Dieses Zusammenfassen** erfolgt mittels einer Berechnungsvorschrift, die entsprechend als **Verdichtungs-, Gruppierungs- oder Aggregationsfunktion** bezeichnet wird.

Innerhalb einer Dimension kann es Fälle geben, in denen auf eine Hierarchieebene alternativ mehrere andere folgen, indem aufgrund verschiedener Merkmale verdichtet wird. In diesem Falle spricht man von **multiplen Hierarchien** oder **Mehrfachhierarchien**. Werden **verzweigende Pfade innerhalb der Hierarchie wieder zusammengeführt**, so spricht man auch von **alternativen Verdichtungspfaden**. Bild 14.4 zeigt für das Beispiel eine einfache Hierarchie auf der Ortsdimension und eine multiple Hierarchie auf der Produktdimension.

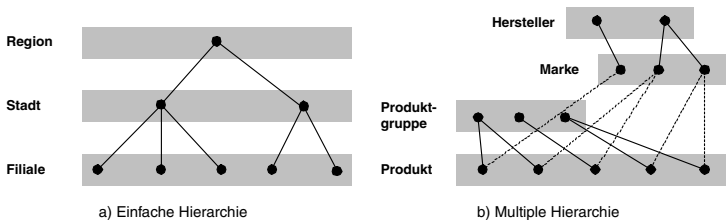


Bild 14.4 Einfache und multiple Hierarchie

Den **Verdichtungsgrad von Daten** innerhalb einer Hierarchie bezeichnet man als **Granularität**. Hierbei besitzen die *Detaildaten* den niedrigsten Verdichtungsgrad bzw. die feinste Granularität, zusammengefasste Daten haben entsprechend einen höheren Verdichtungsgrad und damit eine gröbere Granularität.

Unbalancierte Hierarchie

Es ist möglich, dass bei der Zuordnung von Elementen einer **Hierarchieebene zur nächsthöheren** (oder nächstniedrigeren) Ebene **nicht immer zugehörige Elemente** existieren. In diesem Fall ergibt sich auf Instanzebene ein unbalancierter Baum. Man spricht dann von einer **unbalancierten Hierarchie**.

- ❑ *Beispiel:* Einige Bundesländer haben Bezirke als untergeordnete Instanz, andere nicht. Dementsprechend werden Landkreise entweder Bezirken (und diese dann Bundesländern) oder direkt Bundesländern zugeordnet (→ Bild 14.5).

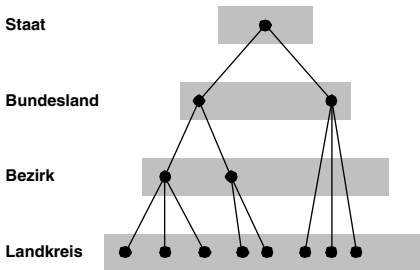


Bild 14.5 Unbalancierte Hierarchie

Anteilige Verrechnung

In /14.24/ wird die Problematik der **anteiligen Verrechnung** vorgestellt, die durch Zuordnung eines Elementes einer Hierarchieebene zu mehreren Elementen der nächsthöheren Ebene mittels einer Berechnungsvorschrift entsteht.

- ❑ *Beispiel:* Eine Woche kann nicht eindeutig einem Kalenderjahr zugeordnet werden, so dass in der Zeit-Hierarchie in Bild 14.6a eine Woche auf zwei Jahre aufgeteilt wird.

Bei einer Verdichtung der Daten muss berücksichtigt werden, dass **ein Wert nicht mehrfach zu 100 % in die Berechnung einbezogen** werden darf. Die herkömmliche Definition einer Dimensionshierarchie mit – auf Instanzebene – eindeutig identifizierbaren Elternknoten ist somit nicht verwendbar.

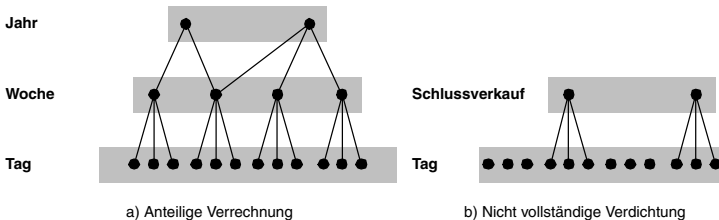


Bild 14.6 Anteilige Verrechnung und nicht vollständige Verdichtung

Nicht vollständige Verdichtung

Ebenso können Fälle auftreten, in denen **nicht alle Instanzen einer Hierarchieebene an der Verdichtung teilnehmen** (→ Bild 14.6b). In diesem Falle

spricht man von einer **nicht vollständigen Verdichtung**. Beim Navigieren entlang einer solchen Verdichtung wird anschaulich die Datenbasis um die nicht an der Verdichtung teilnehmenden Instanzen „ausgedünnt“.

- ▶ *Hinweis:* Hierbei ist insbesondere zu beachten, dass ein Hierarchiepfad, der eine nicht vollständige Verdichtung beinhaltet, i. Allg. nicht wieder mit anderen Hierarchieebenen zusammengeführt werden darf, weil dann aufgrund der zuvor verlorenen Daten falsche Werte zustande kommen.

Aggregierbarkeit

Beim Navigieren entlang der Verdichtungspfade werden die Daten entsprechend einer Verdichtungsoperation verrechnet. Hierbei ist nicht jede Operation für jede Kennzahl anwendbar.

- ❑ *Beispiel:* In einer meteorologischen Datenbank erhält man bei Addition der Kennzahl Temperatur bzgl. der Dimension Ort falsche Werte.

Ein Überblick über diese Problematik findet sich in /14.19/, /14.24/. Die **Eigenschaft einer Kennzahl**, bzgl. einer Dimension **bestimmte Verdichtungsoperatoren zu besitzen**, wird als **Aggregierbarkeit, Additivität oder Summierbarkeit** bezeichnet.

14.2.2 Dynamische Aspekte

Unter den **dynamischen Aspekten** multidimensionaler Datenmodelle werden Operationen auf den statischen Strukturen verstanden.

Häufigste Operation ist das **Wechseln zwischen Hierarchieebenen**, das als **Drilling** bezeichnet wird. Das **Wechseln auf eine gröbere Hierarchieebene** heißt **Roll Up**, **die inverse Operation** – die Verfeinerung der Hierarchieebene – wird als **Drill Down** bezeichnet.

- ❑ *Beispiel:* In Bild 14.7 wird ein Roll Up bzw. Drill Down auf der Zeit-Dimension durchgeführt. Dabei wird zwischen den Hierarchieebenen Quartal und Monat gewechselt, Verdichtungsoperation ist die Addition. So wurde das Produkt in der untersten Zeile des Würfels im ersten Quartal 49-mal verkauft, die Aufteilung auf die Monate erkennt man nach Durchführen der Drill-Down-Operation (Januar 12, Februar 25, März 12).

Bild 14.8 zeigt das **Rotieren oder Pivotisieren**, **bei dem der Datenwürfel in eine für die aktuelle Datenanalyse angemessene Position bewegt wird**.

Schließlich ist auch das benutzergesteuerte Explorieren des Datenwürfels eine wichtige multidimensionale Operation. Durch Selektion der Dimensionselemente werden hierbei die sichtbaren Daten auf einen Teilwürfel (*Dice*) oder eine Scheibe (*Slice*) eingeschränkt.

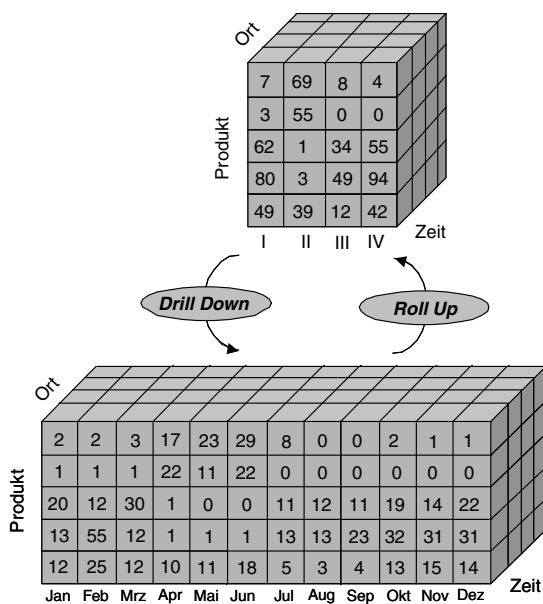


Bild 14.7 Roll Up und Drill Down

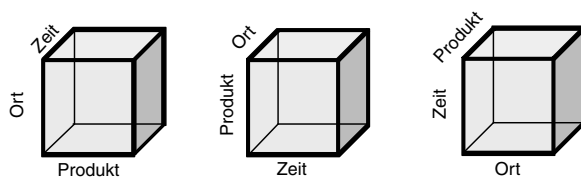
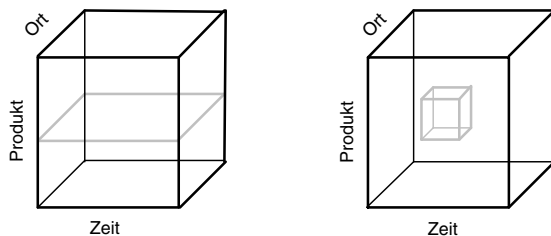


Bild 14.8 Rotation



a) Slice-Operation

b) Dice-Operation

Bild 14.9
Slice und Dice

- ❑ *Beispiel:* Durch Einschränkung auf der Produkt-Dimension wird eine Slice-Operation (→ Bild 14.9a), durch Einschränkung auf allen drei Dimensionen eine Dice-Operation (→ Bild 14.9b) ausgeführt.

14.3 Speicherung und Schemagestaltung

Die im letzten Abschnitt konzeptionell beschriebenen **multidimensionalen Schemata** können in relationalen oder in multidimensionalen Datenbanken oder auch spaltenorientiert gespeichert werden.

14.3.1 Relationale Speicherung

In einem **Schneeflockenschema** (snowflake schema) wird jede **Hierarchieebene** durch eine **Tabelle repräsentiert**, **Verdichtungspfade** lassen sich durch **1:N-Beziehungen** zwischen diesen Dimensionstabellen realisieren. **Im Zentrum** eines Schneeflockenschemas steht die **Fakentabelle** (→ 14.2.1), deren Spalten neben den Fakten durch die Fremdschlüsseinträge der untersten

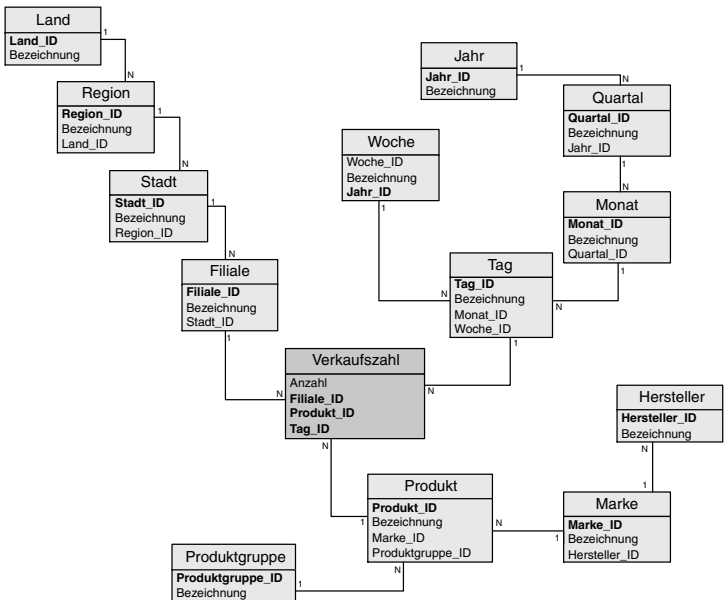


Bild 14.10 Schneeflockenschema

Ebene einer jeden Dimension gebildet werden. Bild 14.10 zeigt das Schneeflockenschema des Verkaufsbeispiels.

In einem **Sternschema** (*star schema*) wird jede Dimension durch nur eine Tabelle dargestellt, d. h., das Sternschema entsteht aus einem Schneeflockenschema durch Denormalisierung der Dimensionstabellen. Bild 14.11 zeigt das Beispiel als Sternschema.

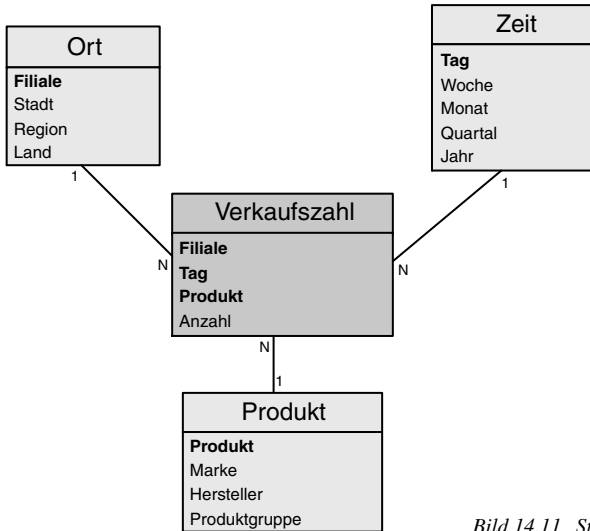


Bild 14.11 Sternschema

Anfragen auf dem Sternschema sind performanter, da weniger Verbundoperationen durchzuführen sind. Das Problem möglicher Anomalien durch Redundanzen ist dadurch entschärft, dass die Daten kontrolliert über den ETL-Prozess (→ 14.1.2) eingefügt werden.

- **Hinweis:** In der Praxis sind abhängig vom OLAP-Server beide Schemaformen gängig. Außerdem sind Mischformen und Erweiterungen, wie das sog. Starflake-Schema, anzutreffen.

14.3.2 Multidimensionale Speicherung

Bei der **multidimensionalen Speicherung** wird der mehrdimensionale Würfel in ein eindimensionales Feld überführt, was man als **Linearisierung** bezeichnet. Dafür wird der Würfel anschaulich in Scheiben geschnitten und die einzelnen Zellen nacheinander in das Feld abgebildet, wobei es verschiedene Formen der Traversierung gibt (→ Bild 14.12).

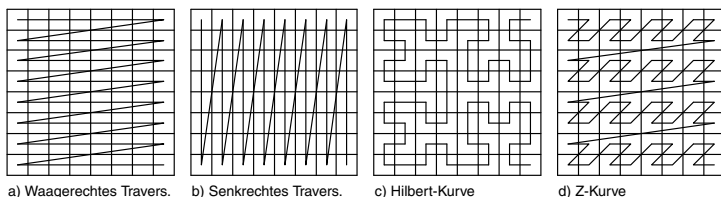


Bild 14.12 Formen der Linearisierung

Vorteil der multidimensionalen gegenüber der relationalen Speicherung ist die größere Performance. Dem steht aber ein hoher Speicherplatzverbrauch gegenüber, da für das Feld Platz in der Größe des gesamten Würfels – und nicht nur der tatsächlich besetzten Zellen – allokiert werden muss. Außerdem ist nach dem Einfügen neuer Daten die gesamte Linearisierung erneut durchzuführen.

Auf relationaler Technologie basierende Systeme werden als **ROLAP** (relationales OLAP), auf multidimensionaler Speicherung aufbauende Systeme als **MOLAP** (multidimensionales OLAP) bezeichnet. In der Praxis ist häufig der **HOLAP-Ansatz** (hybrides OLAP) anzutreffen: Generell werden alle Daten in einem relationalen Schema gespeichert und wichtige Ausschnitte aus dem Datenwürfel, auf die häufig zugegriffen wird, hält das System redundant in multidimensionaler Form vor.

14.3.3 Spaltenorientierte Speicherung

Während in einer relationalen Datenbank die Datensätze standardmäßig Zeile für Zeile abgespeichert werden, werden bei der spaltenorientierten Speicherung die Werte einer Spalte fortlaufend in sog. **Column Stores** abgespeichert [14.1]/, [14.21]. Bild 14.13 stellt diese beiden Ansätze gegenüber.

Die **zeilenorientierte Speicherung** ist in der OLTP-(Online Transaction Processing) Verarbeitung vorteilhaft, weil hier typischerweise einzelne Datensätze angefragt und verarbeitet werden (→ Tabelle 14.1). **Spaltenorientierte Speicherung** kann in analytischen Applikationen in einem DWH-System vorteilhaft sein, weil in diesen Anwendungen häufig eine größere Datenmenge angefragt wird und auf dieser eine statistische Berechnung stattfindet.

Spaltenorientierte Speicherung hat bei sehr breiten Faktentabellen einen deutlichen Vorteil, wenn in einer Anfrage nur wenige Attribute benötigt werden.

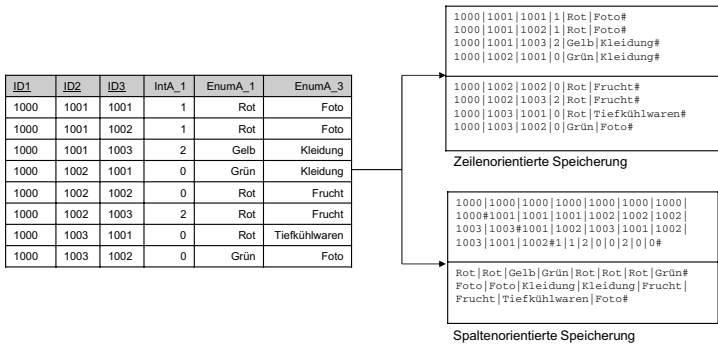


Bild 14.13 Zeilen- und spaltenorientierte Speicherung

- ❑ *Beispiel:* Eine Faktentabelle habe 30 Attribute, die Daten werden zeitlich fortlaufend abgespeichert, es befinden sich Daten von einem Jahr in der Tabelle. In einer Anfrage seien nun die drei Faktattribute 5, 9 und 24 relevant, in der WHERE-Klausel werde bzgl. der Zeitdimension die Anzahl der Datensätze auf zwei zusammenhängende Monate eingeschränkt. Bei zeilenorientierter Speicherung werden die gesamten Datensätze gelesen, was im Beispiel ungefähr 17 % des Gesamtvolumens betrifft. Bei zeilenorientierter Speicherung werden nur die relevanten Daten gelesen, was in diesem Beispiel ca. 2 % der Daten entspricht.

Daten in **zeilenorientierten Systemen** lassen sich nur **schlecht komprimieren**, weil in einem Datensatz typischerweise Werte vieler verschiedener Datentypen gespeichert sind. Durch die **Spaltenorientierung** hingegen stehen die sehr ähnlichen Werte einer Spalte auch physisch nah beieinander und bieten eine **gute Basis für Kompression**. Dabei können pro Spalte (oder sogar die Spalte noch einmal in Teilabschnitte zerlegt) verschiedene Kompressionsverfahren verwendet werden.

Bild 14.14 zeigt exemplarisch drei im Umfeld von Datenbanken populäre Kompressionstechniken. Bei der ganz links dargestellten **Wörterbuchkompression** (/14.1/, /14.23/) werden **statt des Volltextes** als Zeichenkette kurze **binäre Codes** abgespeichert. In der Mitte ist die **Lauf längencodierung** /14.9/ dargestellt, die sich bei Werten mit wenigen Zeichen und langen Sequenzen des gleichen Zeichens anbietet, **anstelle der gesamten Sequenz** wird jeweils die **aufeinander folgende Anzahl an Werten** gespeichert. Ganz rechts in Bild 14.14 ist schließlich die **Delta-Codierung** /14.8/ zu sehen, die bei Sequenzen steigender (oder fallender) ganzzahliger Werte Anwendung findet. **Anstatt die einzelnen Werte zu speichern**, wird jeweils **die Differenz zum Vorgänger** angegeben. Zu jedem der drei Verfahren ist jeweils der Kompressionsfaktor

(KF) als Quotient aus unkomprimierter und komprimierter Speicherung angegeben. Der eingesparte Speicherplatz ergibt sich, indem der Kehrwert des KF von 1 subtrahiert wird.



	Wörterbuchkompression	Laufänglencodierung	Delta-Codierung
Unkomprimierte Speicherung	Kleidung Kleidung Frucht Foto Foto Kleidung Frucht Tiefkühlwaren Tiefkühlwaren Frucht Frucht Frucht Kleidung Foto Tiefkühlwaren Tiefkühlwaren Tiefkühlwaren		121355 121356 121358 121358 121361 121362 121364 121366
Komprimierte Speicherung	Wörterbuch: Kleidung 00 Frucht 01 Tiefkühlwaren 10 Foto 11 Daten: 00 00 01 11 11 00 01 10 10 01 01 01 00 11 10 10 10		... 1 2 3 1 2 2
Kompressionsfaktor	Unkomprimierte Speicherung: 17 Einträge, 139 Zeichen á 1 Byte Komprimiert: Pro Eintrag 2 Bit KF = 32.7	Unkomprimiert: 18 Zeichen Komprimiert: 8 Zeichen KF = 2.25	Unkomprimiert: 7 Integer-Werte á 4 Byte Komprimiert: 7 Byte-Werte KF = 4
Speicherplatzersparnis	97 %	56 %	75 %

Bild 14.14 Kompressionsverfahren in Datenbanken

- **Hinweis:** Bei dieser Berechnung wird der Platzbedarf für das Wörterbuch vernachlässigt. Dies führt im Beispiel zu einer erheblichen Steigerung des KF (sonst 3.2). Bei einer entsprechend großen Anzahl an Einträgen ist diese Annahme aber durchaus realistisch.

Generelle Aussagen über erreichbare Kompressionsraten sind schwer vorzunehmen. Die **Kompressionsrate hängt wesentlich von den Faktoren Datentyp, Anzahl unterschiedlicher Werte einer Spalte, Verteilung der Werte und Anzahl von Nullwerten ab.**

Neben der Kompression ist das Bilden von **Metadaten** eine wichtige Technik, die in spaltenorientierten Systemen genutzt wird. Bild 14.15 zeigt diese Vorgehensweise: Eine **Spalte wird in Gruppen zerlegt** und für jede Gruppe werden **statistische Informationen abgelegt** (z. B. Erstellung eines Histogramms oder Speicherung der Anzahl von Nullwerten) bzw. **Werte vorberechnet** (z. B. Minimum, Maximum und Summe). Diese statistischen Informationen sind im Verhältnis zu den Rohdaten relativ klein und können im Hauptspeicher gehalten werden. Bei späteren Anfragen können sie genutzt werden, einige Anfragen lassen sich hiermit beantworten und das teure **Lesen und Dekomprimieren der gespeicherten Daten entfällt.**

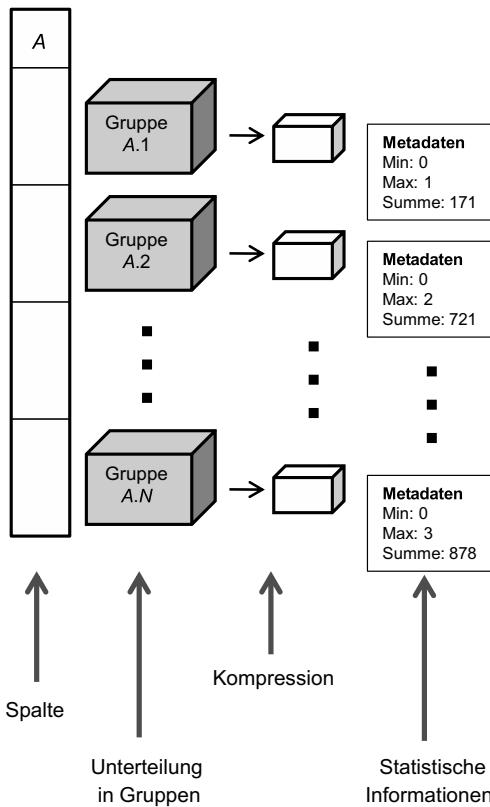


Bild 14.15
Statistische Informationen und Metadaten

14.4 Erweiterung relationaler Datenbanken

Aufgrund der besonderen Eigenschaften von DWH (z. B. großes Datenvolumen, besondere Ladeanforderungen, komplexe Anfragetypen) kann es bei Verwendung konventioneller relationaler Datenbanken zu Performance-Engpässen kommen. Begegnet worden ist diesem Problem von den führenden DBMS-Anbietern (u. a. Oracle, IBM und Microsoft) durch Erweiterungen zur Beschleunigung von Anfragen bzw. Einfügeoperationen. Die wichtigsten Erweiterungen werden in diesem Abschnitt vorgestellt.