

## 3. Multidimensionales Datenmodell

---

- Grundbegriffe
  - Dimensionen, Fakten/Kennzahlen, Würfel
- Analyseoperationen
  - Drill-Down, Roll-Up, Slice und Dice
- Notationen zur konzeptuellen Modellierung
  - ME/R, ADAPT, graphbasierte Ansätze
- Relationale Speicherung
  - Star-Schema, Snowflake-Schema
- Multidimensionale Speicherung

## Motivation

---

- Datenmodell ausgerichtet auf Unterstützung der Analyse
- Datenanalyse im Entscheidungsprozess
  - Betriebswirtschaftliche Kennzahlen (Erlöse, Gewinne, Verluste, etc.) stehen im Mittelpunkt
  - Betrachtung der Kennzahlen aus unterschiedlichen Perspektiven (zeitlich, regional, produktbezogen) → *Dimensionen*
  - Unterteilung der Auswertedimensionen möglich (Jahr, Quartal, Monat) → *Hierarchien* oder *Konsolidierungsebenen*

## Motivation /2

---

- Verfügbare Informationen
  - Qualifizierend
    - Repräsentiert durch „Kategorienattribute“
    - Daten zur Nutzung als Navigationsraster („Drill-Pfade“)
    - Modelliert als Begriffshierarchien im Rahmen von Dimensionen
  - Quantifizierend
    - Bilden Gegenstand der Auswertung („Summenattribute“)
    - Zellen eines Würfels, mit Dimensionen als Kanten

## Dimensionen

---

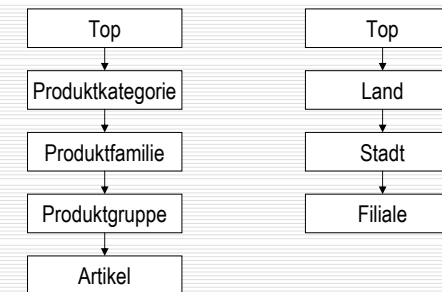
- Dimension:
  - beschreibt mögliche Sicht auf die assoziierte Kennzahl
  - endliche Menge von  $n$  ( $n \geq 2$ ) Dimensionselementen (Hierarchieobjekten), die eine semantische Beziehung aufweisen
  - dienen der orthogonalen Strukturierung des Datenraums
- Beispiele: Produkt, Geographie, Zeit

## Hierarchien in Dimensionen

- Dimensionselemente:
  - Knoten einer Klassifikationshierarchie
  - Klassifikationsstufe beschreibt Verdichtungsgrad
- Darstellung von Dimensionen über Klassifikationsschema (Schema von Klassifikationshierarchien)
- Formen:
  - einfache Hierarchien
  - parallele Hierarchien

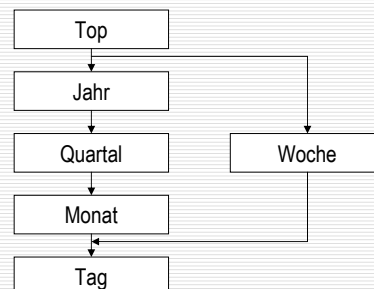
## Einfache Hierarchien

- Höhere Hierarchieebene enthält die aggregierten Werte genau einer niedrigeren Hierarchiestufe
- Oberster Knoten: *Top* → enthält Verdichtung auf einen einzelnen Wert der Dimension



## Parallele Hierarchien

- Innerhalb einer Dimension sind zwei verschiedene Arten der Gruppierung möglich
- Keine hierarchische Beziehung in den parallelen Zweigen
- Parallelhierarchie → Pfad im Klassifikationsschema (*Konsolidierungspfad*)



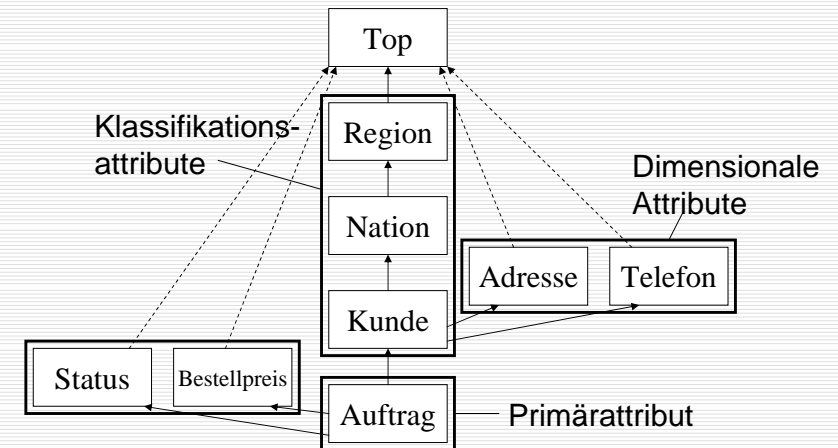
## Schema einer Dimension

- Schema einer Dimension  $D$ 
  - Partiiell geordnete Menge von Kategorienattributen  $(\{D_1, \dots, D_n, Top_D\}; \rightarrow)$ 
    - Generisches max. Element  $Top_D$
    - Funktionale Abhängigkeit →
  - $Top_D$  wird von allen Attributen funktional bestimmt:  $\forall i, 1 \leq i \leq n: D_i \rightarrow Top_D$
  - Genau ein  $D_i$ , das alle anderen Kategorieattribute bestimmt
    - Gibt feinste Granularität einer Dimension vor  $\exists i, 1 \leq i \leq n, \forall j, 1 \leq j \leq n, i \neq j, D_i \rightarrow D_j$

## Kategorienattribute

- Inhaltliche Verfeinerung durch unterschiedliche Rollen
  - Primärattribut
    - Kategorienattribut, das alle anderen Attribute einer Dimension bestimmt
    - Definiert maximale Feinheit
    - Beispiel: „Auftrag“
  - Klassifikationsattribut
    - Element der Menge, die mehrstufige Kategorisierung (Klassifikationshierarchie) bilden
    - Beispiel: „Kunde“, „Nation“, „Region“
  - Dimensionales Attribut
    - Element der Menge der Attribute, die vom Primärattribut oder einem Klassifikationsattribut bestimmt werden und nur  $Top_D$  bestimmen
    - Beispiel: „Adresse“, „Telefon“

## Struktur einer Dimension: Beispiel



## Kennzahlen/Fakten

- Kennzahlen/Fakten (engl. *facts*):
  - (verdichtete) numerische Messgrößen
  - Beschreiben betriebswirtschaftliche Sachverhalte
- Fakt: Basiskennzahl
- Kennzahl:
  - aus Fakten konstruiert (abgeleitete Kennzahl)
  - Durch Anwendung arithmetischer Operationen
- Beispiele:
  - Umsatz, Gewinn, Verlust, Deckungsbeitrag

## Fakt: Schema

- Besteht aus
  - Granularität  $G = \{ G_1, \dots, G_k \}$ 
    - Ist Teilmenge aller Kategorienattribute aller im Schema existierenden Dimensionenschemata  $DS_1, \dots, DS_n$ 
      - $\forall i, 1 \leq i \leq k \exists j, 1 \leq j \leq n: G_i \in DS_j$
      - $\forall i, 1 \leq i \leq k \forall j, 1 \leq j \leq k, i \neq j: G_i \not\rightarrow G_j$  (keine funkt. Abhängigkeit zwischen Kategorienattributen einer Granularität)
  - Summationstyp *SumTyp*

# Kennzahl

- Kennzahl  $M$  ist definiert durch
  - Granularität  $G$
  - Berechnungsvorschrift  $f()$
  - Summationstyp  $SumTyp$
  - Berechnung über nichtleerer Teilmenge der im Schema existierenden Fakten
- $M = (G, f(F_1, \dots, F_k), SumTyp)$

# Kennzahl: Berechnungsvorschrift

- Bildung von  $f()$ 
  - Skalarfunktionen
    - $+$ ,  $-$ ,  $*$ ,  $/$ , mod
    - Beispiel:  
Umsatzsteueranteil = Menge \* Preis \* Steuersatz
  - Aggregatfunktionen
    - Funktion  $H()$  zur Verdichtung eines Datenbestandes, indem aus  $n$  Einzelwerten ein Aggregatwert ermittelt wird  
 $H: 2^{dom(X1)} \times \dots \times 2^{dom(Xn)} \rightarrow 2^{dom(Y)}$
    - $SUM()$ ,  $AVG()$ ,  $MIN()$ ,  $MAX()$ ,  $COUNT()$
  - Ordnungsbasierte Funktionen
    - Definition von Kennzahlen auf Basis zuvor definierter Ordnungen
    - Bsp.: Kumulation,  $TOP(n)$

# Summationstyp

- Zuweisung eines Summationstyps charakterisiert erlaubte Aggregationsoperationen
- FLOW
  - Beliebig aggregierbar
  - Beispiel: Bestellmenge eines Artikels pro Tag
- STOCK
  - Beliebig aggregierbar mit Ausnahme temporaler Dimension
  - Beispiele: Lagerbestand, Einwohnerzahl pro Stadt
- VALUE-PER-UNIT (VPU)
  - Aktuelle Zustände, die nicht summierbar sind
  - Zulässig nur:  $MIN()$ ,  $MAX$ ,  $AVG()$
  - Beispiele: Wechselkurs, Steuersatz

# Summierbarkeit

	FLOW	STOCK: Aggregation über temporale Dimension?		VPU
		nein	ja	
MIN/MAX	+	+		+
SUM	+	-	+	-
AVG	+		+	+
COUNT	+		+	+

## Weitere Eigenschaften

### □ Disjunktheit

- Ein konkreter Wert einer Kennzahl geht exakt einmal in Ergebnis ein
- Bsp.: Studierende im Grundstudium

Studierende	1999	2000	2001	Gesamt
Informatik	15	17	13	<b>28</b>
BWL	10	15	11	<b>21</b>
<b>Gesamt</b>	<b>25</b>	<b>32</b>	<b>24</b>	<b>49</b>

## Weitere Eigenschaften /2

### □ Vollständigkeit

- Kennzahlen auf höherer Aggregationsebene lassen sich komplett aus Werten tieferer Stufen berechnen

Restaurants	2001	2002
Halle	45	46
Magdeburg	52	50
Sonstige	20	22
<b>Gesamt</b>	<b>117</b>	<b>118</b>

## Würfel

- Würfel (engl. *cube*, eigentlich Quader): Grundlage der multidimensionalen Analyse
- Kanten → Dimensionen
- Zellen → ein oder mehrere Kennzahlen (als Funktion der Dimensionen)
- Anzahl der Dimensionen → Dimensionalität
- Visualisierung
  - 2 Dimensionen: Tabelle
  - 3 Dimensionen: Würfel
  - >3 Dimensionen: Multidimensionale Domänenstruktur

## Würfel /2

- Schema  $C$  eines Würfels
  - Menge der Dimensionen(-schemata)  $DS$
  - Menge der Kennzahlen  $M$
- $C = (DS, M) = (\{D^1, \dots, D^n\}, \{M^1, \dots, M^m\})$

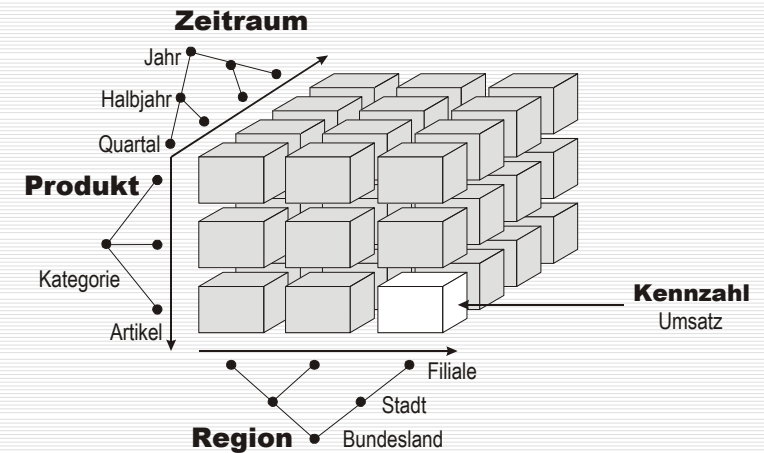
## Orthogonalität

□ In multidimensionalen Schemata gilt Orthogonalität, d.h.

- Keine funktionalen Abhängigkeiten zwischen Attributen unterschiedlicher Dimensionen

$$\forall i, 1 \leq i \leq n, \forall j, 1 \leq j \leq n, i \neq j, \neg \exists k, l: D^i \cdot D_k \rightarrow D^j \cdot D_l$$

## Multidimensionaler Datenwürfel



## Operationen zur Datenanalyse

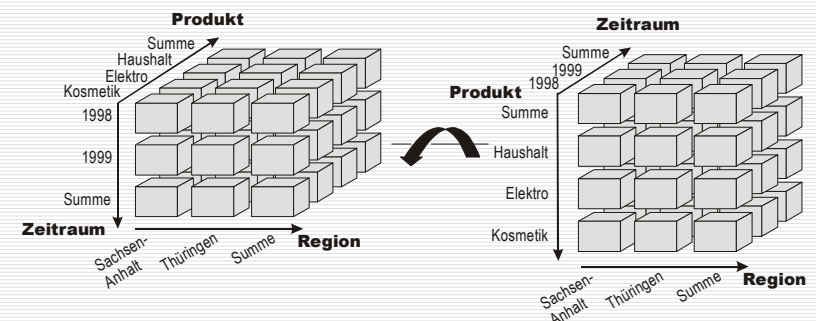
□ OLAP-Operationen auf multidimensionalen Datenstrukturen

□ Standardoperationen

- Pivotierung
- Roll-Up, Drill-Down
- Drill-Across
- Slice, Dice

## Pivotierung/Rotation

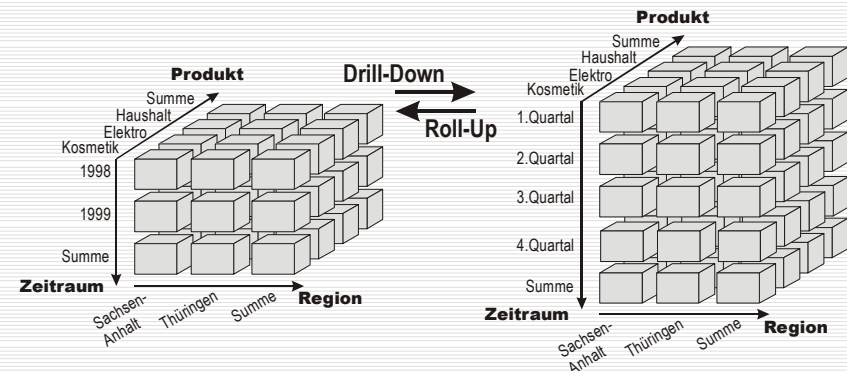
□ Drehen des Würfels durch Vertauschen der Dimensionen → Analyse der Daten aus verschiedenen Perspektiven



## Roll-Up, Drill-Down, Drill-Across

- Roll-Up:
  - Erzeugen neuer Informationen durch Aggregation der Daten entlang des Konsolidierungspfades
  - Dimensionalität bleibt erhalten
  - Beispiel: Tag → Monat → Quartal → Jahr
- Drill-Down:
  - komplementär zu Roll-Up
  - Navigation von aggregierten Daten zu Detail-Daten entlang der Klassifikationshierarchie
- Drill-Across:
  - Wechsel von einem Würfel zu einem anderen

## Roll-Up, Drill-Down

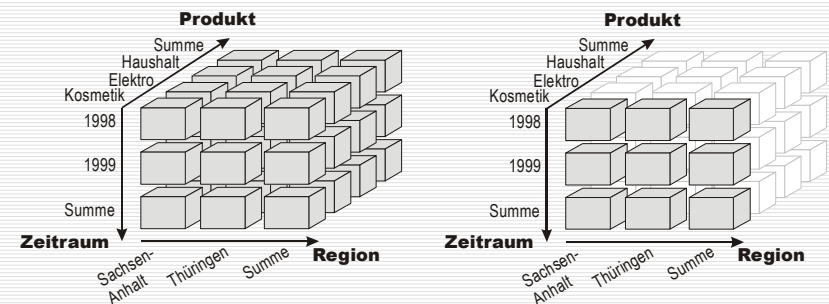


*Bemerkung: im Bsp. fehlt rechts die Ebene 1998*

## Slice und Dice

- Erzeugen individueller Sichten
- Slice:
  - Herausschneiden von „Scheiben“ aus dem Würfel
  - Verringerung der Dimensionalität
  - Beispiel: alle Werte des aktuellen Jahres
- Dice:
  - Herausschneiden einen „Teilwürfels“
  - Erhaltung der Dimensionalität, Veränderung der Hierarchieobjekte
  - Beispiel: die Werte bestimmter Produkte oder Regionen

## Slice



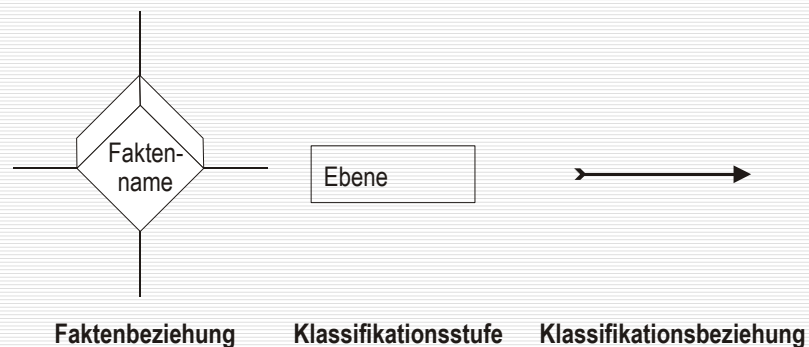
## Konzeptuelle Modellierung

- Konzeptuelle Modellierung:
  - formale Beschreibung des Fachproblems und der im Anwendungsbereich benötigten Informationsstrukturen
- Probleme konventioneller Entwurfstechniken (ER, UML):
  - Unzureichende Semantik für multidimensionales Datenmodell
  - *hier*: Verzicht auf universelle Anwendbarkeit, stattdessen Konzentration auf Analyse
  - Beispiel: Klassifikationsstufe, Fakt → Entity ?

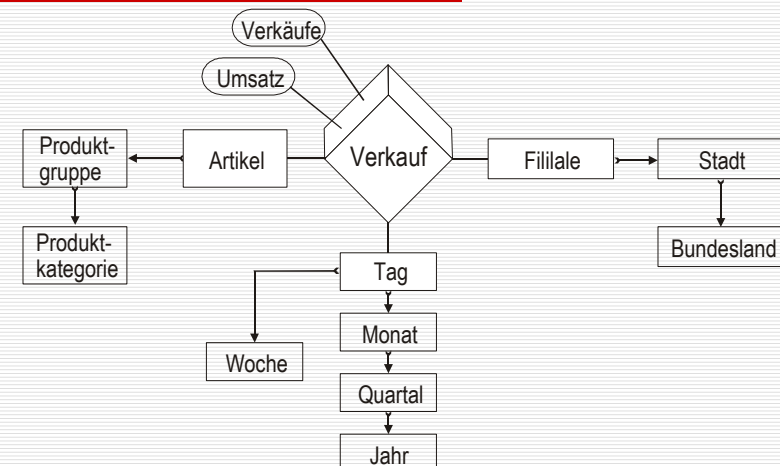
## ME/R-Modell

- Multidimensional Entity/Relationship [Sapia et. al. (LNCS 1552)]
- Erweiterung des klassischen ER-Modells
  - Entity-Menge „*Dimension Level*“ (Klassifikationsstufe)
    - keine explizite Modellierung von Dimensionen
  - n-äre Beziehungsmenge „*Fact*“
    - Kennzahlen als Attribute der Beziehung
  - Binäre Beziehungsmenge „*Classification*“ bzw. „*Roll-Up*“ (Verbindung von Klassifikationsstufen)
    - definiert gerichteten, nicht-zyklischen Graphen

## ME/R-Modell: Notation



## ME/R-Modell: Beispiel



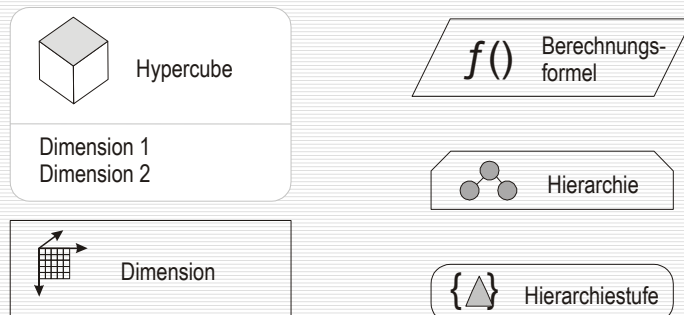
# ADAPT

- Application Design for Analytical Processing Technologies (Bulos)
- neue Entwicklung für multidimensionale Datenmodellierung
- Beschreibung sämtlicher Metadaten-Objekte
- Unterstützung von Berechnungsvorschriften
- teilweise Werkzeugunterstützung (CASE, Visio, etc.)
- keine formale Grundlage

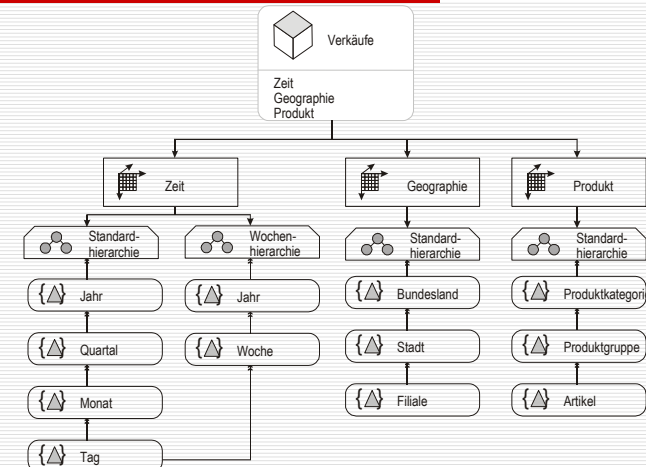
# ADAPT: Elemente

- Hypercube:
  - multidimensionale Datenstruktur
  - enthält nur eine Kennzahl
  - Assoziationen zu beliebig vielen Dimensionen
- Dimension:
  - beschreibt Dimension
  - bestehend aus Hierarchiestufen, Dimensionselementen, Attributen
- Hierarchie:
  - Eindeutiger Konsolidierungspfad

# ADAPT: Notation



# ADAPT: Beispiel



# Graphbasierte Modellierungsansätze

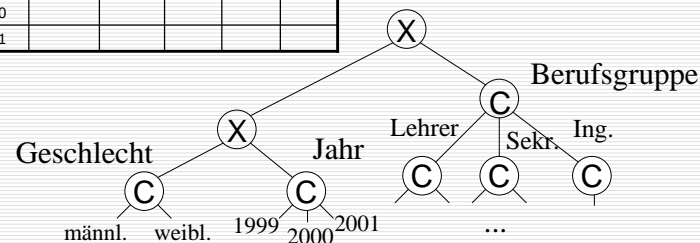
- Idee: Beschreibung konzeptioneller Schemata in Form von Graphen
  - Ausgangspunkt: „statistische“ Tabelle mit Kopfzeile und seitlicher Gliederung, teilweise Summenbildung über Zeilen und Spalten
  - Repräsentation der kategorisierenden Daten sowie der Attributbeziehungen durch *gerichteten, azyklischen Graphen*
  - Navigationshilfe für Benutzer

# Graphstruktur

- Kanten: Beziehungen der Attribute
- Knoten: unterschiedliche Semantik (in Abhängigkeit von konkreter Notation)
  - Basistypen:
    - Kategorien- (Cluster) Knoten (C)
      - Repräsentiert Gruppierung einzelner Elemente gemäß Kategorienhierarchie
    - Kreuzprodukt-Knoten (X)
      - Aufspannen eines mehrdimensionalen Adressierungsraumes mit Hilfe der Kategorienattribute über C-Knoten

# Graphbasiertes Schema: Beispiel

		Lehrer		Sokr.		Ing.
		Grundschule	Realschule	Chef-sekr.	Sokr.	Bau-Ing.
männl.	1999					
	2000					
	2001					
weibl.	1999					
	2000					
	2001					

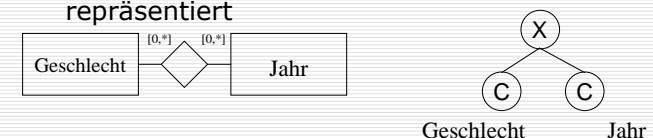


# Zuordnungsregeln

- Funktionale Abhängigkeit
  - Wird direkt durch Kante zwischen beiden C-Knoten repräsentiert



- N:M-Beziehung
  - Wird indirekt durch Einführung eines X-Knotens repräsentiert



## Weitere Knotentypen

- Terminale Knoten (*t<sub>n</sub>-Knoten*)
  - Repräsentation eines der möglichen Werte aus dem Wertebereich des übergeordneten Kategorieattributes
  - Beispiel: „männlich“, „weiblich“ für „Geschlecht“
- Summenknoten (*S-Knoten*)
  - Explizite Spezifikation des quantitativen Teils eines Objektgraphen (Mehrfachverwendung von Graphen)
  - Beispiel: „mittleres Einkommen“, „Anteil am Gesamteinkommen“ zu X-Knoten
- Topic-Knoten (*T-Knoten*)
  - Repräsentation einer Menge statistischer Objekte
    - Dekomposition statistischer Sachverhalte
    - Logische Verbindung von S-Knoten

## Modellierung der Abstraktion

- Aggregation (*A-Knoten*)
  - Zusammenfassung logisch zusammengehöriger Einzelfakten
  - Beispiel: (Straße, Stadt, Land) zu Wohnort, (PersNr, Name, Wohnort, Beruf) zu Erwerbstätige
- Generalisierung (*G-Knoten*)
  - Definition einer übergeordneten Klasse abstrakter Objekte
  - Beispiel: Erwerbstätige, Erwerbslose zu Erwerbsperson

## Zusammenfassung

- Weitere Notationen:
  - Erweiterungen von ER:
    - Dimensional Fact Modeling
  - Erweiterungen von UML:
    - mUML (multidimensional UML)
  - Graphbasiert:
    - SUBJECT, GRASS, STORM, ADaS, ...
- Zur Zeit kein Standard verfügbar
- Graphbasierte Ansätze zwar mächtig + flexibel, aber kaum verbreitet

## Umsetzung des multidim. Datenmodells

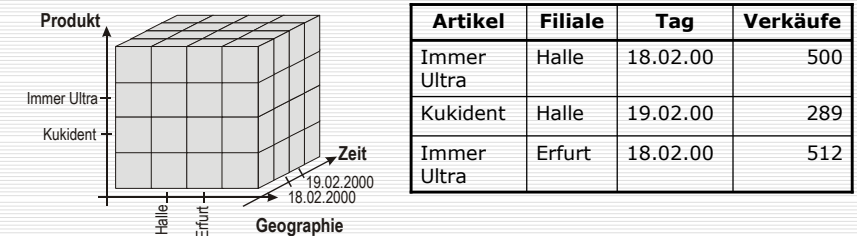
- Multidimensionale Sicht
  - Modellierung der Daten
  - Anfrageformulierung
- Interne Verwaltung der Daten erfordert Umsetzung auf
  - relationale Strukturen (Tabellen) → *ROLAP (relationales OLAP)*
    - Verfügbarkeit, Reife der Systeme
  - multidimensionale Strukturen (direkte Speicherung) → *MOLAP (multidimensionales OLAP)*
    - Wegfall der Transformation
- Aspekte
  - Speicherung
  - Anfrageformulierung bzw. -ausführung

## Relationale Speicherung: Anforderungen

- ❑ Vermeidung des Verlustes anwendungsbezogener Semantik (aus dem multidimensionalen Modell, z.B. Klassifikationshierarchien)
- ❑ effiziente Übersetzung multidimensionaler Anfragen
- ❑ effiziente Verarbeitung der übersetzten Anfragen
- ❑ Einfache Pflege der entstandenen Relationen (z.B. Laden neuer Daten)
- ❑ Berücksichtigung der Anfragecharakteristik und des Datenvolumens von Analyseanwendungen

## Relationale Speicherung: Faktentabelle

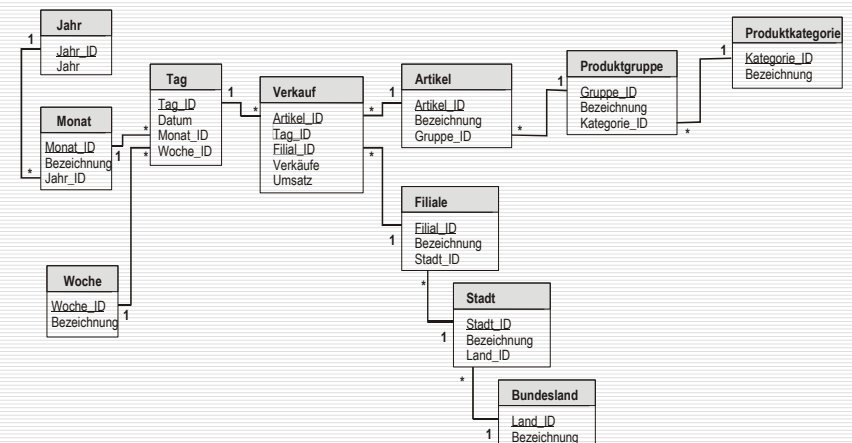
- ❑ Ausgangspunkt: Umsetzung des Datenwürfels ohne Klassifikationshierarchien
  - Dimensionen, Kennzahlen → Spalten der Relation
  - Zelle → Tupel



## Snowflake-Schema

- ❑ Abbildung von Klassifikationen: eigene Tabelle für jede Klassifikationsstufe (z.B. Artikel, Produktgruppe, etc.)
- ❑ Tabelle enthält
  - ID für Klassifikationsknoten
  - beschreibendes Attribut (z.B. Marke, Hersteller, Bezeichnung)
  - Fremdschlüssel der direkt übergeordneten Klassifikationsstufe
- ❑ Faktentabelle enthält (neben Kenngrößen):
  - Fremdschlüssel der jeweils niedrigsten Klassifikationsstufe
  - Fremdschlüssel bilden zusammengesetzte Primärschlüssel für Faktentabelle

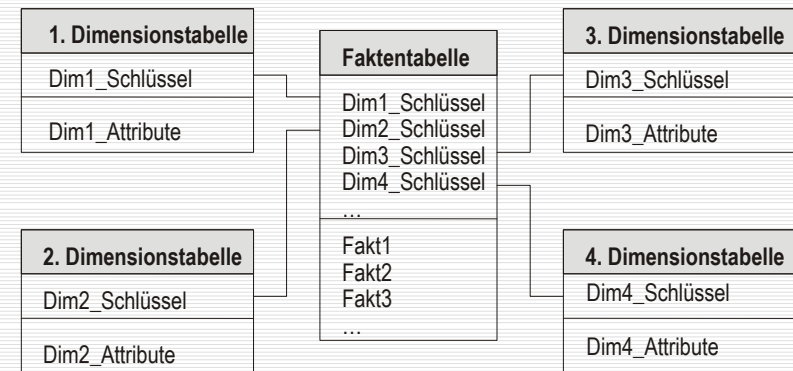
## Snowflake-Schema: Beispiel



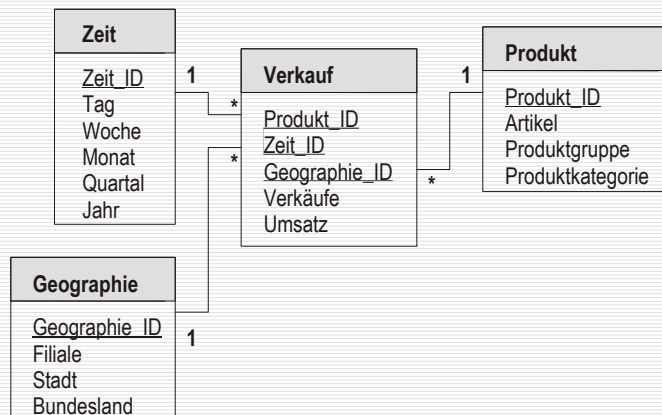
# Star-Schema

- ❑ Snowflake-Schema ist normalisiert: Vermeidung von Update-Anomalien → aber: erfordert Join über mehrere Tabellen!
- ❑ Star-Schema:
  - Denormalisierung der zu einer Dimension gehörenden Tabellen
  - für jede Dimension genau eine *Dimensionstabelle*
  - Redundanzen in der Dimensionstabelle für schnellere Anfragebearbeitung
  - Beispiel: Artikel, Produkt, Produktgruppe etc. als Spalten in einer Tabelle Produkt

# Star-Schema



# Star-Schema: Beispiel



# Star-Schema: Muster

- ❑ Multidimensionales Schema mit  $n$  Dimensionen
  - Dimensionstabellen  $D^1, \dots, D^n$  der Form  $D^i(PA_{i1}, A_{i1}, \dots, A_{ik})$
  - Faktentabelle  $F(PA_{11}, \dots, PA_{n1}, f_{11}, \dots, f_{k1})$
- ❑ Jeder Teil des kompositen Primärschlüssels der Faktentabelle ist Fremdschlüssel zum Primärschlüsselattribut der korrespondierenden Dimension

## CREATE DIMENSION in Oracle

- Fremdschlüsselbedingungen in SQL ausdrückbar
- Aber: funktionale Beziehungen zwischen Attributen innerhalb einer Dimension nicht spezifizierbar
- Oracle-Erweiterung: CREATE DIMENSION
  - „informative“ Zusicherung
  - Korrektheit wird vom DBS nicht überprüft
  - Nutzung beim Query Rewriting über materialisierten Sichten

## CREATE DIMENSION /2

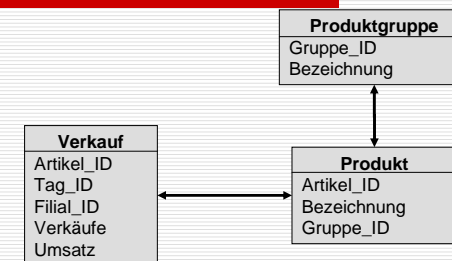
- Beispiel:

```
CREATE DIMENSION ORDER_DIM
  LEVEL ORDER IS ORDERS.ORDERKEY
  LEVEL CUSTOMER IS ORDERS.CUSTOMERKEY
  LEVEL NATION IS ORDERS.NATIONKEY
  LEVEL REGION IS ORDERS.REGIONKEY
HIERARCHY ORDER_HIERARCHY (
  ORDER CHILD OF
  CUSTOMER CHILD OF
  NATION CHILD OF
  REGION )
ATTRIBUTE ORDER DETERMINES
(O_STATUS, O_DATE, ...)
ATTRIBUTE CUSTOMER DETERMINES
(C_NAME, C_ADDRESS, ...);
```

## CREATE DIMENSION /3

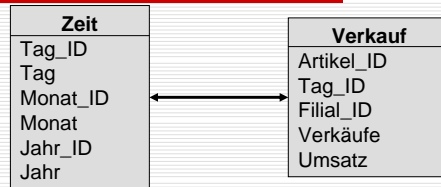
- **LEVEL**
  - definiert Klassifikationsstufen
- **HIERARCHY**
  - Festlegung der Abhängigkeiten der Klassifikationsstufen
- **ATTRIBUTE ... DETERMINES**
  - Definiert Abhängigkeit zwischen Klassifikationsattribut und dimensionalen Attributen

## Beispiel: Snowflake-Schema



```
CREATE DIMENSION s_pg
  LEVEL Produkt IS (Produkt.Artikel_ID)
  LEVEL Produktgruppe IS (Produktgruppe.Gruppe_ID)
HIERARCHY pg_rollup (
  Produkt CHILD OF Produktgruppe)
JOIN KEY (Produkt.Gruppe_ID)
REFERENCES Produktgruppe.Gruppe_ID)
```

## Beispiel: Star-Schema



```

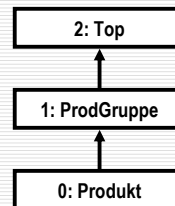
CREATE DIMENSION s_time
  LEVEL Tag IS (Zeit.Tag_ID)
  LEVEL Monat IS (Zeit.Monat_ID)
  LEVEL Jahr IS (Zeit.Jahr_ID)
  HIERARCHY pg_rollup (
    Tag CHILD OF Monat CHILD OF Jahr)
  ATTRIBUTE Tag_ID DETERMINES (Tag)
  ATTRIBUTE Monat_ID DETERMINES (Monat)
  ATTRIBUTE Jahr_ID DETERMINES (Jahr)
    
```

## Star- vs. Snowflake-Schema

- Charakteristika von DW-Anwendungen
  - typischerweise Einschränkungen in Anfragen auf höherer Granularitätsstufe (Join-Operationen)
  - geringes Datenvolumen der Dimensionstabellen im Vergleich zu Faktentabellen
  - seltene Änderungen an Klassifikationen (Gefahr von Update-Anomalien)
- Vorteile des Star-Schemas
  - einfache Struktur (vereinfachte Anfrageformulierung)
  - einfache und flexible Darstellung von Klassifikationshierarchien (Spalten in Dimensionstabellen)
  - effiziente Anfrageverarbeitung innerhalb einer Dimension (keine Join-Operation notwendig)

## Annahmen für Kostenbetrachtungen

- D Dimensionen, je K Klassifikationsstufen plus Top
- Jeder Klassifikationsknoten hat 3 Kinder
  - Level i=K: 1=3<sup>0</sup> Knoten (Top)
  - Level i=K-1: 3=3<sup>1</sup> Knoten
  - Level i=K-2: 9=3<sup>2</sup> Knoten
  - ...
  - Level i=0: Höchste Granularität, 3<sup>K</sup> Knoten

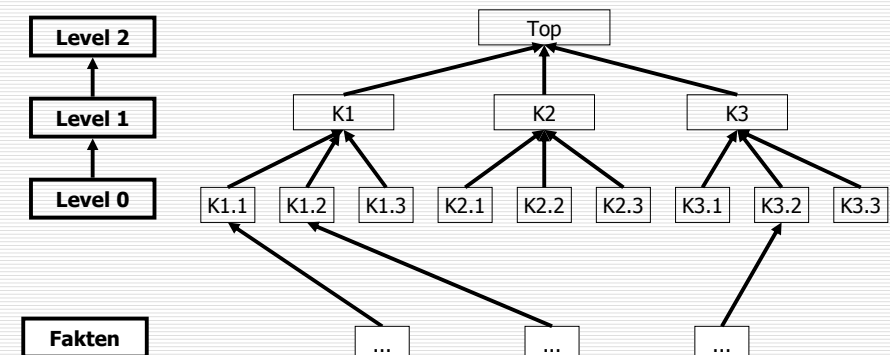


$$\Rightarrow N_D = \sum_{i=0..K} 3^i \text{ Knoten pro Dimension}$$

- M Fakten, gleichverteilt in Dimensionen
- Attribut: b Bytes; Knoten haben nur ID; f Faktattribute

## Volle Klassifikationshierarchie

- Zu jedem Knoten gibt es (gleich viele) Fakten



## Star- vs. Snowflake-Schema /2

- Speicherplatz Snowflake: Ein Tupel pro Klassifikationsknoten

$$((D + f) * M + D * N_D) * b$$

Fremdschlüssel in Faktentabelle →  $f$   
Anz. Faktenattribute →  $M$   
Ein Tupel pro Klassifikationsknoten →  $N_D$

- Speicherplatz Star: Ein Tupel pro Klassifikationsknoten Level 0

$$((D + f) * M + D * 3^K * K) * b$$

Ein Attribut pro Klassifikationsstufe →  $K$

## Star- vs. Snowflake-Schema /3

- Anfrage: Verkäufe der Produktgruppe „Soft-Drink“ pro Filiale und Jahr
- Snowflake-Schema:
 

```

SELECT F.Bezeichnung, J.Jahr, SUM(Verkäufe)
FROM Verkauf V, Filiale F, Artikel A,
        Produktgruppe PG, Tag T, Monat M, Jahr J
WHERE V.Artikel_ID = A.Artikel_ID AND
        A.Gruppe_ID = PG.Gruppe_ID AND
        PG.Bezeichnung = 'Soft-Drink' AND
        V.Tag_ID = T.Tag_ID AND
        T.Monat_ID = M.Monat_ID AND
        M.Jahr_ID = J.Jahr_ID AND
        V.Filial_ID = F.Filial_ID
GROUP BY F.Bezeichnung, J.Jahr
            
```
- Anzahl der Joins: 6 (steigt linear mit Anzahl der Aggregationspfade)

## Star- vs. Snowflake-Schema /4

- Anfrage für Star-Schema:
 

```

SELECT G.Filiale, Z.Jahr, SUM(Verkäufe)
FROM Verkauf V, Geographie G, Produkt P,
        Zeit Z
WHERE V.Produkt_ID = P.Produkt_ID AND
        V.Zeit_ID = Z.Zeit_ID AND
        V.Geographie_ID = G.Geographie_ID AND
        P.Produktgruppe = 'Soft-Drink'
GROUP BY G.Filiale, Z.Jahr
            
```
- Anzahl der Joins: 3 (unabhängig von der Länge der Aggregationspfade)

## Mischformen

- Abbildung einzelner Dimensionen analog Snowflake-Schema oder Star-Schema
- Entscheidungskriterien:
  - Änderungshäufigkeit der Dimensionen: Reduzierung des Pflegeaufwandes durch Normalisierung (Snowflake)
  - Anzahl der Klassifikationsstufen einer Dimension: mehr Klassifikationsstufen → größere Redundanz im Star-Schema

## Mischformen /2

- Entscheidungskriterien (fortg.):
  - *Anzahl der Dimensionselemente:*  
Einsparung durch Normalisierung bei vielen Elementen einer Dimension auf niedrigster Klassifikationsstufe
  - *Materialisierung von Aggregaten:*  
Performance-Verbesserung durch Normalisierung bei materialisierten Aggregaten für eine Klassifikationsstufe

## Galaxie

- Star-Schema
  - eine Faktentabelle
  - mehrere Kennzahlen nur möglich bei gleichen Dimensionen
- Galaxie
  - mehrere Faktentabellen
  - teilweise mit gleichen Dimensionstabellen verknüpft
  - auch: Multi-Faktentabellen-Schema, Multi-Cube, Hyper-Cube

## Fact Constellation

- Speicherung vorberechneter Aggregate in Faktentabelle
  - Beispiel: Umsatz für Region
  - Unterscheidung in Dimensionstabelle über spezielle Attribute (Bsp.: „Stufe“)
- Alternative: Auslagerung in eigene Faktentabelle  
→ Fact-Constellation-Schema  
(Spezialfall eines Galaxie-Schemas)

## Darstellung von Klassifikationshierarchien

- **Horizontal:** Modellierung der Stufen der Klassifikationshierarchie als Spalten der denormalisierten Dimensionstabelle
- Vorteil:
  - Einschränkungen auf höherer Granularität ohne Join
- Nachteile:
  - Duplikateliminierung beim Anfragen bestimmter Stufen (Bsp.: Produktgruppe innerhalb einer Kategorie)
  - Schemaänderung beim Hinzufügen neuer Stufen

Produkt_ID	Artikel	Produktgruppe	Produktkategorie
1234	Immer Ultra	Hygiene	Kosmetik
1235	Putzi	Hygiene	Kosmetik
2345	Rohrfrei	Reiniger	Haushalt

## Darstellung von Klassifikationshierarchien

- **Vertikal (rekursiv):** normalisierte Dimensionstabelle mit Attributen
  - Dimensions\_ID: Schlüssel für Faktentabelle
  - Eltern\_ID: Attributwert der Dimensions-ID der nächsthöheren Stufe
- Vorteile:
  - Einfache Änderung am Klassifikationsschema
  - Einfache Behandlung vorberechneter Aggregate
- Nachteil:
  - Self-Join für Anfragen einzelner Stufen (Bsp.: Produktgruppe innerhalb einer Kategorie)

Dimensions_ID	Eltern_ID
Immer Ultra	Hygiene
Hygiene	Kosmetik
Putzi	Hygiene

## Darstellung von Klassifikationshierarchien

- **Kombiniert:** Verbindung beider Strategien
  - Repräsentation der Klassifikationsstufen als Spalten (jedoch generische Bezeichnung)
  - Speicherung der Knoten aller höheren Stufen als Tupel
  - Zusätzliches Attribut „Stufe“ → Angabe der bezeichneten Klassifikationsstufe

Dimensions_ID	Stufe1_ID	Stufe2_ID	Stufe
Immer Ultra	Hygiene	Kosmetik	0
Putzi	Hygiene	Kosmetik	0
Hygiene	Kosmetik	NULL	1
Kosmetik	NULL	NULL	2

## Vermeidung von Semantikverlusten

- Semantikverlust bei relationaler Abbildung:
  - Unterscheidung zwischen Kennzahl und Dimension (Attribute der Faktentabelle)
  - Attribute von Dimensionstabellen (beschreibend, Aufbau der Hierarchie)
  - Aufbau der Dimensionen (Drill-Pfade)
- Ausweg:
  - Erweiterung des Systemkatalogs um Metadaten für multidimensionale Anwendungen
  - Beispiel: `CREATE DIMENSION, HIERARCHY` in Oracle

## Probleme der relationalen Umsetzung

- Transformation multidimensionaler Anfragen in relationale Repräsentation notwendig → komplexe Anfragen
- Einsatz komplexer Anfragewerkzeuge notwendig (OLAP-Werkzeuge)
- Semantikverlust
- daher: direkte multidimensionale Speicherung ?

## Multidimensionale Speicherung

- Verwendung unterschiedlicher Datenstrukturen für Datenwürfel und Dimension
- Speicherung des Würfels als Array
- Ordnung der Dimension für Adressierung der Würfelzellen notwendig
- häufig proprietäre Strukturen (und Systeme)

## Datenstrukturen

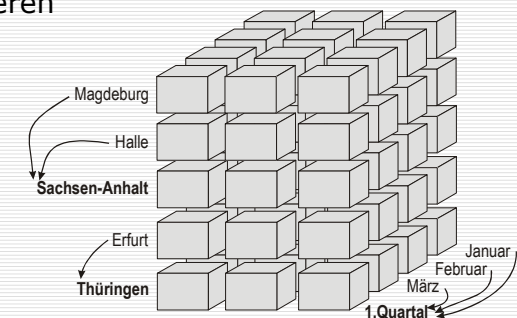
- Dimension:
  - endliche, geordnete Liste von Dimensionswerten
  - Dimensionswerte: einfache unstrukturierte Datentypen (String, Integer, Date)
  - Ordnung der Dimensionswerte (interne ganze Ordnungszahl 2 oder 4 Byte) → Endlichkeit der Werteliste

## Datenstrukturen

- Würfel:
  - Für  $n$  Dimensionen:  $n$ -dimensionaler Raum
  - $m$  Dimensionswerte einer Dimension: Aufteilung des Würfels in  $m$  parallele Ebenen
  - durch Endlichkeit der Dimensionswerteliste: endliche, gleichgroße Liste von Ebenen je Dimension
  - Zelle eines  $n$ -dimensionalen Würfels wird eindeutig über  $n$ -Tupel von Dimensionswerten identifiziert
  - Zelle kann ein oder mehrere Kennzahlen eines zuvor definierten Datentyps aufnehmen
  - Bei mehreren Kennzahlen: Alternative → mehrere Datenwürfel

## Klassifikationshierarchien

- Dimensionswerte umfassen alle Ausprägungen der Dimension: Elemente (Blätter) und Knoten der höheren Klassifikationsstufen
- Knoten der höheren Stufen bilden weitere Ebenen



## Berechnung von Aggregationen

- Echtzeit:
  - bei Anfrage von Zellen, die Werte einer höheren, aggregierten Klassifikationsstufe repräsentieren → Berechnung aus Detaildaten
  - hohe Aktualität, jedoch hoher Aufwand
  - eventuell Caching
- Vorberechnung:
  - nach Übernahme der Detaildaten → Berechnung und Eintragen der Aggregationswerte in entsprechende Zellen
  - Neuberechnung nach jeder Datenübernahme notwendig
  - hohe Anfragegeschwindigkeit, jedoch Zunahme der Würfelgröße und Laufzeitaufwand
- Ausweg: *inkrementelle Vorberechnung*

## Weitere Datenstrukturen

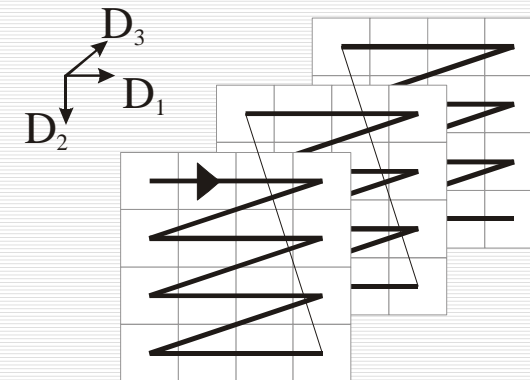
- Attribute
  - klassifizierende Merkmale einer Dimension
  - Identifizierung von Untermengen von Dimensionswerten (z.B. „Produktfarbe“)
  - nicht zur Vorberechnung vorgesehen
- Virtueller Würfel
  - umfasst abgeleitete Daten („Gewinn“, „prozentualer Umsatz“)
  - Ableitung aus anderen Würfeln durch Anwendung von Berechnungsfunktionen ≅ Sichten im relationalen Modell
- Teilwürfel
  - Kombination mehrerer Ebenen eines Würfels → virtuell

## Array-Speicherung

- Speicherung des Würfels:  $n$ -dimensionales Array → Linearisierung in eine eindimensionale Liste
- Indizes des Arrays ≅ Koordinaten der Würfelzellen (Dimensionen  $D_i$ )
- Indexberechnung für Zelle mit Koordinaten

$$\begin{aligned} x_1 \dots x_n \quad \text{Index}(z) &= x_1 + (x_2 - 1) \cdot |D_1| \\ &+ (x_3 - 1) \cdot |D_1| \cdot |D_2| + \dots \\ &+ (x_n - 1) \cdot |D_1| \cdot \dots \cdot |D_{n-1}| \end{aligned}$$

## Linearisierungsreihenfolge



## Array-Speicherung: Probleme

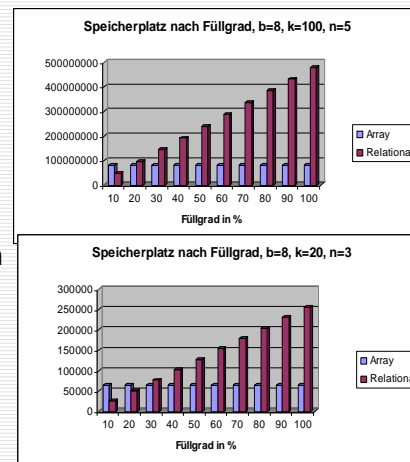
- Zahl der Plattenzugriffe bei ungünstigen Linearisierungsreihenfolgen
  - Reihenfolge der Dimensionen ist bei Definition des Würfels zu beachten
- Caching zur Reduzierung notwendig
- Speicherung dünn besetzter Würfel

## Speicherverbrauch

	Array	Relational (Star-Schema)
Speicherung Koordinaten	Implizit (Linearisierung)	Explizit (redundant)
Leere Zellen	Belegen Platz	Belegen keinen Platz
Neue Klas. Knoten	Komplette Reorganisation Starkes Wachstum im Speicherverbrauch	Neue Zeile in Dimensionstabelle Kaum Wachstum im Speicherverbrauch
Speicherverbrauch	$b * \prod_{i=1}^n d_i$	$b * M * n$ (plus Fakten)

## Vergleich Speicherplatz

- Faktoren
  - Füllgrad
  - k: Anz. Knoten
  - n: Anz. Dim.
- Schon bei geringen Füllgraden ist Array SP-effizienter
- Performance hängt von vielen Faktoren ab
  - Indexierung
  - Sequential reads
  - ...



## Grenzen der multidim. Speicherung

- Skalierbarkeitsprobleme aufgrund dünn besetzter Datenräume
- teilweise einseitige Optimierung bezüglich Leseoperationen
- Ordnung der Dimensionen notwendig (durch Array-Speicherung) → erschwert Änderungen an den Dimensionen
- keine Standard für multidimensionale DBMS
- Spezialwissen notwendig

## Hybride Speicherung

---

- HOLAP: Verbindung der Vorteile beider Welten
  - Relational (Skalierbarkeit, Standard)
  - Multidimensional (analytische Mächtigkeit, direkte OLAP-Unterstützung)
- Speicherung:
  - Relationale Datenbank: Detaildaten
  - Multidimensionale Datenbank: aggregierte Daten
  - Multidim. Speicherstrukturen als „intelligenter“ Cache für häufig angeforderte Datenwürfel
- transparenter Zugriff über multidimensionales Anfragesystem

## Literatur

---

- Vassiliadis: „Modeling Multidimensional Databases, Cubes, and Cube Operations“, SSDBM, 1998
- Sapia et al. „Extending the E/R Model for the Multidimensional Paradigm“, Workshop DWDM, 1998
- Lenz, Shoshani: „Summarizability in OLAP and Statistical Databases“, SSDBM, 1997