

Data Warehousing und Data Mining

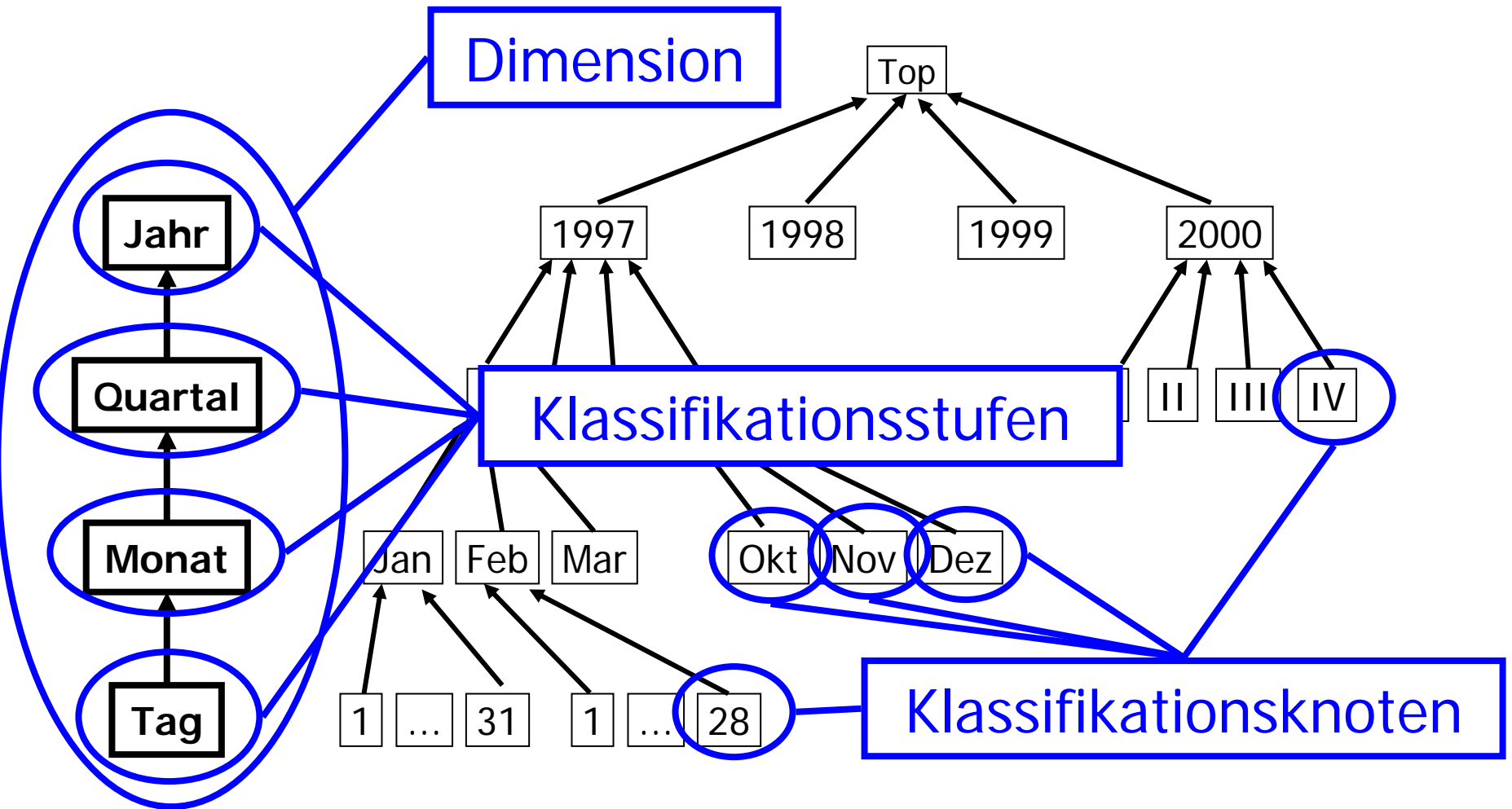
Speicherung multidimensionaler Daten



Ulf Leser
Wissensmanagement in der
Bioinformatik



Dimension



OLAP Operationen

- MDDM entspricht der Sprache des Betriebswirts
 - Umgang mit Objekten seiner täglichen Welt (Kunden, Waren, etc.)
 - Keine technischen Details
- **Operationen auf dem MDDM** müssen Analysevorgänge abbilden bzw. unterstützen
 - Schnelle und intuitive Manipulation der Daten
 - Abbildung der **Aufgaben von Betriebswirten**
- Die wichtigsten Operationen
 - **Aggregation** (Roll-Up): Granularität wird erniedrigt
 - **Verfeinerung** (Drill-Down): Granularität wird erhöht
 - Aggregation bzgl. einer Funktion f wie SUM, AVG, MEDIAN, ...
 - Und viele mehr, die später kommen ...

Datenpunkte und Würfelkoordinaten

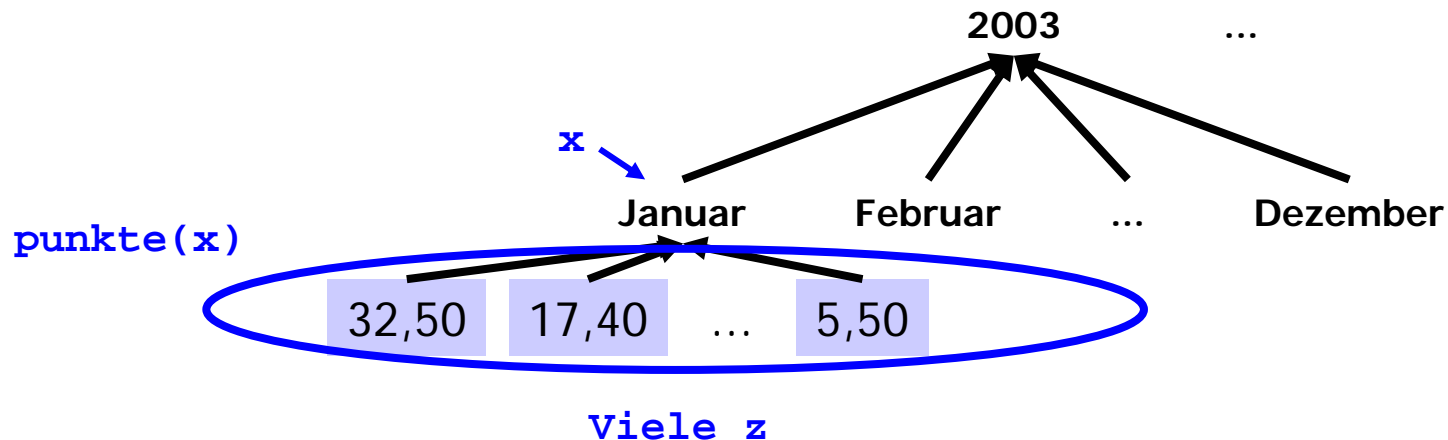
- *Definition*

Gegeben Würfel W , Dimension D und ein Klassifikationsknoten x aus D . Sei z ein Datenpunkt (Fakt) aus W und $D(z)$ seine Koordinate bzgl. D . Dann gilt

- z liegt in x gdw. $D(z)=x$ oder $D(z) \in \text{nachfahren}(x)$
- $\text{punkte}(x)$ ist die Menge aller Datenpunkte z mit $D(z)$ in x

- *Bemerkung*

- Erweiterung auf mehrere Koordinaten durch Schnittmengenbildung
 - $\text{punkte}(x,y,z) = \text{punkte}(x) \cap \text{punkte}(y) \cap \text{punkte}(z)$
mit x in D_1 , y in D_2 , z in D_3



Aggregation in Hierarchien

- *Definition*

- Gegeben

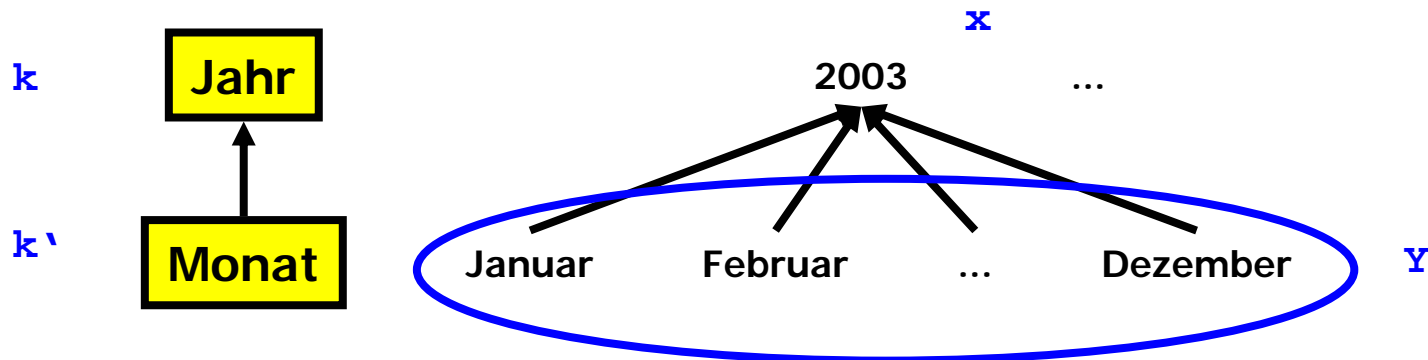
- Dimension D , Pfad $P = \{k_0 \rightarrow \dots \rightarrow k' \rightarrow k \rightarrow \dots \rightarrow \text{TOP}\}$ in D
 - x ein Klassifikationsknoten der Klassifikationsstufe k aus P
 - Aggregatfunktion f , Measure F

- Sei $Y = \text{kinder}(x)$ die Menge $\{y_1, \dots, y_n\} \subseteq \text{knoten}(k')$ von Klassifikationsknoten von k' , von denen x funktional abhängt

- Die *Aggregation von Y nach x* bzgl. f und F bezeichnet die Berechnung des *aggregierten Faktes* $F(x) = f(F(y_1), \dots, F(y_n))$

- Die *Aggregation von k' nach k* bzgl. f und F bezeichnet die Berechnung von $F(x)$ für alle $x \in \text{knoten}(k)$

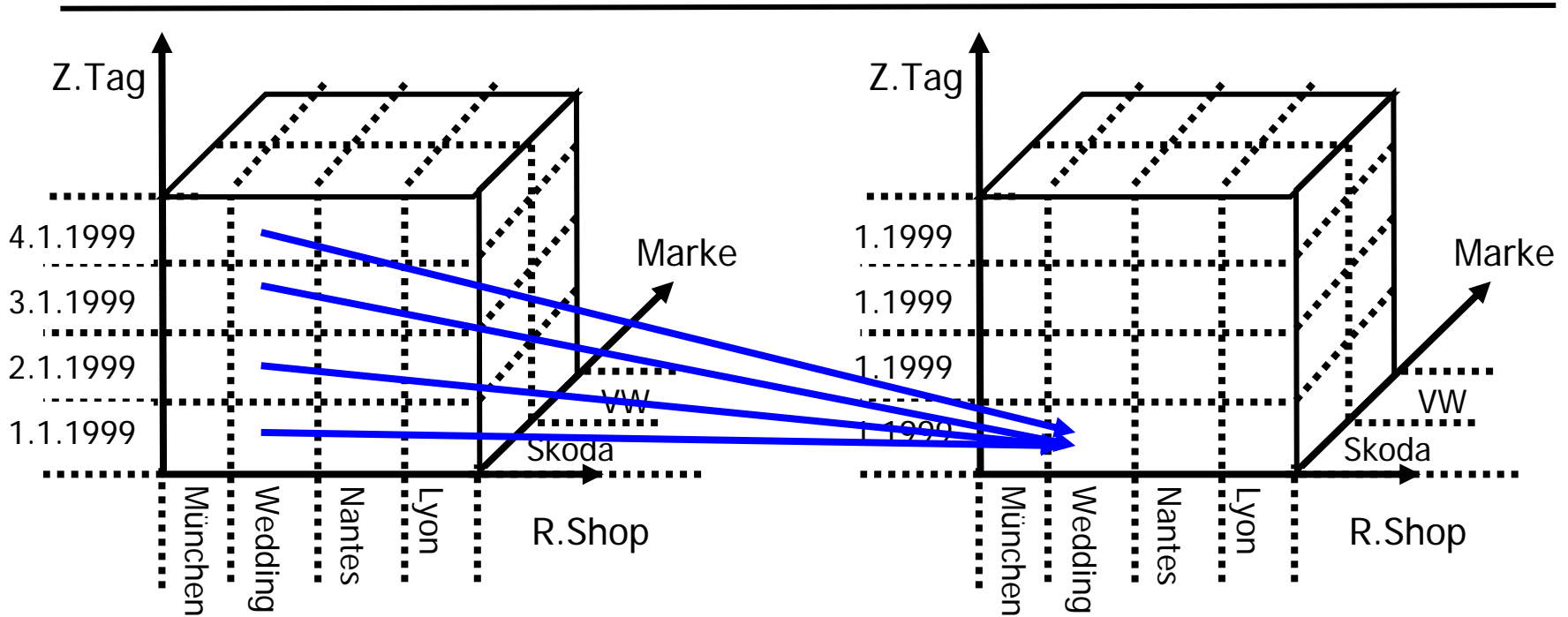
- Das schreiben wir kurz als $F(k)$



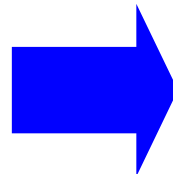
Wo sind wir?

- Wir können entlang eines Pfades in einer Dimension für ein Measure mit einer Aggregatfunktion die aggregierten Fakten für höherstufige Knoten berechnen
- Zwei Möglichkeiten
 - Aggregierte Measures von Knoten einer Stufe aus aggregierten Measures der nächstkleineren Stufe berechnen
 - Verlangt Präaggregation
 - Aggregierte Measures immer direkt aus den Werten aller Datenpunkte mit den entsprechenden Koordinaten berechnen
 - Ist eine Anfrage über dem Cube höchster Granularität

Übersicht



(1.101, 1.1.1999, Wedding, Skoda)
 (129, 2.1.1999, Wedding, Skoda)
 (225, 3.1.1999, Wedding, Skoda)
 (1.540, 4.1.1999, Wedding, Skoda)
 (2.500, 5.1.1999, Nantes, Skoda)
 ...



(3.000, 1.1999, Wedding, Skoda)
 (2.500, 2.1999, Nantes, Skoda)
 ...

Rollup Tag→Monat

Würfelinhalt

- Ein Würfel $W = (G, F)$
 - Granularität $G = (D_1.k_1, \dots, D_n.k_n)$
 - Menge Measures $F = \{F_1, \dots, F_m\}$ mit Aggregatfunktionen f_1, \dots, f_m
- Schreibweise für **Zellen eines Würfels**
 - $W(x_1, x_2, \dots, x_n) = (F_1, \dots, F_m)$
 - x_i sind die Koordinaten im Würfel bzgl. G : $x_i \in \text{knoten}(k_i)$
 - Pro Punkt in W gibt es m (evt. aggregierte) Measures F_1, \dots, F_m
- Betrachten wir den **einfachen Fall**: $n = m = 1$
 - Dimension D mit Knoten x , ein Measure F mit Aggregatfunktion f
 - Dann: $W(x) = F(x) = f(\{F(y) \mid y \in \text{kinder}(x)\})$
- Bemerkung
 - Auf unterster Ebene ist $\text{kinder}(x) = \text{punkte}(x)$

Operationen auf Würfeln

- **OLAP Operationen** überführen einen Würfel $W=(G,F)$ in einen Würfel $W'=(G',F')$
- Dabei gilt
 - Aggregation: $G < G'$
 - Verfeinerung $G > G'$
- Eine **einfache Operation** verändert nur die Klassifikationsstufe einer Dimension in G
 - Komplexe Operationen können auf natürliche Weise aus einfachen Operationen durch Verkettung zusammengesetzt werden
 - Wir betrachten im folgenden **nur einfache Operationen**

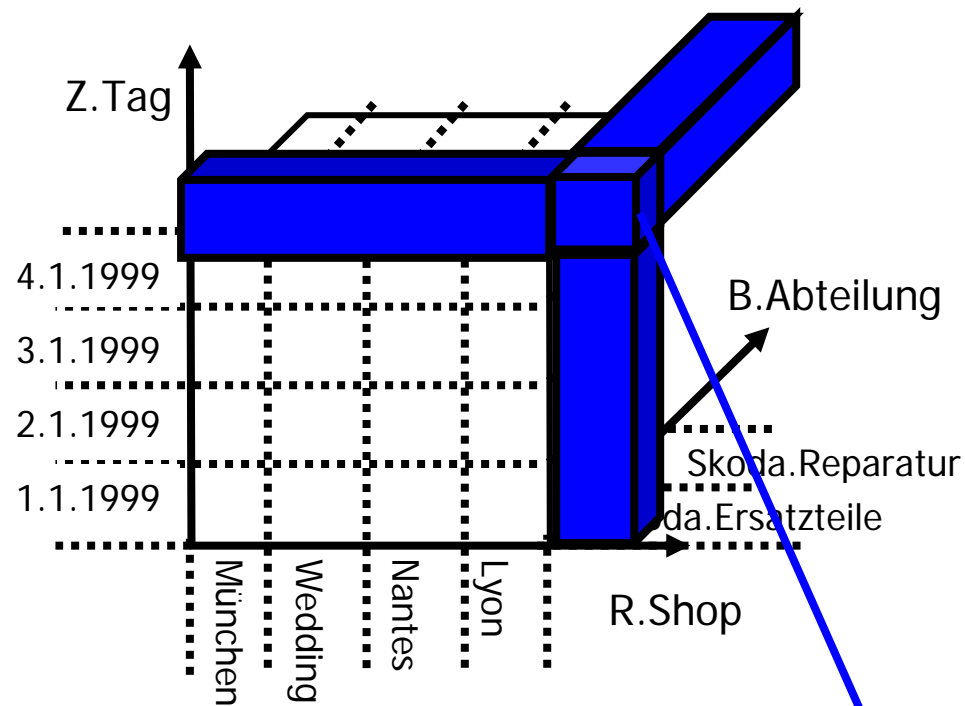
OLAP Operation: Aggregation (Roll-Up)

- Definition

- Gegeben ein Würfel $W=(G,F)$ mit $G=(D_1.k_1, \dots, D_i.k_i, \dots, D_n.k_n)$ und $F=(F_1, \dots, F_m)$
- Sei $P=\{k_0 \rightarrow \dots \rightarrow k_{i''} \rightarrow k_i \rightarrow k_{i'} \rightarrow \dots \rightarrow TOP\}$ ein Pfad in D_i
- Die *einfache Aggregation in W entlang P mit Aggregatfunktion f überführt W in*

 - $W' = ((D_1.k_1, \dots, D_i.k_{i''}, \dots, D_n.k_n), (F_{1'}, \dots, F_{m'}))$
 - $(F_{1'}, \dots, F_{m'}) = (F_1(\text{punkte}(k_1) \cap \dots \text{punkte}(k_{i'}) \cap \dots \text{punkte}(k_n)), \dots, F_m(\text{punkte}(k_1) \cap \dots \text{punkte}(k_{i'}) \cap \dots \text{punkte}(k_n)))$

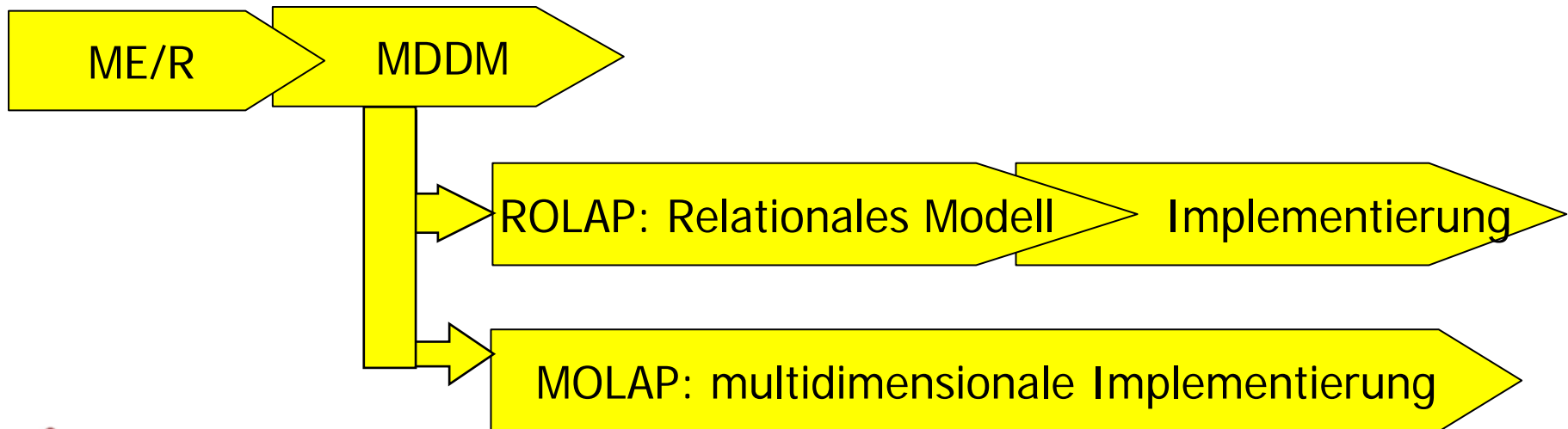
... in mehreren Dimensionen



$$G = (Z.TOP, R.TOP, B.TOP)$$

ME/R [SPHD98]

- Logisches MDDM
 - Fakten, Klassifikationsstufen, Dimensionen, ...
- Warum ein graphisches Modell für MDDM?
- E/R nicht ausreichend
 - Keine Repräsentation von MDDM Konzepten
- **ME/R: Erweitertes E/R Modell**



Klassen von Aggregatfunktionen [LS97]

- Definition

Gegeben eine Menge X und eine Partitionierung (X_1, X_2, \dots, X_n) von X . Eine Aggregatfunktion f heißt:

- *distributiv* gdw $\exists g: f(X) = f(g(X_1), g(X_2), \dots, g(X_n))$
- *algebraisch* gdw $f(X)$ berechenbar aus fester Menge von g 's
 - *Deren Zahl und Art unabhängig von X ist*
- *holistisch* gdw $f(X)$ kann nur aus den Grundelementen von X berechnet werden
 - Die Menge von g 's ist dann nur durch die Größe von X begrenzt
 - Keine Präaggregation möglich

- Bemerkungen

- X entspricht einem Klassifikationsknoten, (X_1, X_2, \dots, X_n) seinen Kindern
- Die Definition verallgemeinert offensichtlich für Hierarchien

Beispiele

Distributiv	Summe, Count, Max, Min, ...
Algebraisch	AVG (mit G_1 =SUM und G_2 =CNT) STDDEV, MaxN, ...
Holistisch	MEDIAN, RANK, PERCENTILE Highest Frequency, ...

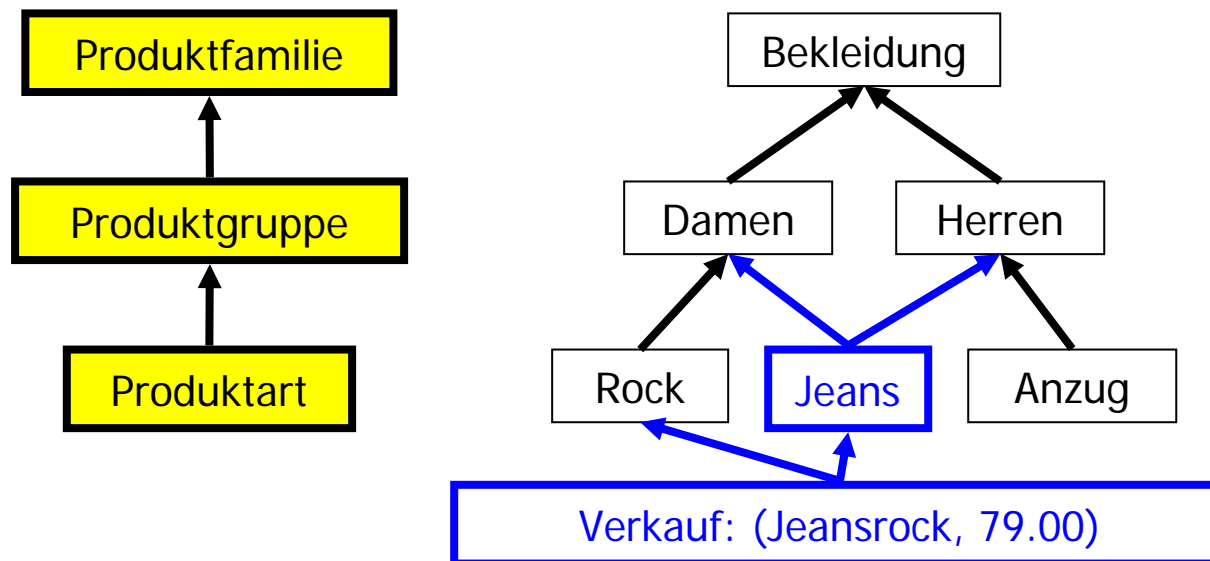
- COUNT: Man wendet SUM (als g) auf den Counts an
- Highest Frequency
 - Merke Werte und jeweilige Frequenz: $((v_1, f_1), (v_2, f_2), \dots)$
 - Merge zweier Sets möglich
 - Aber: Keine feste Grenze für Platzbedarf, da Anzahl unterschiedlicher Werte nicht fest

Überlappungsfreiheit

- Definition

*Eine Klassifikationshierarchie ist **überlappungsfrei** gdw*

- *Jeder Klassifikationsknoten mit Level i ist höchstens einem Klassifikationsknoten in Level $i+1$ zugeordnet*
- *Jeder Datenpunkt ist höchstens einem Klassifikationsknoten mit Level 0 zugeordnet*

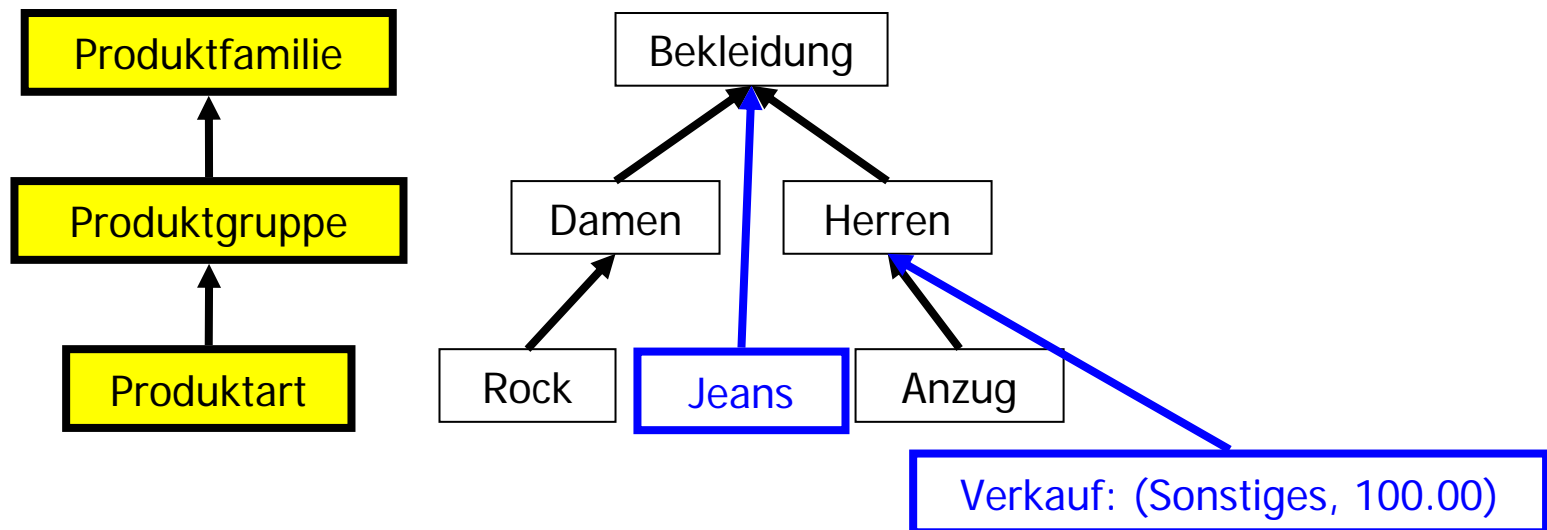


Vollständigkeit

- Definition

*Eine Klassifikationshierarchie ist **vollständig** gdw*

- *Jeder Klassifikationsknoten mit Level i ist mindestens einem Klassifikationsknoten in Level $i+1$ zugeordnet*
- *Jeder Datenpunkt ist mindestens einem Klassifikationsknoten mit Level 0 zugeordnet*



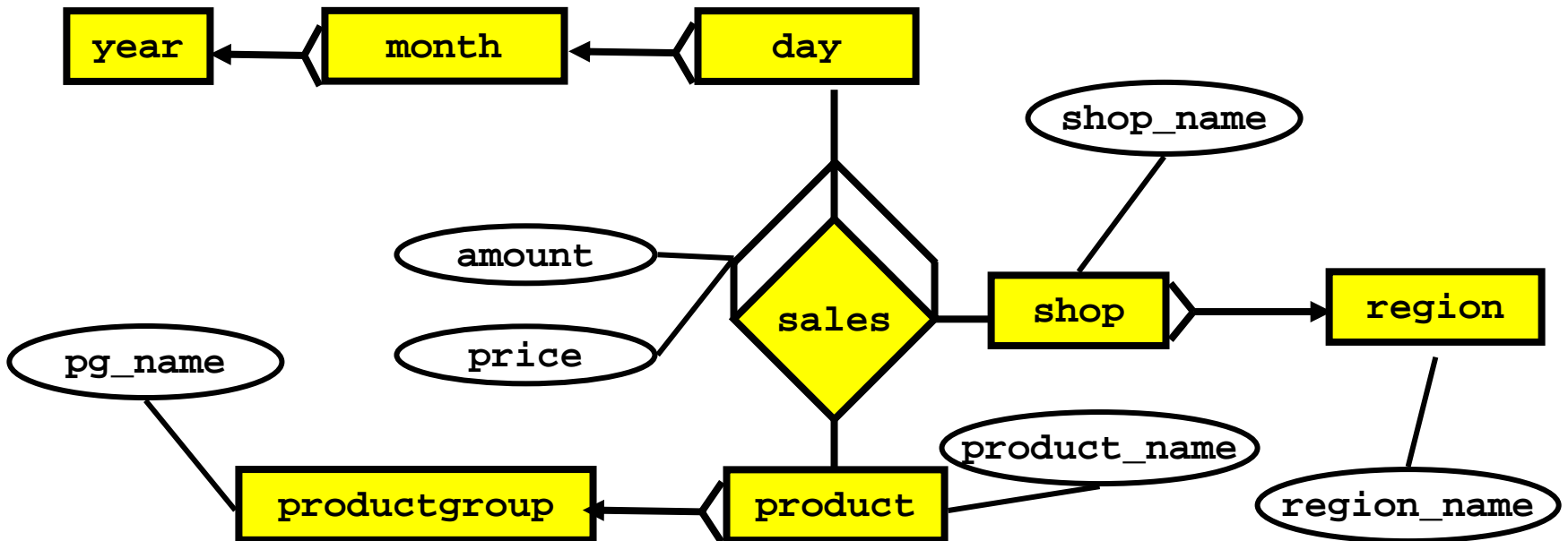
Inhalt dieser Vorlesung

- Relationales OLAP (ROLAP)
 - Snowflake-, Star- und Fullfactschema
 - Speicherplatz und Queries
 - Änderungen in den Dimensionen
 - Schemavarianten
 - Beispiel: ROLAP in Oracle
- Multidimensionales OLAP (MOLAP)

Relationales OLAP

- MDDM und relationales Modell nicht vergleichbar
 - Konzeptionelle versus logische Ebene
- **Vollkommen falsch**
 - „Relationales Modell ist zweidimensional, MDDM ist beliebig-dimensional“
- **Richtig**
 - „Relationales Modell kann beliebig-dimensionale Daten repräsentieren (in Tabellenform), benötigt dazu aber einen **Transformationsschritt**“
- **Verschiedene Abbildungen existieren**
 - Unterschiede in Speicherverbrauch, Redundanz, Performance von INSERT/SELECT/UPDATE, ...

Beispielmodell



Annahmen für Kostenbetrachtungen

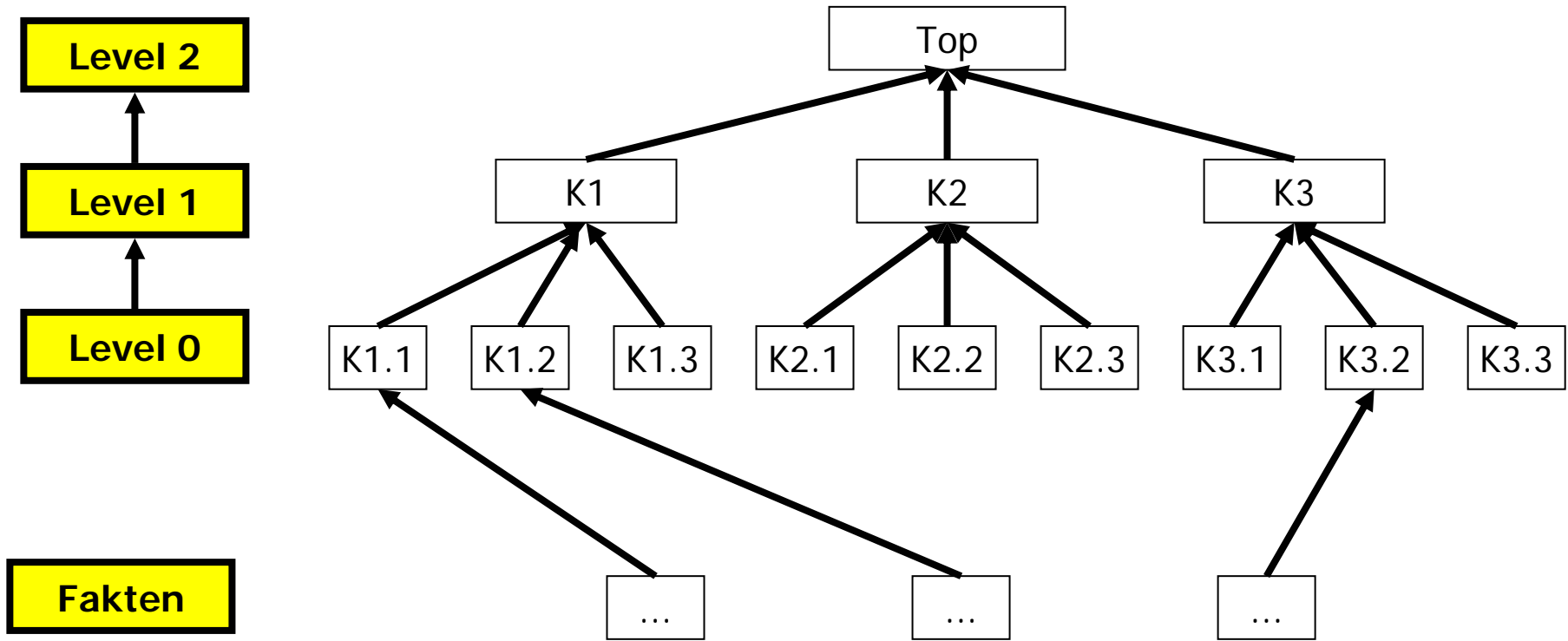
- d Dimensionen, je k Klassifikationsstufen plus Top
- Jeder **Klassifikationsknoten hat 3 Kinder**
 - Level k: $1=3^0$ Knoten (Top)
 - Level k-1: $3=3^1$ Knoten
 - Level k-2: $9=3^2$ Knoten
 - ...
 - Level 0: Höchste Granularität, 3^k Knoten

➤ $n = \sum_{i=0 \dots k} 3^i$ Knoten pro Dimension

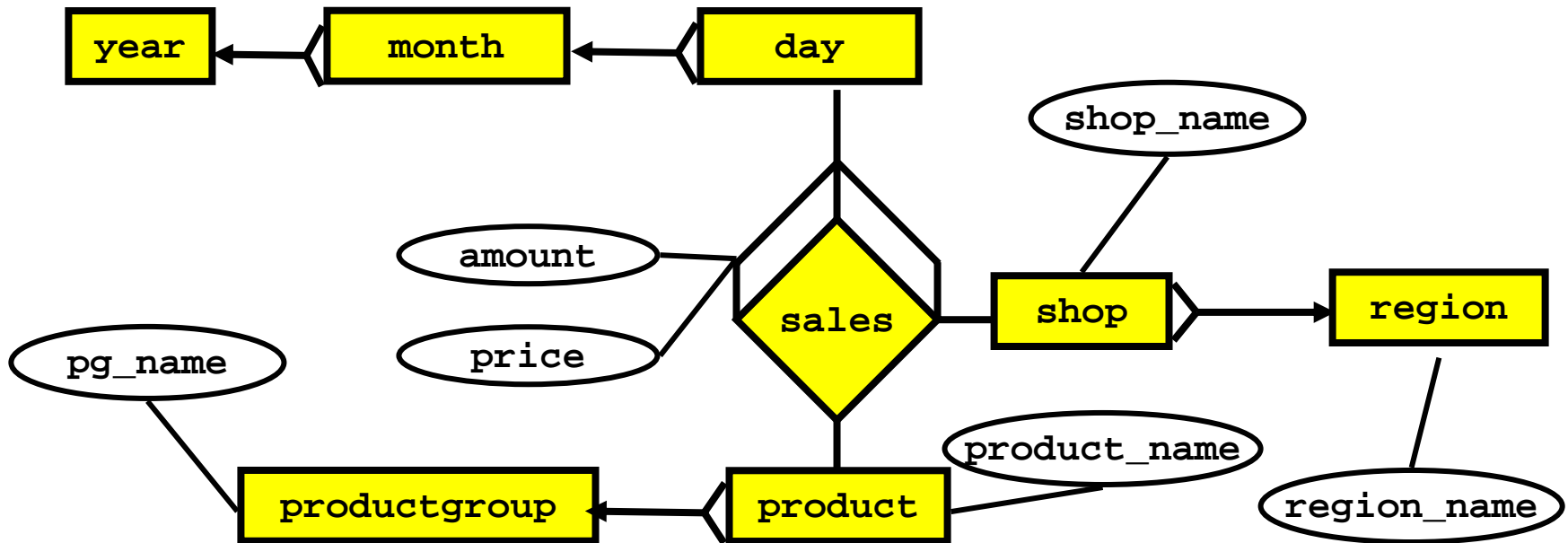
- m Fakten, **gleichverteilt** in allen Dimensionen
- Attribut: b Bytes
- Knoten haben nur ID
- Es gibt f Measures

Volle Klassifikationshierarchie

Zu jedem Knoten gibt es **gleich viele Kinder/Fakten**

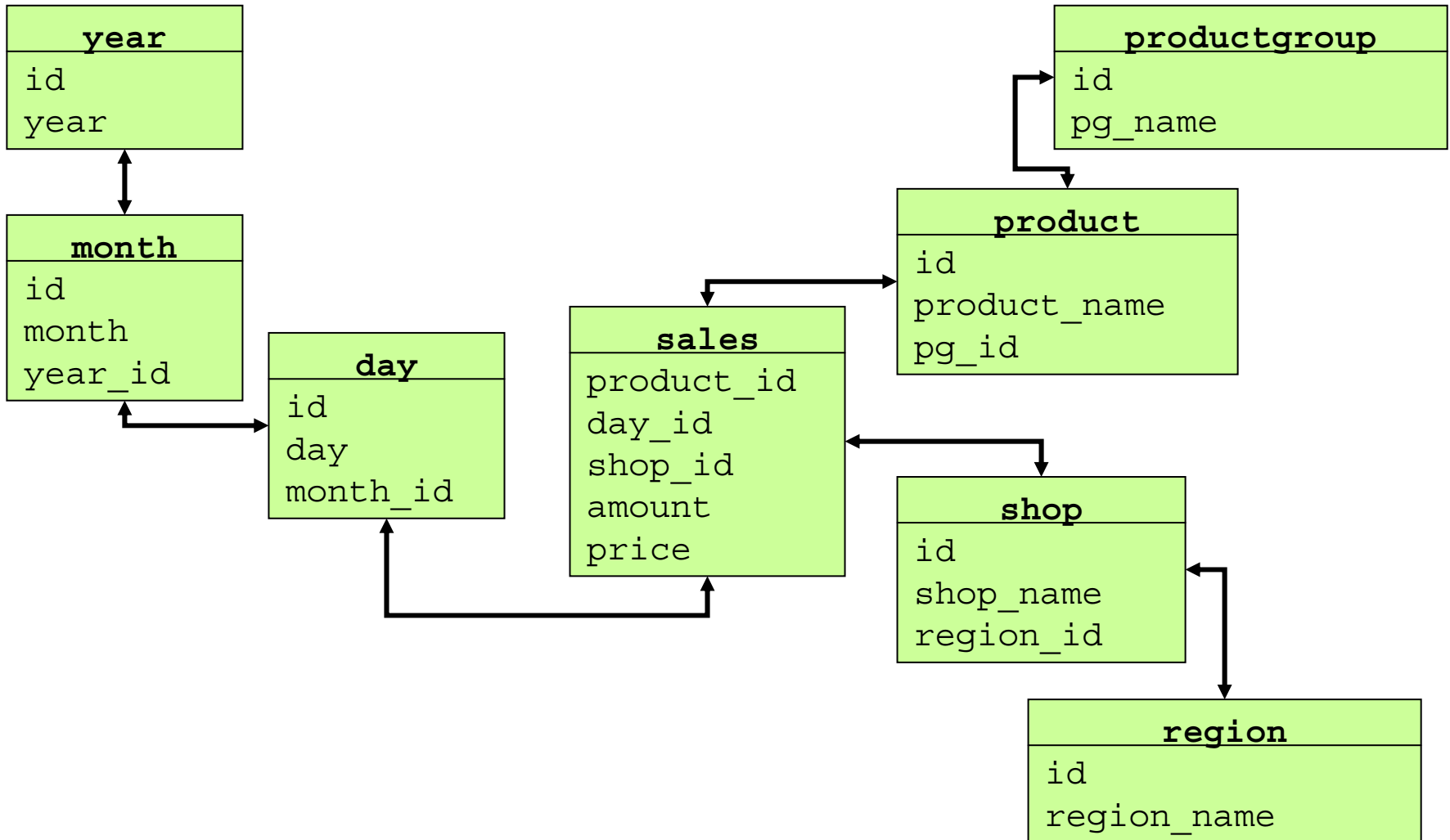


Beispielquery



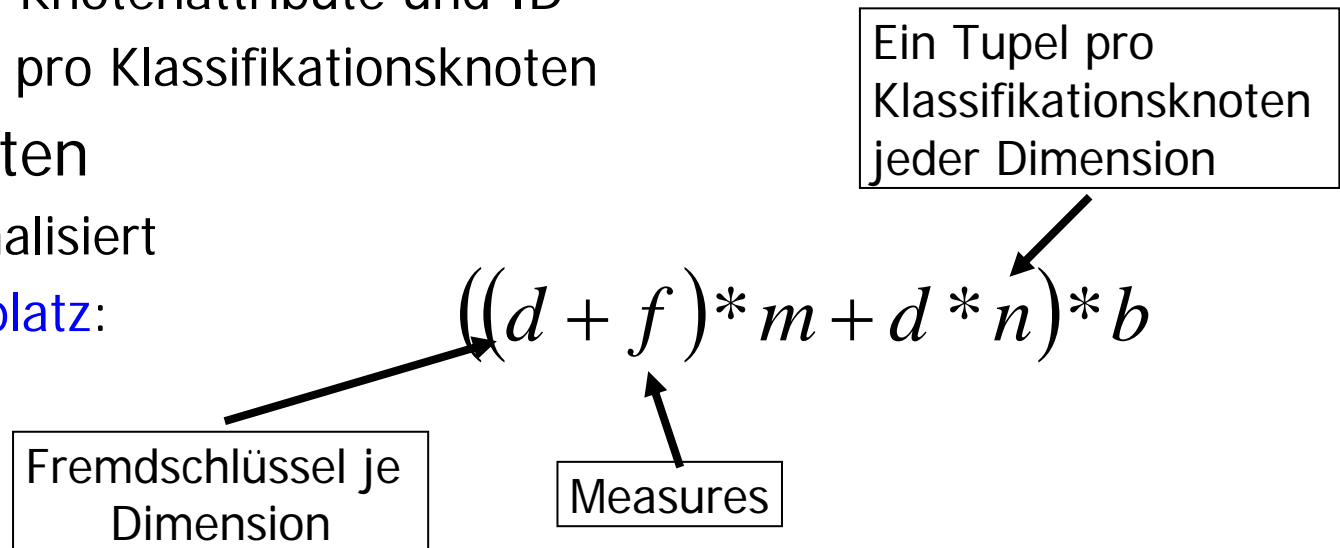
Alle Verkäufe der Produktgruppe „Wasser“ nach Shop und Jahr

Variante 1 - Snowflake



Snowflake Schema

- Faktentabelle
 - Measures plus Foreign Key der kleinsten Klassifikationsstufe jeder Dimension
- Dimensionstabellen
 - Eine Tabelle pro **Klassifikationsstufe**
 - Attribute: Knotenattribute und ID
 - Ein Tupel pro Klassifikationsknoten
- Eigenschaften
 - Voll normalisiert
 - **Speicherplatz:**

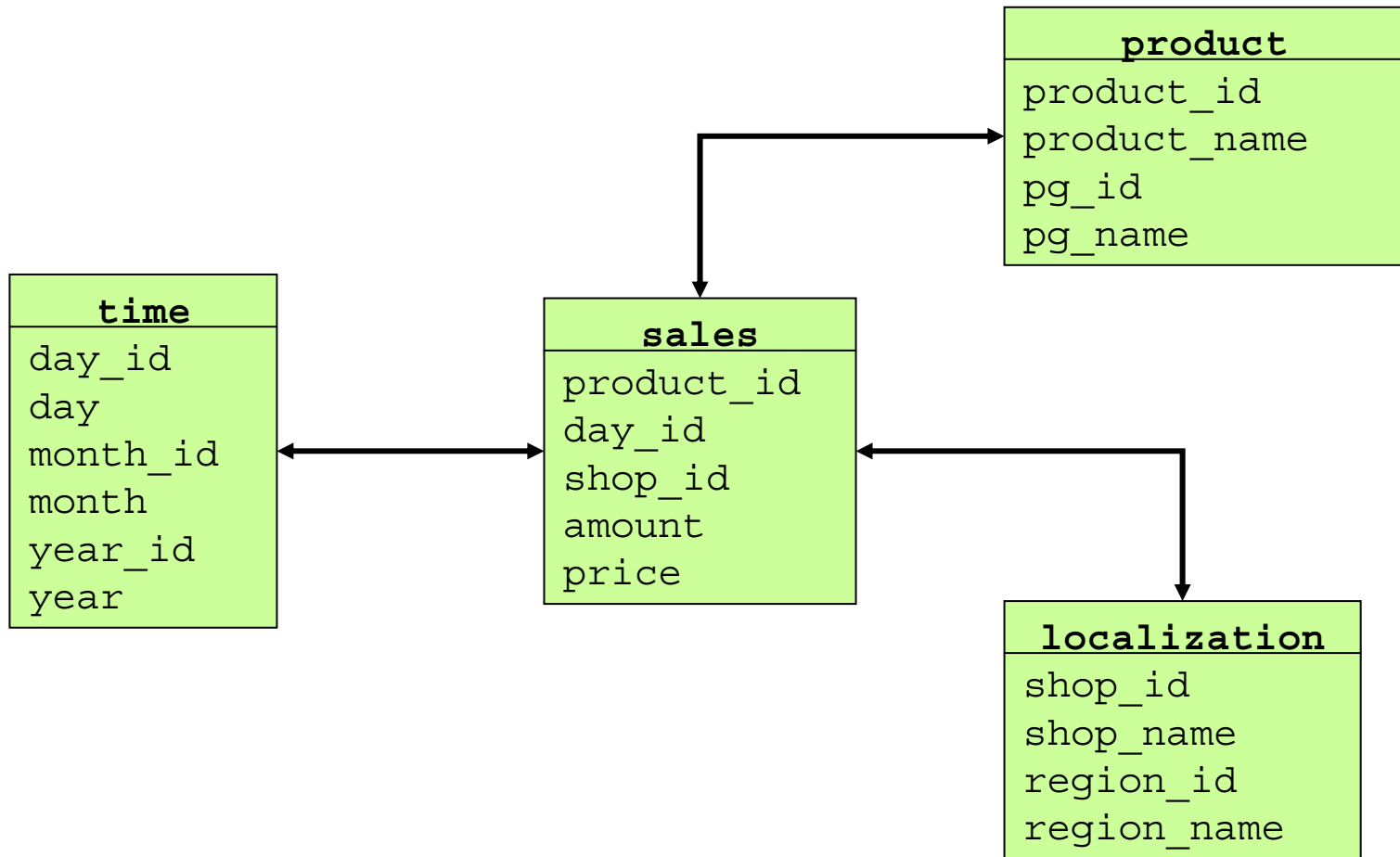


Snowflake - Query

```
SELECT    X.shop_name, y.year, sum(amount*price)
FROM      sales S, ... PG, P, D, M, Y, shop X
WHERE     PG.name=„Wasser“ AND
          PG.id = P.pg_id AND
          P.id = S.product_id AND
          D.id = S.day_id AND
          M.id = D.month_id AND
          Y.id = M.year_id AND
          X.id = S.shop_id
group by  X.shop_name, Y.year
```

- 6 Joins
- 4 Joins, wenn shop_name / year nicht notwendig
- Anzahl Joins **steigt in jeder Dimension linear** mit Länge der Aggregationspfade in der Anfrage

Variante 2: Star Schema



Star Schema

- Faktentabelle
 - Measures plus Foreign Key der kleinsten Klassifikationsstufe jeder Dimension
- Dimensionstabellen
 - Eine Tabelle pro **Dimension**
 - Attribute: ID und Knotenattribute aller Klassifikationsstufen
 - Ein Tupel pro Klassifikationsknoten mit Level 0
- Eigenschaften
 - Denormalisierte Dimensionen
 - **Speicherplatz**

Ein Tupel pro
Klassifikationsknoten Level 0

$$\left((d + f) * m + d * 3^k * k \right) * b$$

Ein Attribut pro Klassifikationsstufe

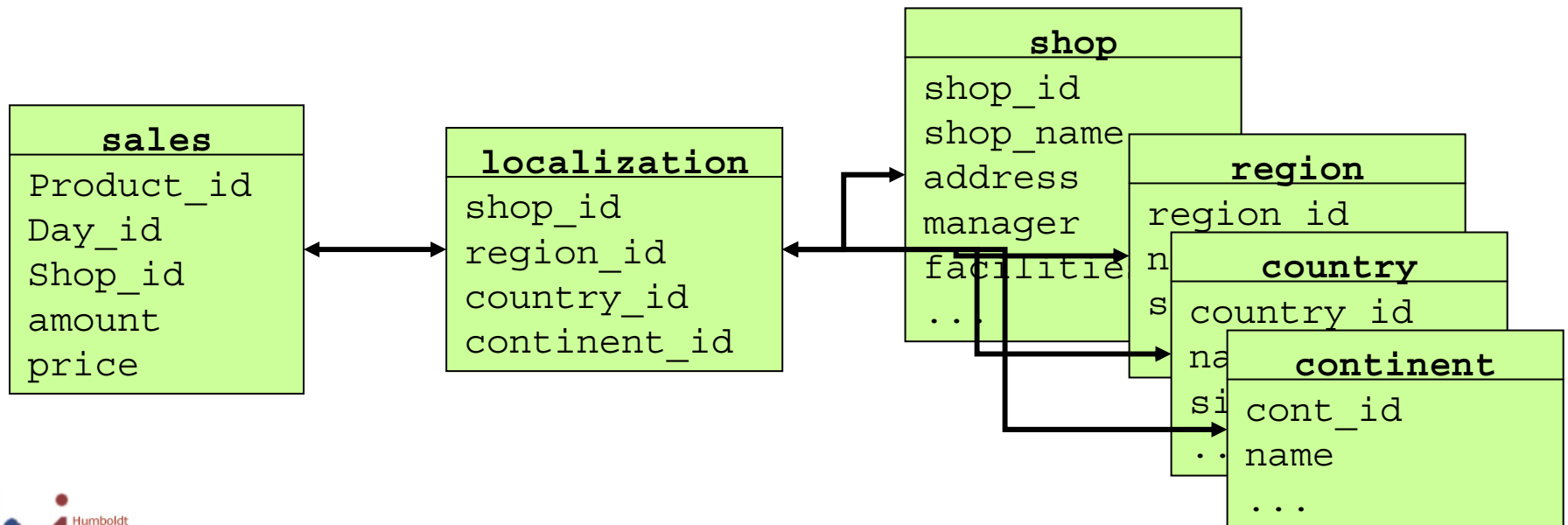
Star - Query

```
SELECT      L.shop_name, T.year, sum(amount*price)
FROM        sales S, product P, time T, localization L
WHERE       P.pg_name=„Wasser“ AND
            P.product_id = S.product_id AND
            T.day_id = S.day_id AND
            L.shop_id = S.shop_id
group by   L.shop_name, T.year
```

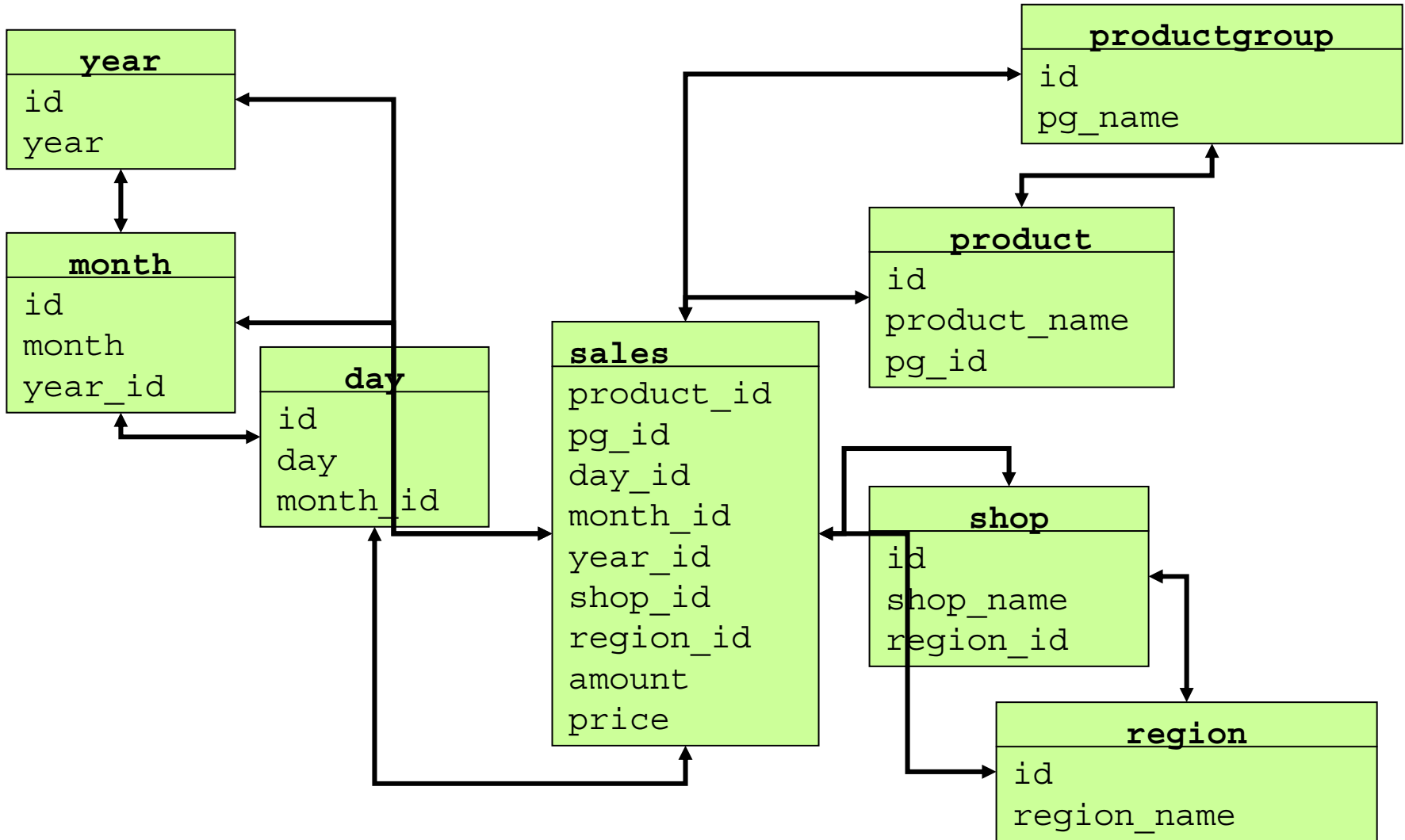
- 3 Joins
- Keine Reduktion wenn Jahrname nicht notwendig
- 2 Joins, wenn Shopname nicht notwendig
- Anzahl Joins
 - **Unabhängig von Länge** der Aggregationspfade in der Anfrage
 - Steigt **linear mit der Anzahl Dimensionen** in der Anfrage

Variante: Knotenattribute im Star Schema

- Eigenen Tabellen pro Klassifikationsstufe
 - Dimensionstabelle repräsentiert Hierarchie und enthält nur Keys
 - Normalisierter Entwurf, speicherplatzeffizient, zusätzliche Joins notwendig
- Dimensionstabelle
 - Teilweise redundante Daten, zusätzlichen Joins für „sprechende“ Attribute notwendig



Variante 3: Fullfact



Fullfact

- Faktentabelle
 - Measures plus Foreign Key jeder Klassifikationsstufe
- Dimensionstabellen
 - Eine Tabelle pro Klassifikationsstufe
 - Attribute: Knotenattribute und ID
 - Ein Tuple pro Klassifikationsknoten
- Eigenschaften
 - Normalisierte Dimensionen, denormalisierte Fakten
 - Speicherverbrauch

$$((d * k + f) * m + d * n) * b$$

Ein FK pro Klassifikationsstufe

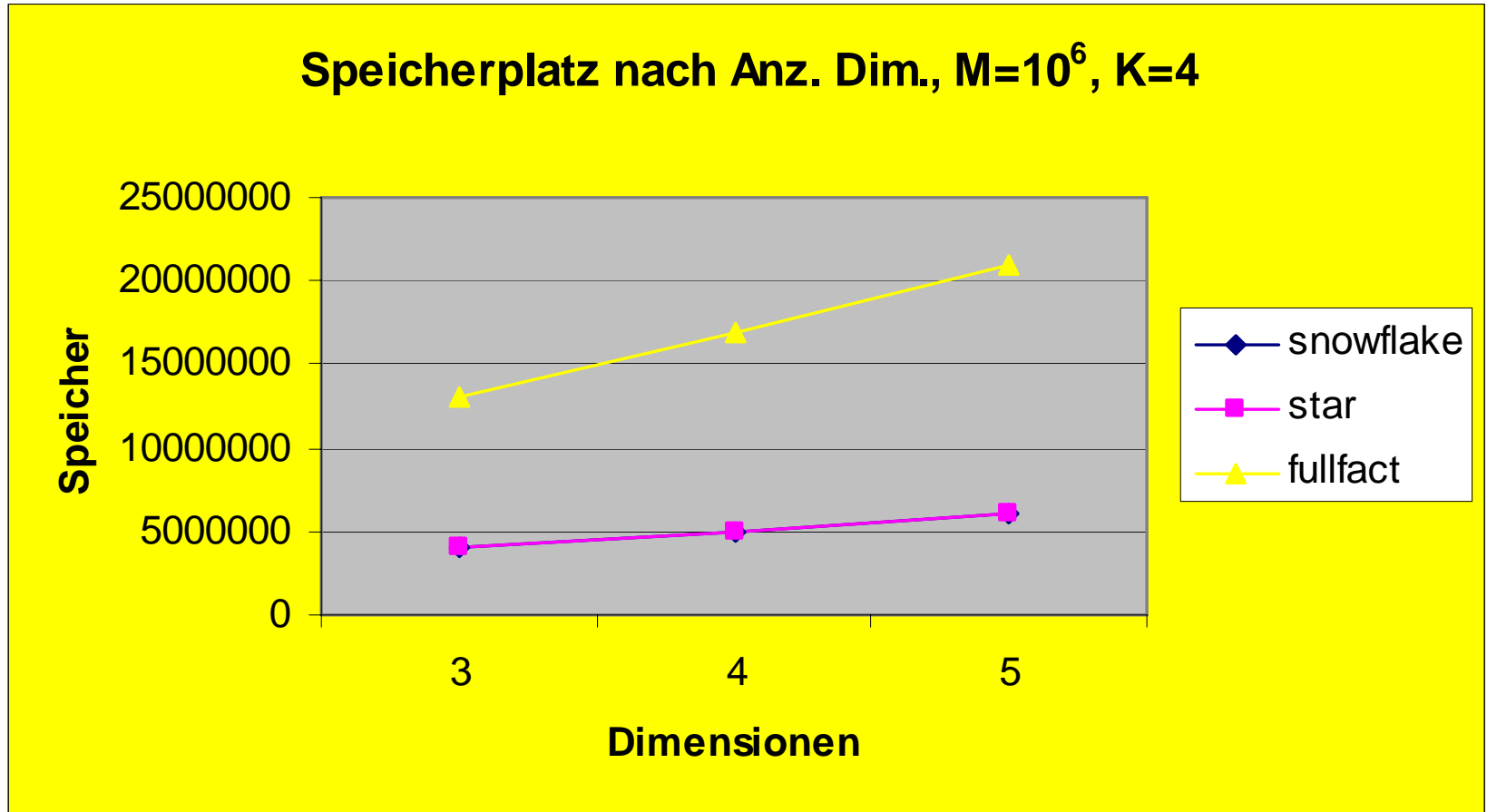


Fullfact Query

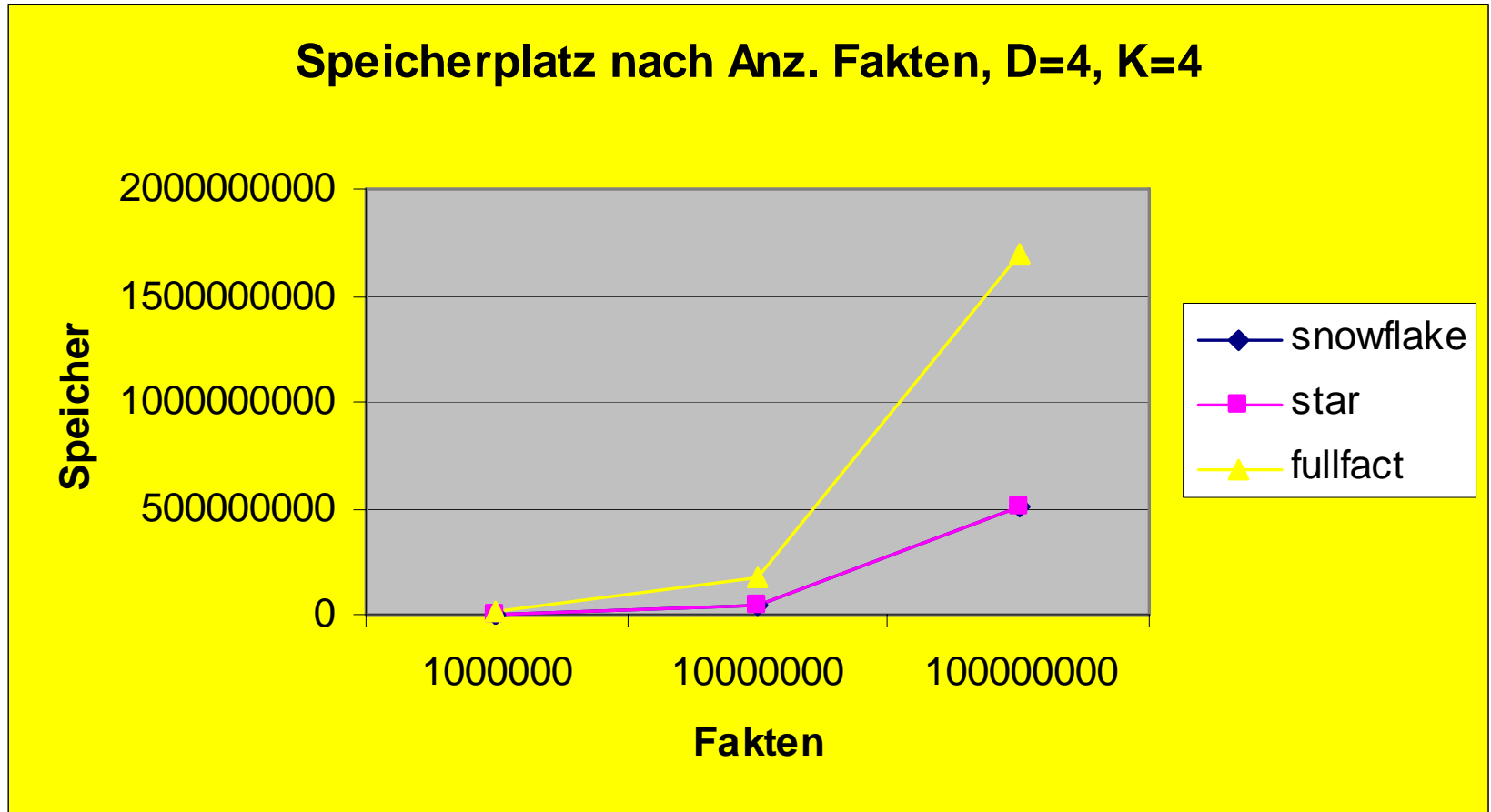
```
SELECT    X.shop_name, T.year, sum(amount*price)
FROM      sales S, productgroup PG, year Y, shop X
WHERE     PG.pg_name=„Wasser“ AND
          PG.id = S.pg_id AND
          Y.id = S.year_id AND
          X.id = S.shop_id
group by  X.shop_name, Y.year
```

- 3 Joins
- 1 Join, wenn Shopname / Jahrname nicht notwendig
- Wichtiger Effekt (Vergleich zu Star Schema)
 - Dimensionstabellen sind kleiner in allen Klassifikationsstufen mit Level != 0
- Anzahl Joins
 - **Unabhängig** von Länge der Aggregationspfade in der Anfrage
 - Steigt linear mit der Anzahl Klassifikationsstufen in der Anfrage

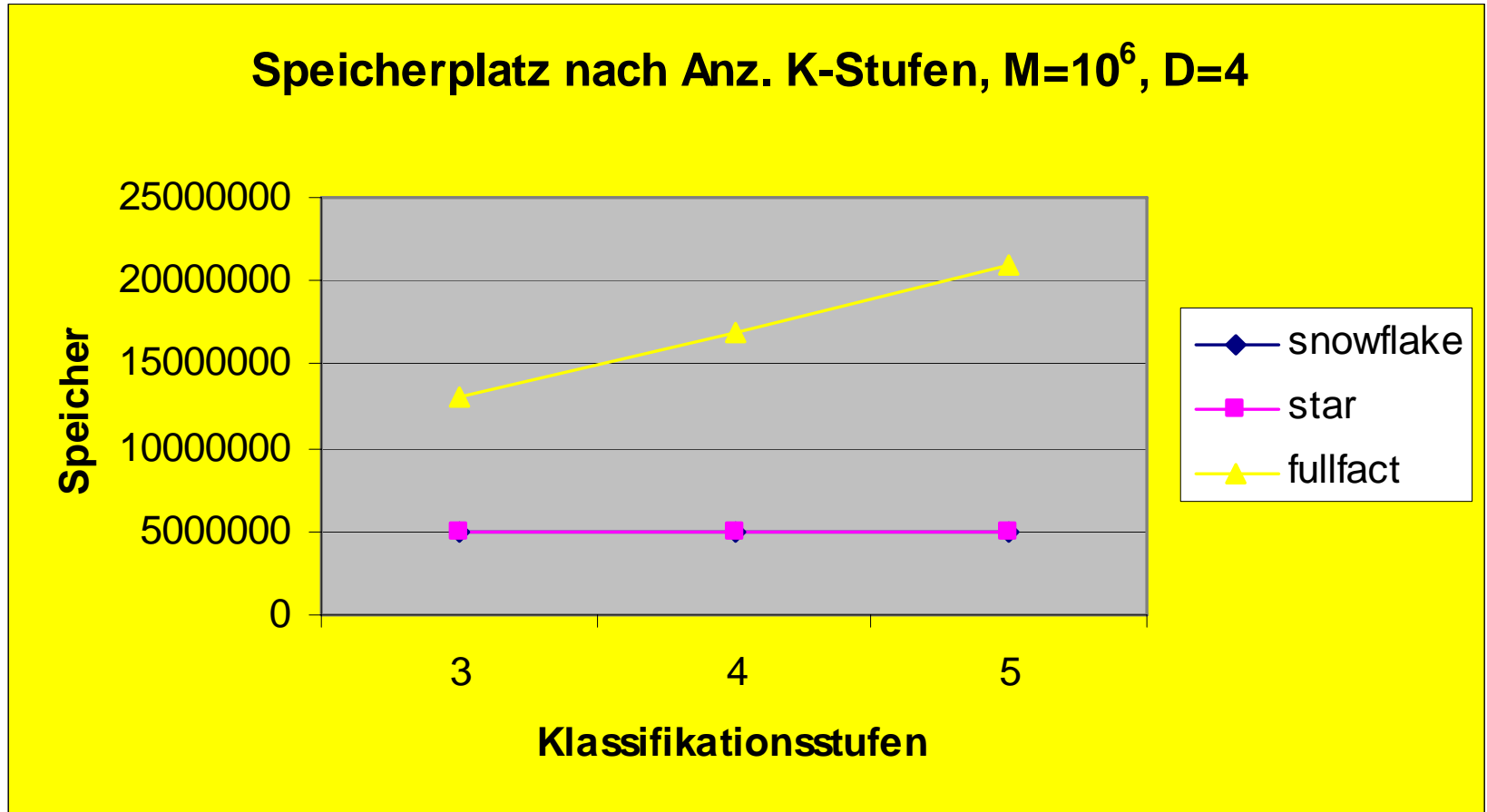
Speicherverbrauch 1



Speicherverbrauch 2



Speicherverbrauch 3



Fazit – Speicher und Query

- Speicherverbrauch Snowflake / Star praktisch identisch
 - Wenn Bedarf für Dimensionen vernachlässigbar
- Fullfact mit deutlich höherem Speicherverbrauch
 - Dafür minimale Anzahl Joins (im Idealfall 0)
- Laufzeitverhalten hängt von mehr Faktoren als dem Schema ab
 - Bereichs- oder Punktanfrage
 - Indexierung
 - Selektivität der Bedingungen
 - Gruppierung und Aggregation
 - ...
- ... aber **Joins sind tendenziell teuer**

Inhalt dieser Vorlesung

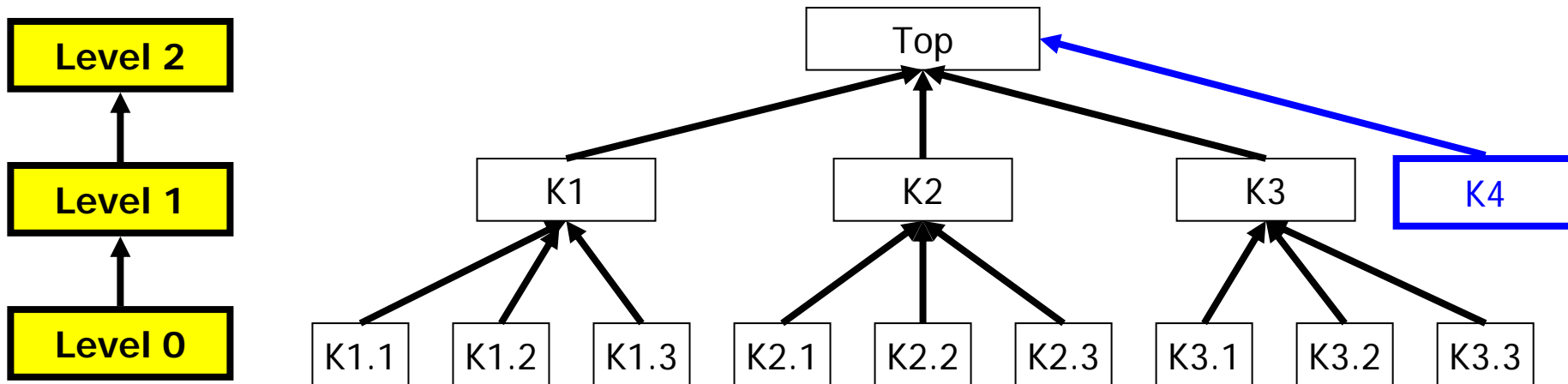
- Relationales OLAP (ROLAP)
 - Snowflake-, Star- und Fullfactschema
 - Speicherplatz und Queries
 - [Änderungen in den Dimensionen](#)
 - Schemavarianten
 - Beispiel: ROLAP in Oracle
- Multidimensionales OLAP (MOLAP)

Änderungen in Klassifikationshierarchie

- Auch Dimensionen verändern sich
- Mögliche Operationen
 - Gelöschte / neue Klassifikationsknoten
 - Gelöschte / neue Klassifikationsstufen
 - Neue Dimensionen
- Aufwand für Operationen abhängig von Modellierung
- Im folgenden: 4 Operationen

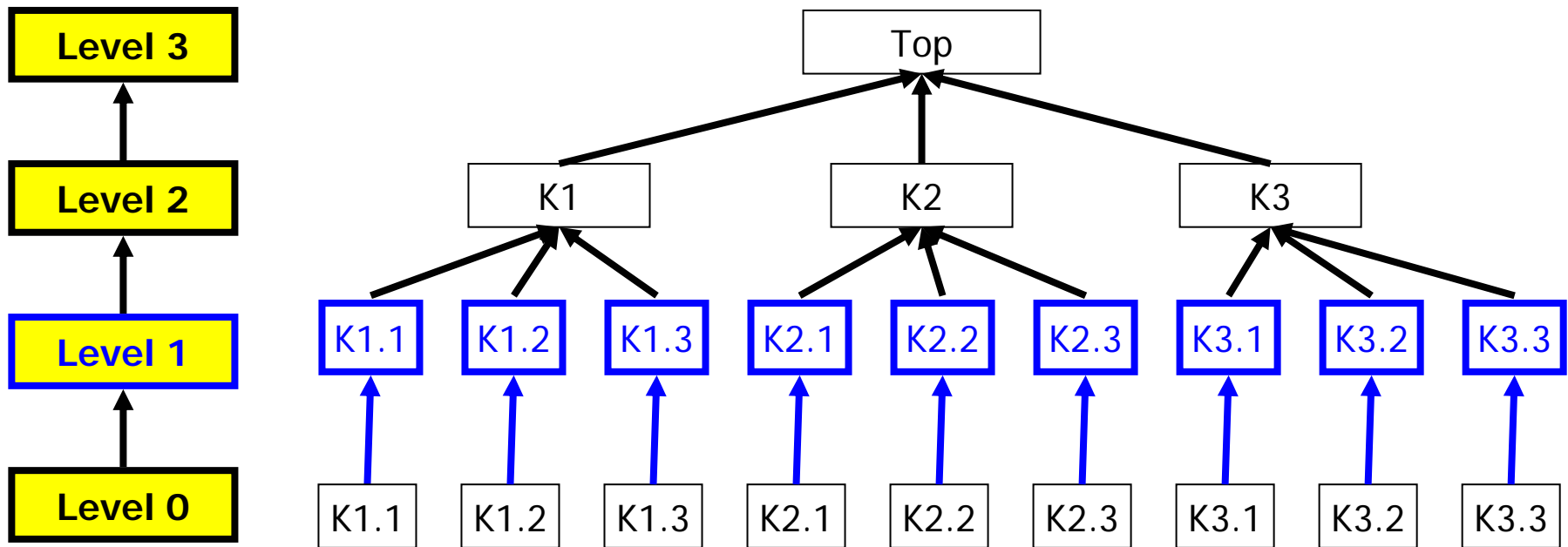
Änderung: InsN

Einfügen eines neuen Klassifikationsknoten ohne Nachkommen in einen Pfad auf Level $i \neq 0$



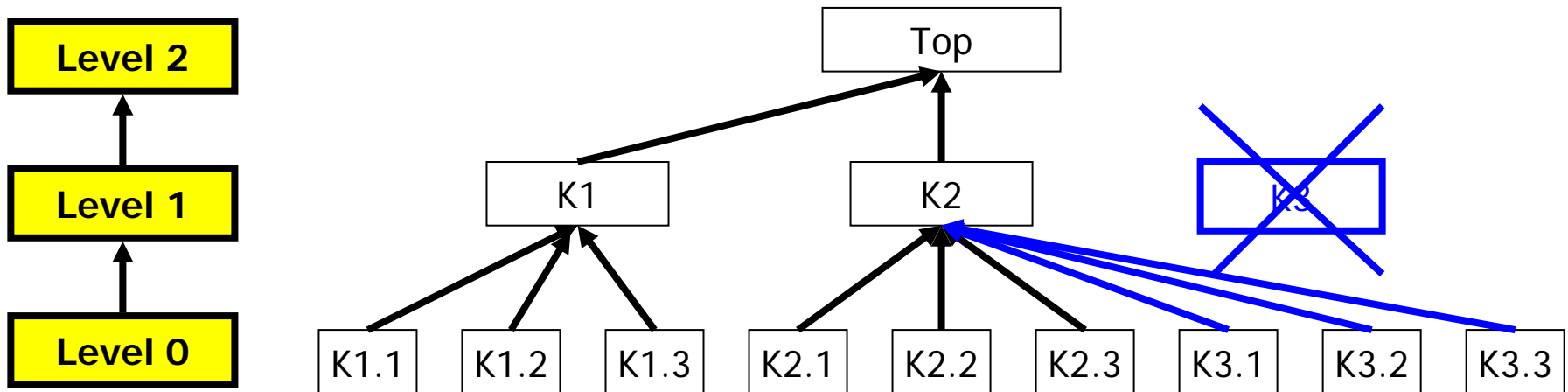
Änderung: InsS

Einfügen einer neuen Klassifikationsstufe in einen Pfad zwischen Level i und $i-1$ ($i > 1$) mit $3^{(k+1)-i}$ Klassifikationsknoten mit Umhängung Knoten Level $i-1$



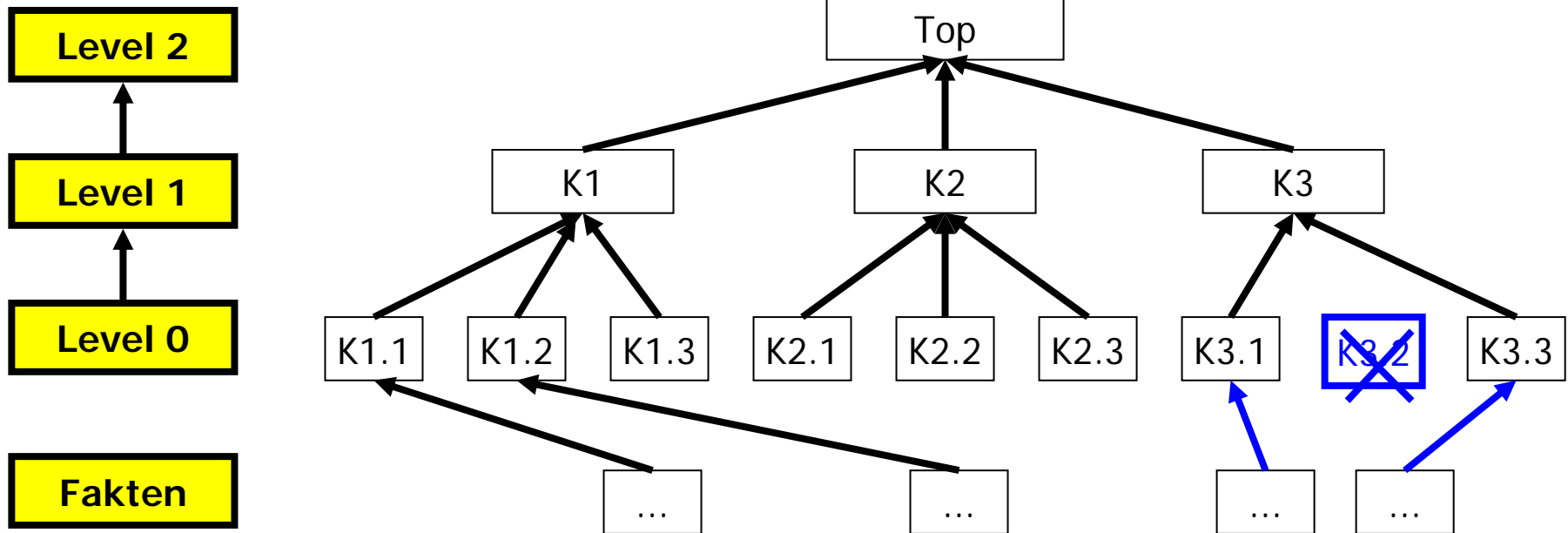
Änderung: DelN

Löschen eines Klassifikationsknoten in einem Pfad an Level $i \neq 0$; Umhängen der Nachkommen auf beliebigen anderen Knoten desselben Levels



Änderung: DelN₀

Löschen eines Klassifikationsknoten in einem Pfad an Level $i=0$; Umhängen der Fakten auf andere Knoten mit Level 0



Änderungskosten

	Snowflake	Star	Fullfact
InsN			
InsS			
DelN			
DelN ₀			

Änderungskosten

Semantisch fragwürdig: NULL in Knoten mit Level $j < i$

	Snowflake	Star	Fullfact
InsN	1 Insert	(1 Insert)	1 Insert
InsS	1 neue Tabelle, $3^{(k+1)-i}$ Updates (Umhängung)	1 neues Attribut, 3^k Updates (Wert des Attrib.)	1 neue Tabelle, 1 neues Attribut, m Updates (in Faktentabelle)
DelN	1 Delete, 3 Update	0 Delete, 3 Update	1 Delete, $m/3^{k-i}$ Updates (Level i: 3^{k-i} Knoten)
DelN₀	1 Delete, $m/3^k$ Update	1 Delete, $m/3^k$ Update	1 Delete, $m/3^k$ Updates

Fazit

- Insert/Delete in Dimensionen
 - Teuer bei Fullfakt
 - Moderat bei Star / Snowflake
- Insert/Delete von Blatt – Klassifikationsknoten
 - Werden durch FK aus Faktentabelle adressiert
 - Immer teuer (je nach Größe und Werteverteilung innerhalb der Dimension)
- Star Schema verlangt **balancierte Hierarchien**
 - Sonst NULL Werte in Knoten-IDs
 - Probleme beim Aggregieren/Gruppieren über NULLs

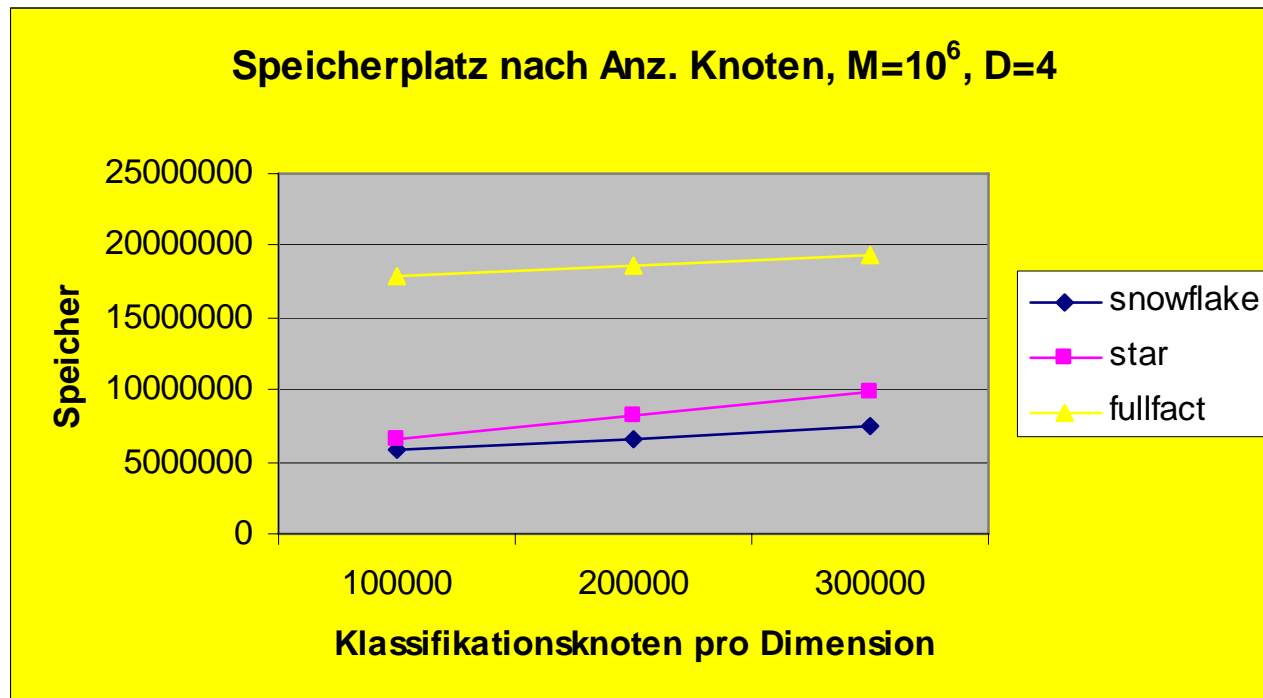
Inhalt dieser Vorlesung

- Relationales OLAP (ROLAP)
 - Snowflake-, Star- und Fullfactschema
 - Speicherplatz und Queries
 - Änderungen in den Dimensionen
 - Schemavarianten
 - Beispiel: ROLAP in Oracle
- Multidimensionales OLAP (MOLAP)

Entartete DWH

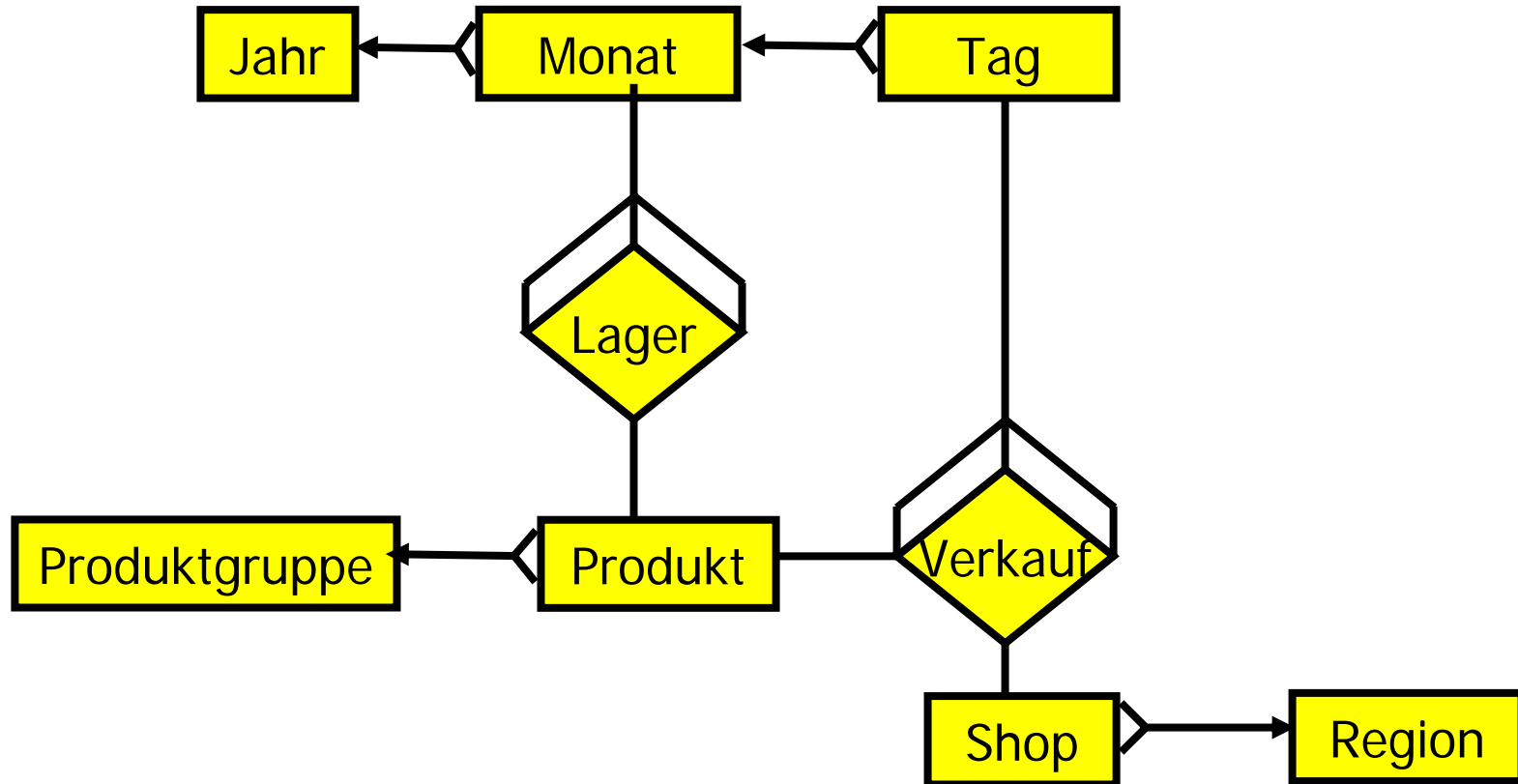
Anzahl Klassifikationsknoten nähert sich Anzahl Fakten

- Bsp.: Experimentelle Daten (Knoten = Proben, Werte = Ergebnisse von Tests)



Galaxy Schema

Mehrere Faktentabellen

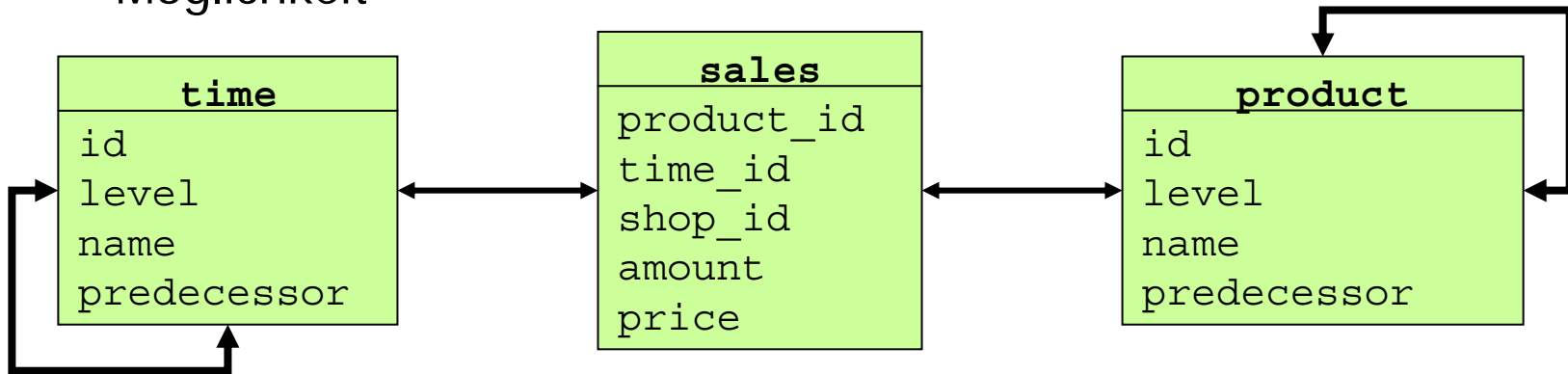


Konstellation Schema

- Modellierung aggregierter Daten
- Zwei Wege
 - Einbeziehung als spezielle Fakten in Faktentabelle
 - Modellierung in eigenen Tabellen
- Konstellation, „Fact Constellation“ Schema = Star Schema mit eigenen Summentabellen
- Siehe „Materialized Views“

Rekursive Modellierung

- Unbeschränkt lange, unbalancierter Klassenhierarchien
 - Beispiel: Stücklisten (Teil von ...)
 - Weder in Star noch Snowflake Schema möglich
 - Erweiterungen immer DDL Operationen
- Möglichkeit



- Probleme
 - Rekursive Anfragen oder viele Self-Joins (wie viele ?)
 - Keine individuellen Attribute pro Stufe
 - Eingeschränkte referenzielle Integrität
 - `sales.time_id` könnte auf Knoten in time mit `Level≠0` zeigen

Fazit

- Sehr viele Joins, kein Platzproblem, große, statische Klassifikationshierarchien mit wenigen Stufen
 - Fullfact Schema
- Viele Änderungen in Klassifikation, viele Klassifikationsstufen
 - Kein Fullfact Schema
- Viele Klassifikationsstufen, Aggregate über alle Stufen
 - Eher Star als Snowflake
- Unbeschränkt lange Klassifikationshierarchien
 - Rekursive Modellierung
- **Standard: Star oder Snowflake Schema**
 - Star Schema – Star Join

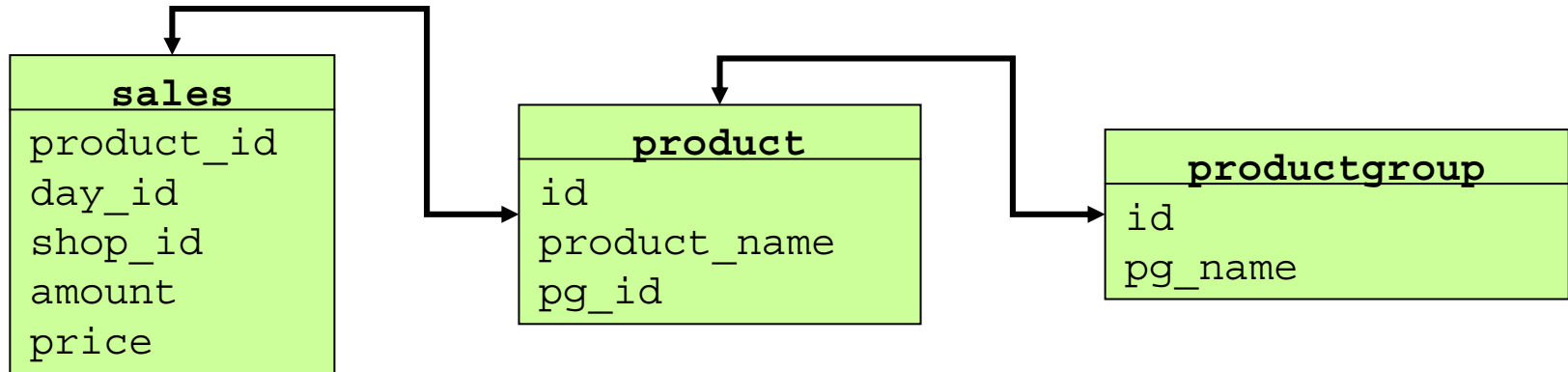
Inhalt dieser Vorlesung

- Relationales OLAP (ROLAP)
 - Snowflake-, Star- und Fullfactschema
 - Speicherplatz und Queries
 - Änderungen in den Dimensionen
 - Schemavarianten
 - [Beispiel: ROLAP in Oracle](#)
- Multidimensionales OLAP (MOLAP)

MDDM Konstrukte in Oracle

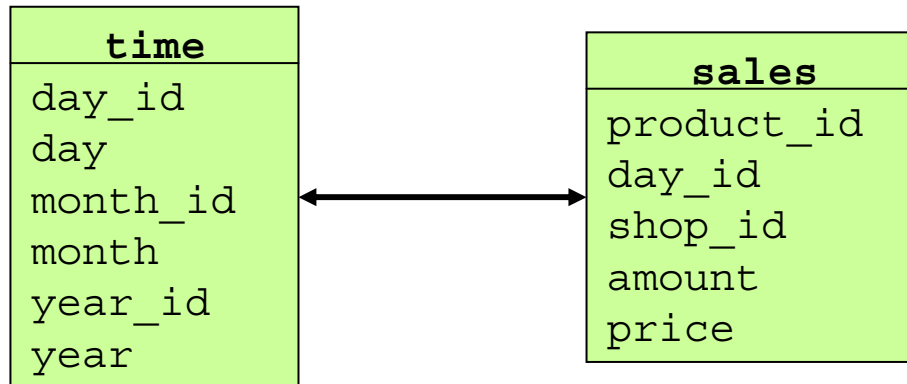
- Oracle kennt als **Sprachelemente**
 - Dimensionen
 - Klassifikationsstufen
 - Klassifikationspfade
- Definition mit DDL Befehlen
- Setzen auf existierende relationale Schema auf
 - Star oder Snowflake Schema

Beispiel: Snowflake Schema



```
CREATE DIMENSION s_pg
  LEVEL product IS (product.id)
  LEVEL productgroup IS (productgroup.id)
HIERARCHY pg_rollup (
  product CHILD OF productgroup
  JOIN KEY (product.id)
  REFERENCES productgroup
);
```

Beispiel: Star Schema



```
CREATE DIMENSION s_time
  LEVEL day IS (time.day_id)
  LEVEL month IS (time.month_id)
  LEVEL year IS (time.year_id)
HIERARCHY pg_rollup (
  day CHILD OF
  month CHILD OF
  year)
ATTRIBUTE day_id DETERMINES (day)
ATTRIBUTE month_id DETERMINES (month)
ATTRIBUTE year_id DETERMINES (year)
```

Multiple Pfade

Dimension

Klassifikationsstufen

```
CREATE DIMENSION times_dim
  LEVEL day IS TIMES.TIME_ID
  LEVEL month IS TIMES.CALENDAR_MONTH_DESC
  LEVEL quarter IS TIMES.CALENDAR_QUARTER_DESC
  LEVEL year IS TIMES.CALENDAR_YEAR
  LEVEL fis_week IS TIMES.WEEK_ENDING_DAY
  LEVEL fis_month IS TIMES.FISCAL_MONTH_DESC
  LEVEL fis_quarter IS TIMES.FISCAL_QUARTER_DESC
  LEVEL fis_year IS TIMES.FISCAL_YEAR
HIERARCHY cal_rollup (
  day CHILD OF
  month CHILD OF
  quarter CHILD OF
  year )
HIERARCHY fis_rollup (
  day CHILD OF
  fis_week CHILD OF
  fis_month CHILD OF
  fis_quarter CHILD OF
  fis_year )
<attribute determination clauses>...
```

Klassifikationspfad 1

Klassifikationspfad 2

Eigenschaften

- Erwartet vollständige und überlappungsfreie Hierarchien
 - „The *child_key_columns* must be non-null and the *parent key* must be unique and non-null. You need not define constraints to enforce these conditions, but *queries may return incorrect results if these conditions are not true.*“
- Analyse und Anzeige über PL/SQL Packages
 - **DEMO_DIM, DBMS_OLAP, ...**
- **Attribute ... Determines** Klausel
 - Spezifiziert funktionale Abhängigkeiten **innerhalb einer Tabelle**
 - Wird benutzt, um ggf. Präaggregation über abhängige Werte benutzen zu können
- Benutzung
 - Für **Query Rewrite** (siehe „Materialisierte Sichten“)
 - Für Client-Software / Visualisierung / Navigation

Inhalt dieser Vorlesung

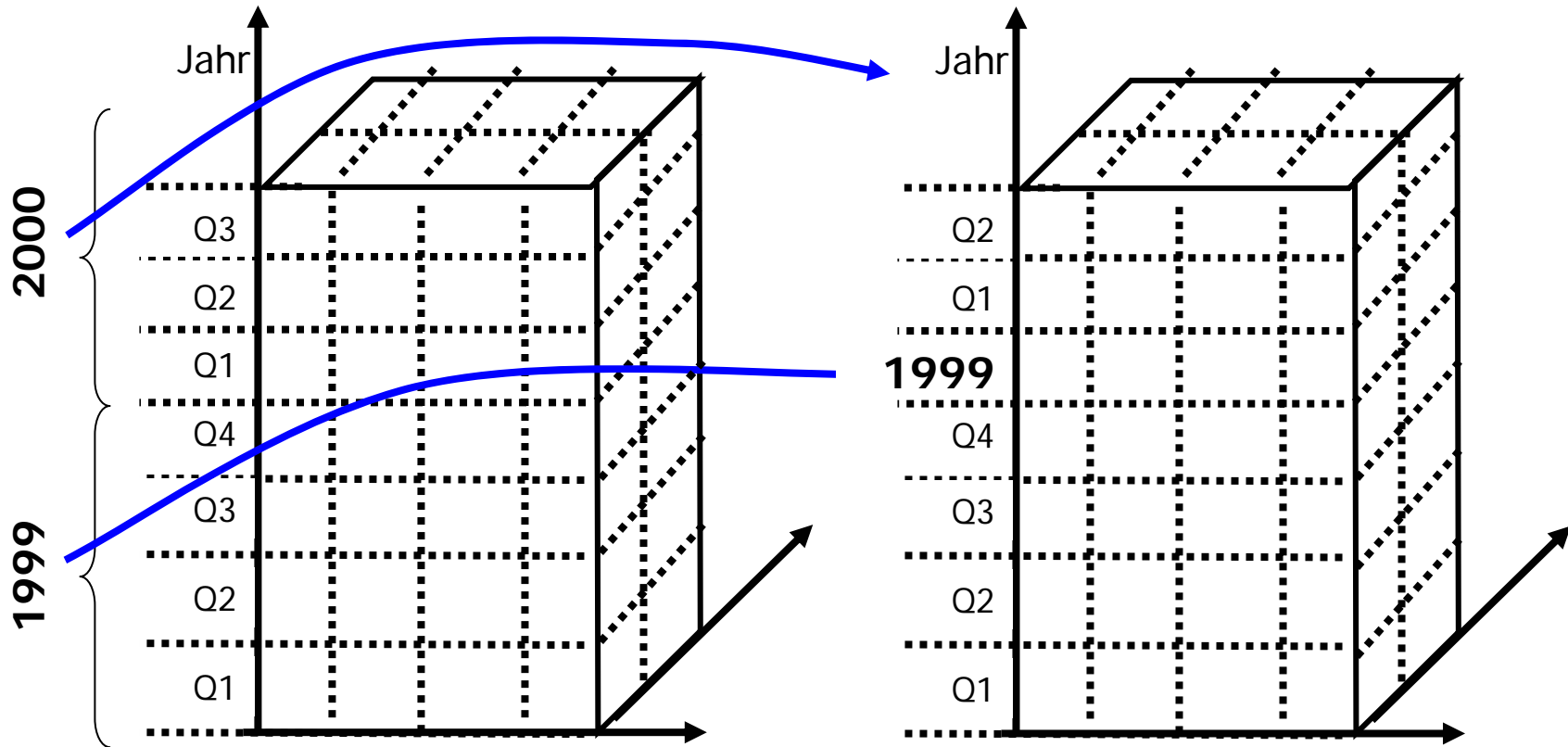
- Relationales OLAP (ROLAP)
 - Snowflake-, Star- und Fullfactschema
 - Speicherplatz und Queries
 - Änderungen in den Dimensionen
 - Schemavarianten
 - Beispiel: ROLAP in Oracle
- Multidimensionales OLAP (MOLAP)

Multidimensionales OLAP

- Grundidee
 - Speicherung multidimensionaler Daten ...
 - ... in [multidimensionalen Arrays](#)
- Kommerzielle Produkte verwenden proprietäre (und geheime) Techniken
 - Siehe auch [multidimensionale Indexstrukturen](#) (später)
 - OLAP will vor allem aggregieren
 - [Range-Queries und Speicherung von prä-aggregierten Daten](#)
- Im folgenden nur
 - 1 Würfel, 1 Fakt
 - Keine Betrachtung von Knotenattributen
 - Klassifikationsknoten haben nur diskrete Werte
 - Bei numerischen Knoten: Diskretisierung

Klassifikationshierarchien

Modellierung als eingebettete Knoten der untersten Stufe



Array Adressierung

- Sei $d_i = |D_i|$, d Dimensionen,
- b : Speicherplatz pro Attribut (Key oder Wert)
- Linearisierte Darstellung
 - $\langle 1,1,1 \rangle, \langle 1,1,2 \rangle, \dots, \langle 1,1,d_1 \rangle, \langle 1,2,1 \rangle, \dots$
- Offset der Zelle $\langle a_1, \dots, a_n \rangle$

$$b * \sum_{i=1}^d \left((a_i - 1) * \prod_{j=1}^{i-1} d_j \right)$$

Arrayspeicherung - Probleme

- **Naive MOLAP**: Unterschiedliche Performanz je nach Dimensionsreihenfolge in Array / Anfrage
 - Daten liegen nach D_n geordnet auf Platte
 - Zugriff auf $\langle 2,2,X \rangle$ - sequentieller Read
 - Zugriff auf $\langle X,1,3 \rangle$ oder $\langle 3,X,3 \rangle$ - Zugriff auf viele Blöcke
 - **Multidimensionale Datenstrukturen** notwendig
- Aber: RDBMS hat ähnliche Probleme
 - Zugriff abhängig von Anordnung der Tupel auf der Platte
 - Forschung: „Column-wise“ relational databases
- Speicherung verschiedener Werte pro Zelle
 - Mehrere Verkäufe von „CocaCola“ an einem Tag in einem Shop
 - Schwierig, da **keine Obergrenze für Anzahl**
 - Auflösung durch Listen etc.
 - Modell vor allem für aggregierte Werte geeignet
- Alternative
 - ROLAP für den feinsten Granularitätslevel
 - MOLAP für höhere Level

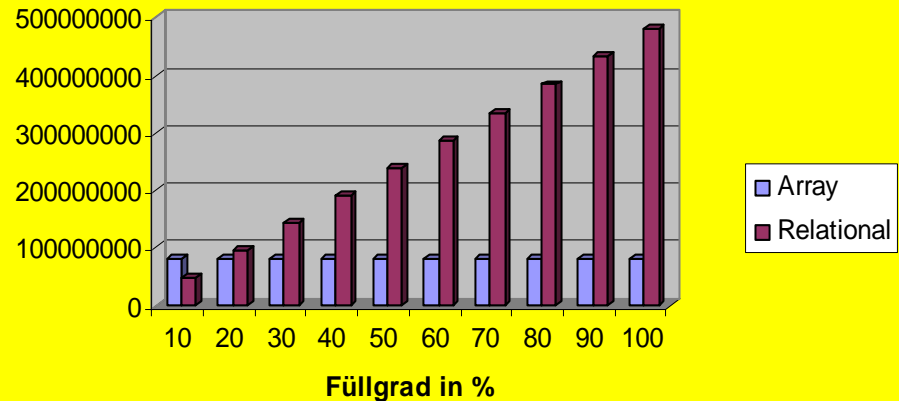
Speicherverbrauch

	Array	Relational (Star Schema)
Speicherung Koordinaten	Implizit (Linearisierung)	Explizit (redundant)
Leere Zellen	Belegen Platz	Belegen keinen Platz
Neue Klassifikations- knoten	Komplette Reorganisation Starkes Wachstum im Speicherverbrauch	Neue Zeile in Dimensionstabelle Kaum Wachstum im Speicherverbrauch
Speicher- verbrauch	$b * \prod_{i=1}^d d_i$	$b * m * d$ (plus Measures)

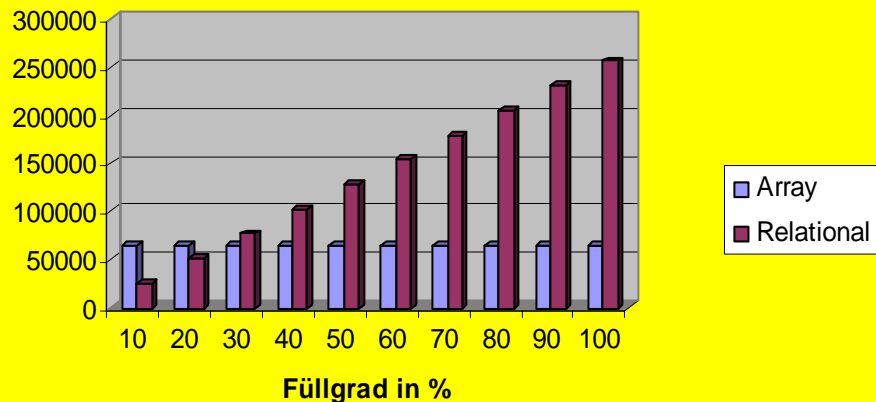
Vergleich Speicherplatz

- Faktoren
 - Füllgrad
 - k: Anz. Knoten
 - d: Anz. Dimensionen
- Schon bei **Füllgraden** ~15% ist Array platzeffizienter
- Aber: Tatsächlicher Füllgrad ist bei vielen Dimensionen bzw. Klassifikationsknoten oft erstaunlich gering
 - Bei weitem **nicht alle Kombinationen** vorhanden

Speicherplatz nach Füllgrad, b=8, k=100, d=5



Speicherplatz nach Füllgrad, b=8, k=20, d=3



Sparse Matrices

- Viele Klassifikationsstufen und -knoten
 - Dünn besetzte Matrizen
 - Geringer Füllungsgrad
 - Ineffiziente Speicherplatznutzung
 - **Überflüssiges I/O** (Auch NULLs müssen gelesen werden)
- Komprimierung
 - Run-Length Encoding (RLE): Schlecht indizierbar
 - 2-Ebenen Speicherung
 - Ebene 1: Array mit dünn besetzten Dimensionen
 - Ebene 2: Arrays mit dicht besetzten Dimensionen
 - Gewinn: Viele Ebene 2 Blöcke leer – weglassen
 - Parallelität zu **multidimensionalen Indexstrukturen**

Fazit MOLAP

- Verschiedene Verfahren möglich
- Kein Standard vorhanden
- Verwendung in **proprietären Produkten**
- Volksmeinung
 - Schnell für kleine – mittlere Datenmengen
 - Schlechte Skalierbarkeit (> 50GB)
 - Wenn der Hauptspeicher ausgeht
- Typischer Einsatz
 - (Regelmäßige) Snapshots des (ROLAP) Cubes auf Client in MOLAP Datenbank laden
 - Vorberechnung der notwendigen Aggregate
 - Read-Only
 - Zweck ist ein sehr schnelles, **intuitives User-Interface**
 - Real-Time OLAP