Data Engineer

DWH: Star Schema Grundlagen, Datenmodellierung, Erstellung Star Schema in RDBMS

..

Datenbankentwurf: Schritte

- Entwurfsschritte
  - Anforderungsanalyse
  - konzeptioneller DB-Entwurf
    - → konzeptionelles Schema (unabhängig vom Zieldatendatenmodell)
  - logischer DB-Entwurf
    - → logisches Schema (in konkretem Datenmodell)
  - physischer DB-Entwurf
    - → internes/physisches Schema (in konkretem Datenbanksystem)
- iterativer Entwurfsprozess
- konzeptueller und logischer DB-Entwurf nicht immer leicht zu trennen

Datenbankentwurf: Vergleich

<table>
<tr><th></th><th>Relationaler DB-Entwurf</th><th colspan="2">Multidimensionaler DB-Entwurf</th></tr>
<tr><td>konzeptuelles Schema</td><td>E/R-Modellierung, UML</td><td colspan="2">ME/R, mUML, …</td></tr>
<tr><td rowspan="2">logisches Schema</td><td rowspan="2">Relationen mit Attributen</td><td colspan="2">Datenwürfel mit Summenattributen: Fakten und Kennzahlen</td></tr>
<tr><td colspan="2">Dimensionshierarchien mit Kategorienattributen</td></tr>
<tr><td>internes/physisches Schema</td><td>Speicherorganisation: Indexstrukturen, Partitionierung, …</td><td>Relationale Speicherorganisation (ROLAP)</td><td>Multidimensionale Speicherorganisation (MOLAP)</td></tr>
</table>

alfatraining

Data Warehousing

Data-Warehouse-Entwurf
- Konzeptueller Datenbankentwurf -

Datenmodell für Data Warehouses

alfatraining

Woran ist ein Analyst interessiert?

- an **Kennzahlen**
  - Umsatz, Gewinn, Menge, Kilometer
- gruppiert nach **Dimensionen**
  - **Produkt**: Produktkategorie, Produktklasse, Produktgruppe, Produktabteilung
  - **Region**: Filiale, Gemeinde, Landkreis, Bundesland, Staat
  - **Zeit**: Tag, Woche, Monat, Quartal, Jahr
  - **Auftraggeber/Kunde**: Alter, Geschlecht, Beruf, Einkommen, …
  - …
  - ... und allen sinnvollen **Kombinationen**

Datenmodell für Data Warehouses

- Datenmodell muss zwei Arten von Informationen unterscheiden:
  - **Quantifizierende** Informationen (Kennzahlen)
  - **Qualifizierende** Informationen: (Dimensionen)

Quantifizierende Informationen

Quantifizierende Informationen sind oft **numerisch** und bestehen aus:

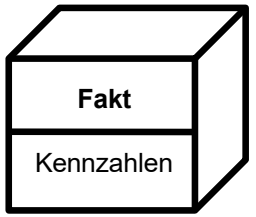
- **Fakten** („Basiskennzahlen“)
  - z. B.: Einnahmen aus einer Dienstleistung
- **Kennzahlen** („abgeleitete Kennzahlen“)
  - Berechnungsvorschrift über existierende Fakten
    - Zählung: `count()`
    - Summierung: `sum()`
    - Mittelwertbildung: `avg()`
    - Minimum: `min()`
    - Maximum: `max()`
    - Varianz
    - Standardabweichung
    - …

Konzeptueller DB-Entwurf

- Erweiterung bestehender Entwurfstechniken an Anforderungen durch Multidimensionalität
  - ME/R
    multidimensionales E/R
  - mUML
    multidimensionales UML

ME/R-Modellierung

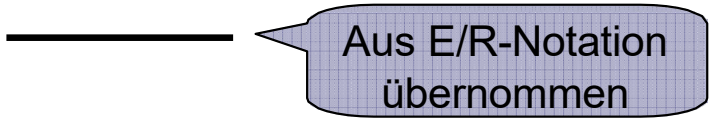
- **Erweiterung** des E/R-Modells um **multidimensionale** Konstrukte:
  - Fakt



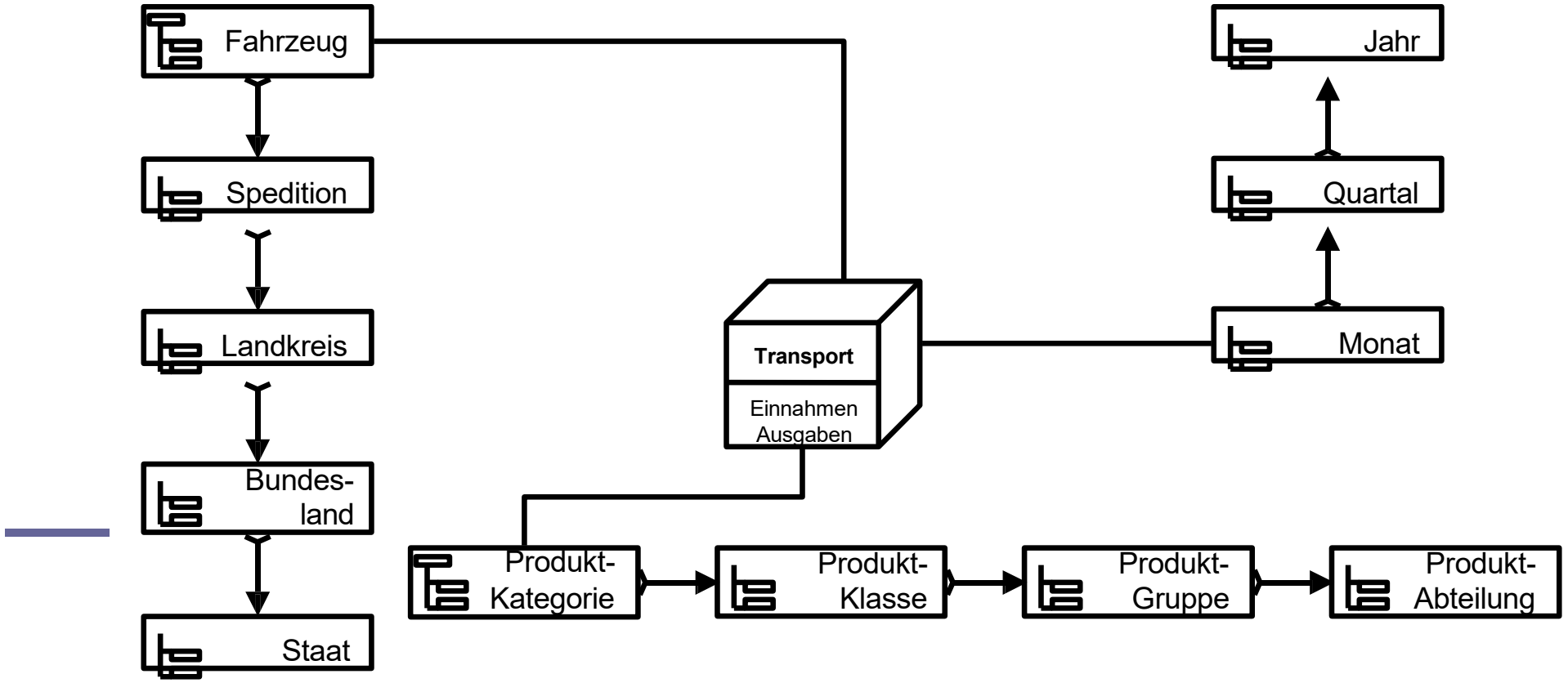
  - Dimensionsebene

  - Klassifikationsbeziehung

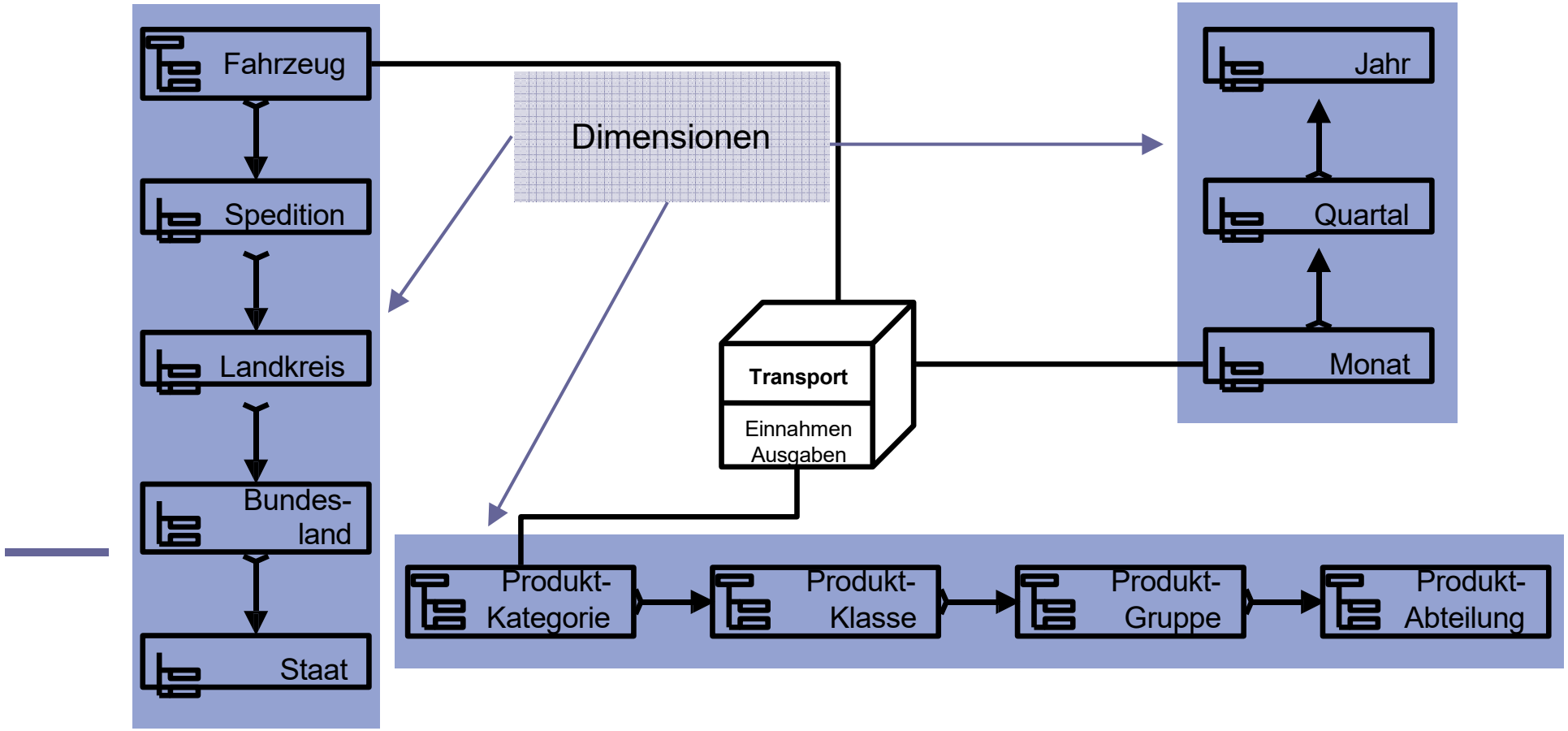
  - Faktbeziehung



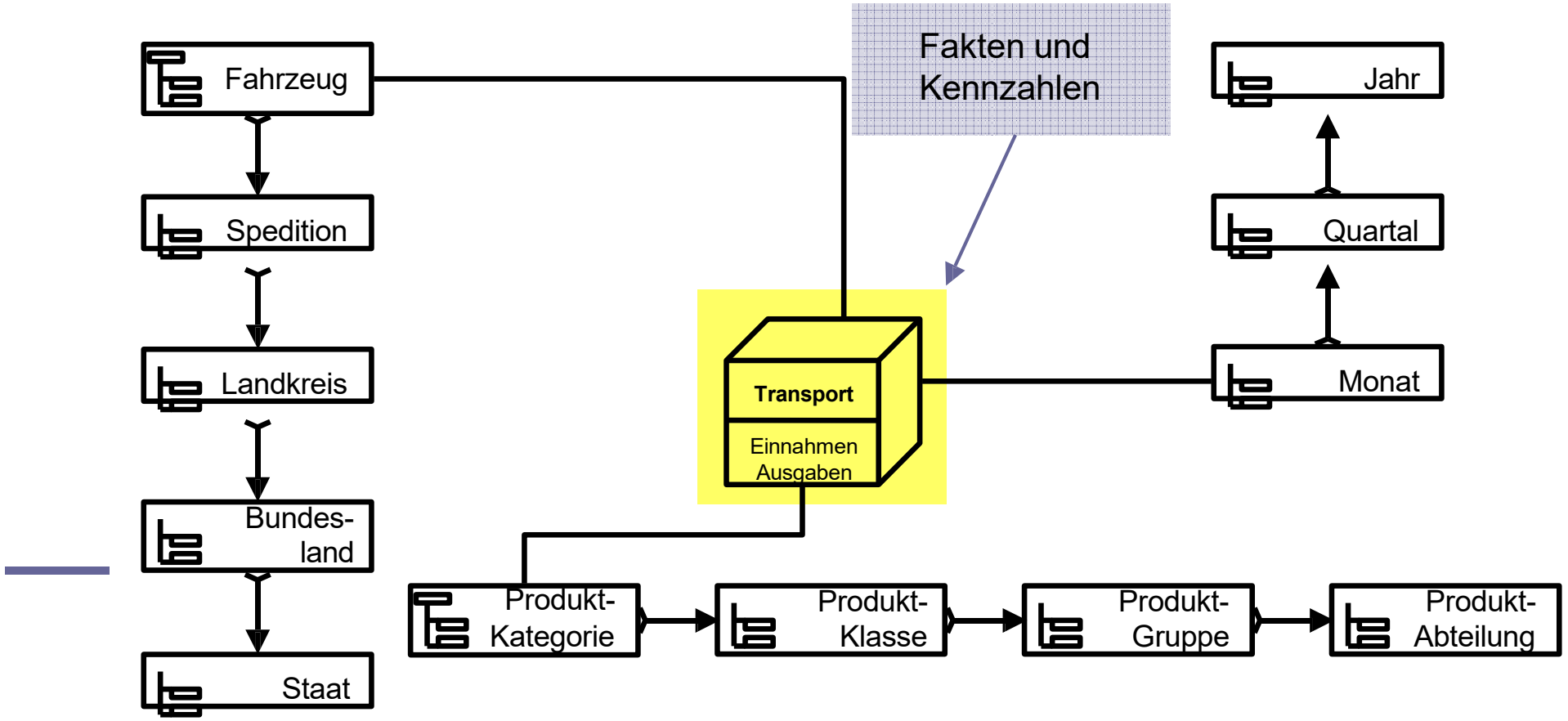
ME/R-Modellierung: Beispiel



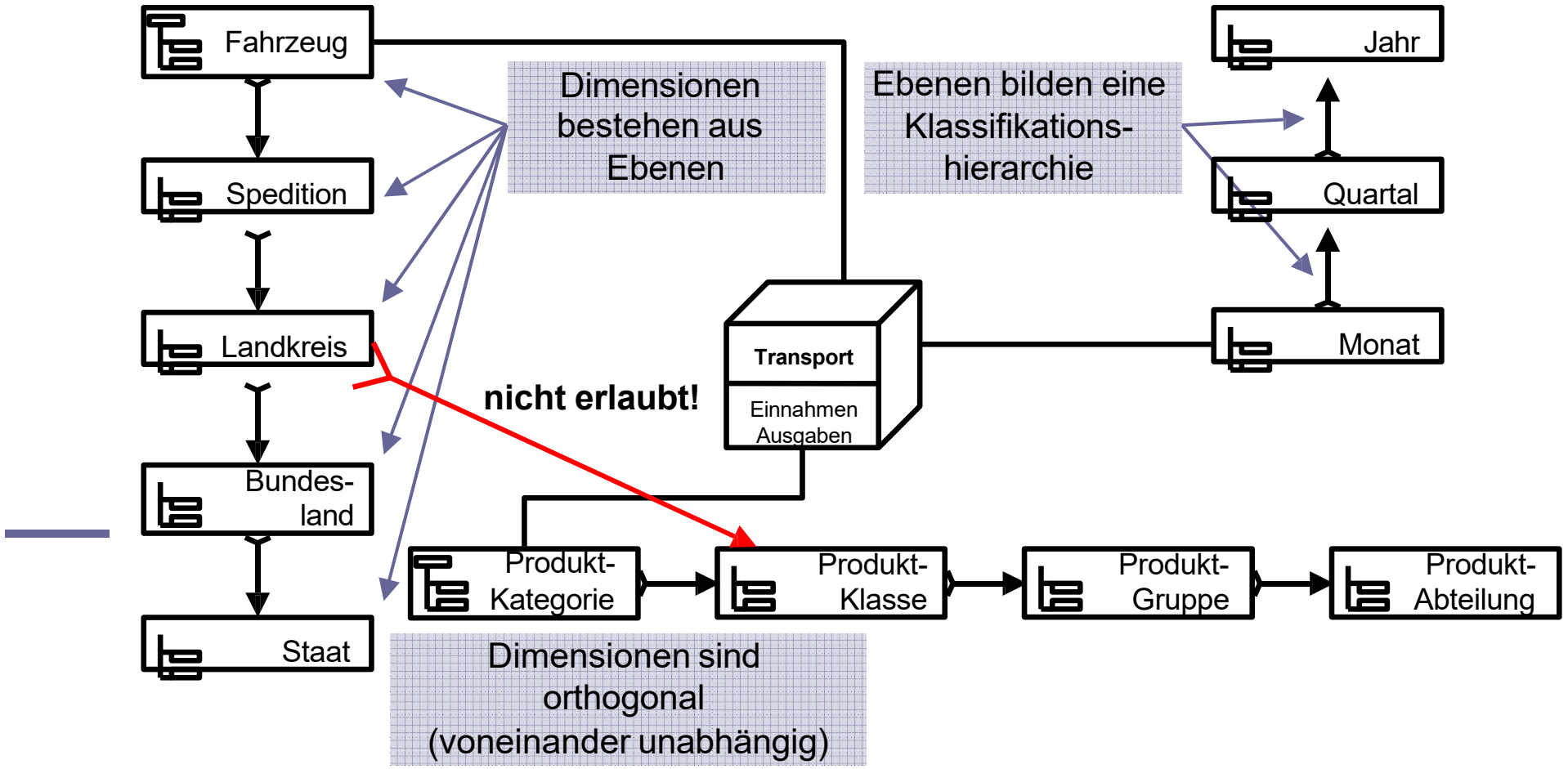
ME/R-Modellierung: Beispiel



ME/R-Modellierung: Beispiel

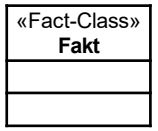


ME/R-Modellierung: Beispiel

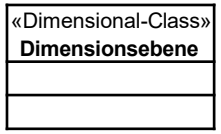


mUML-Modellierung

- **Erweiterung** des UML-Klassendiagramms um **multidimensionale** Konstrukte durch **Steoreotypen**:
  - Fakt

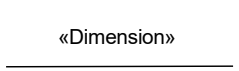


  - Dimensionsebene

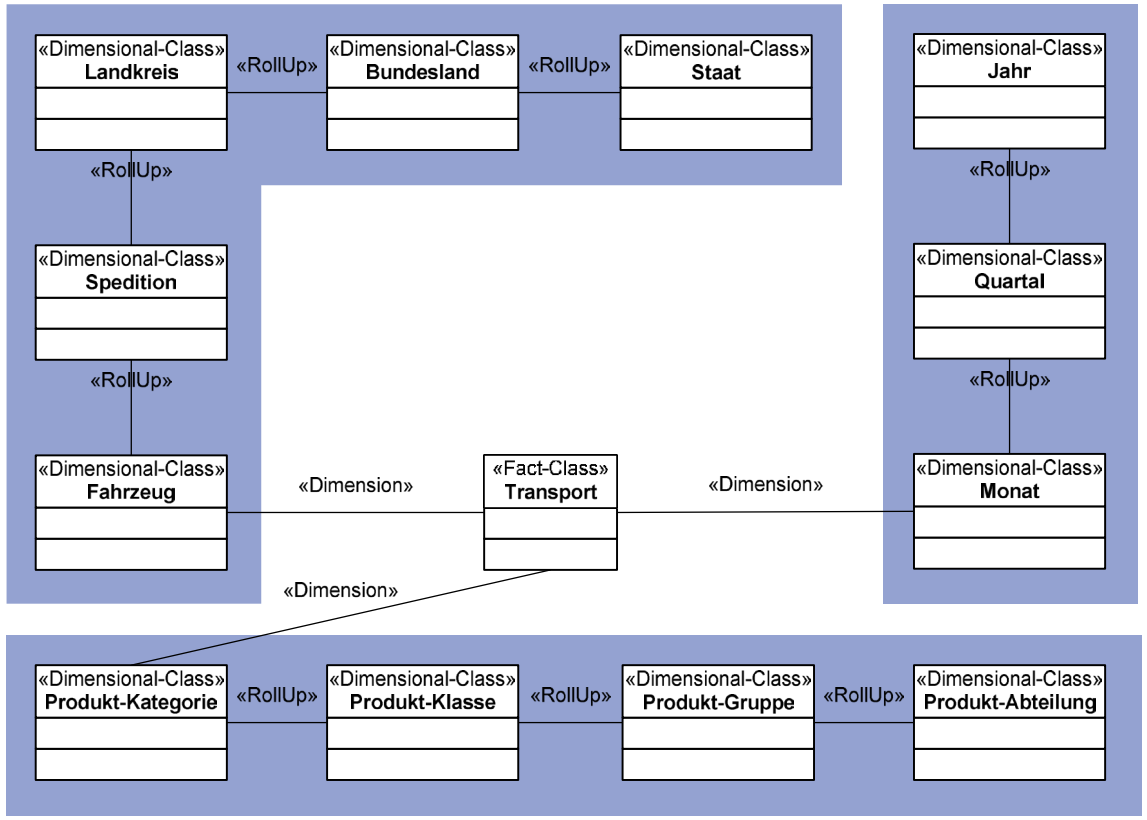


  - Klassifikationsbeziehung

  - Faktbeziehung



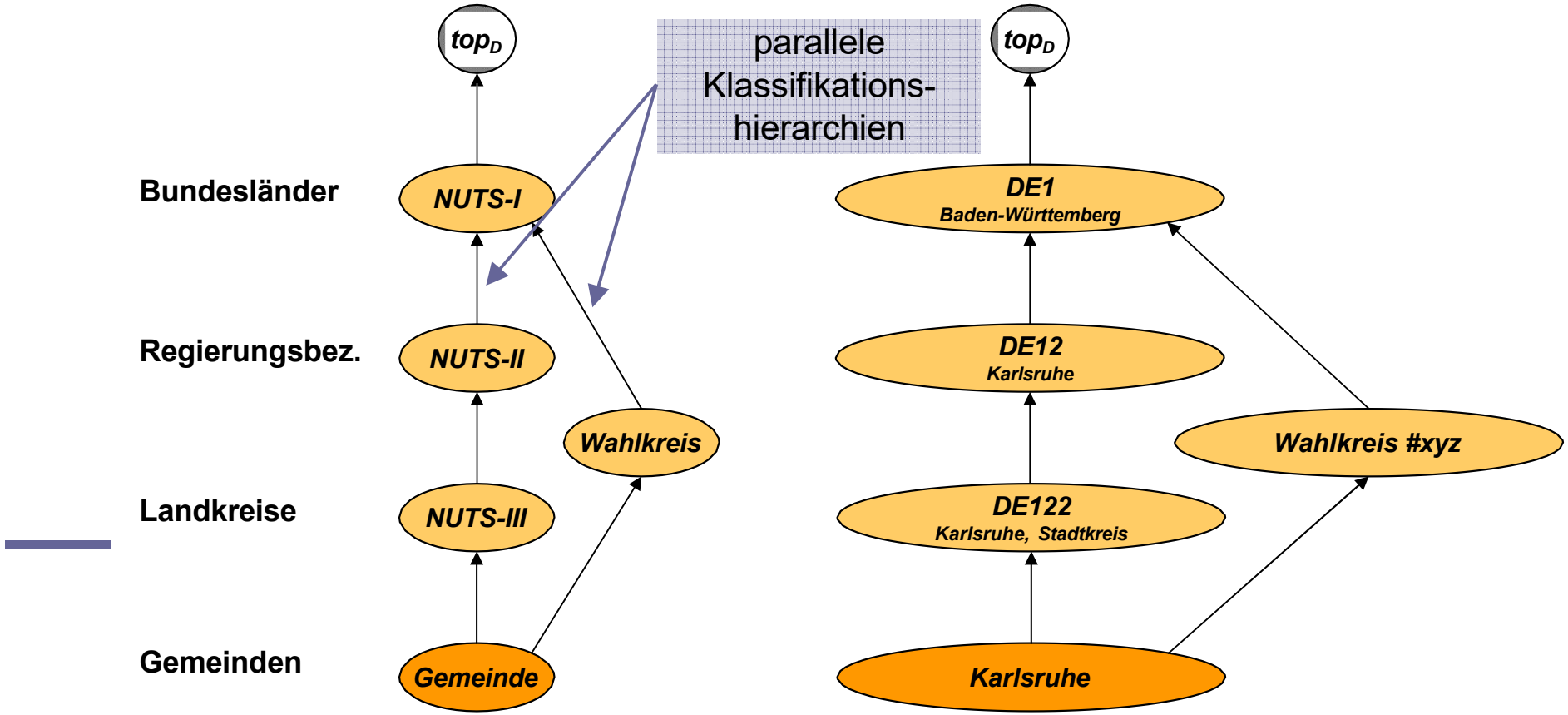
mUML-Modellierung: Beispiel



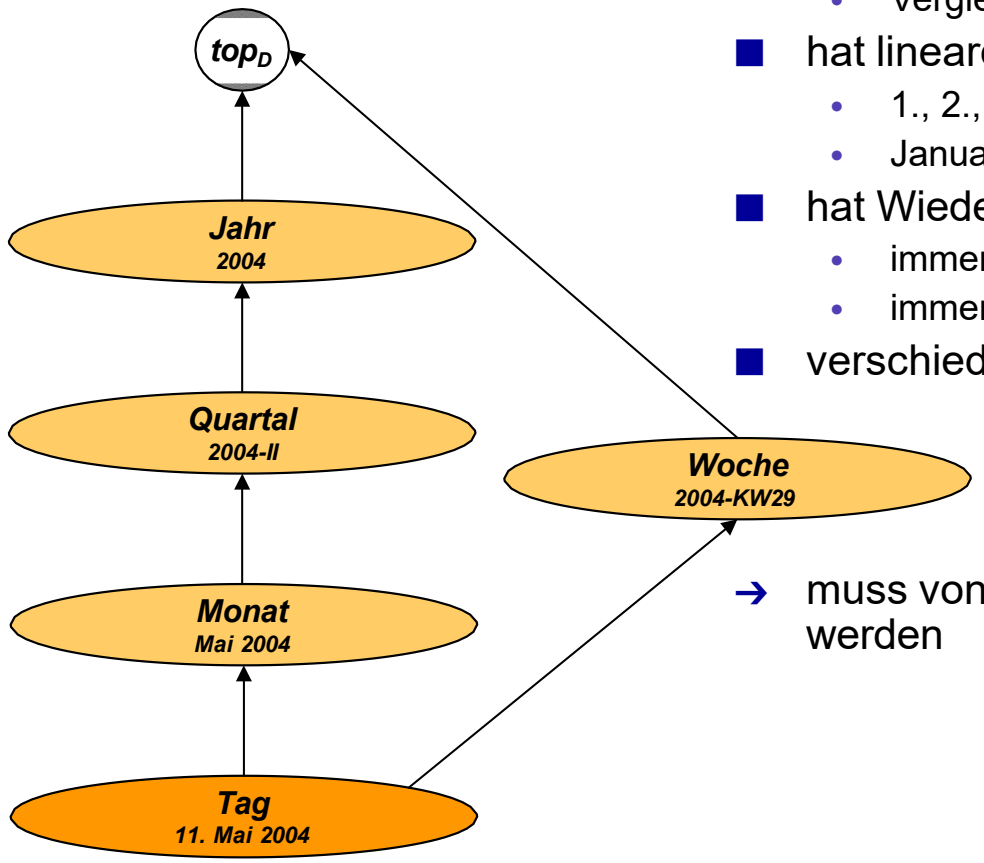
Qualifizierende Informationen

- Dimensionen
  - Dimensionen sind **orthogonal** (voneinander unabhängig)
  - Dimensionen bestehen aus **Dimensionsebenen**
  - Dimensionsebenen bilden eine **Klassifikationshierarchie**

Dimension: Ebenen



Dimension: Zeit



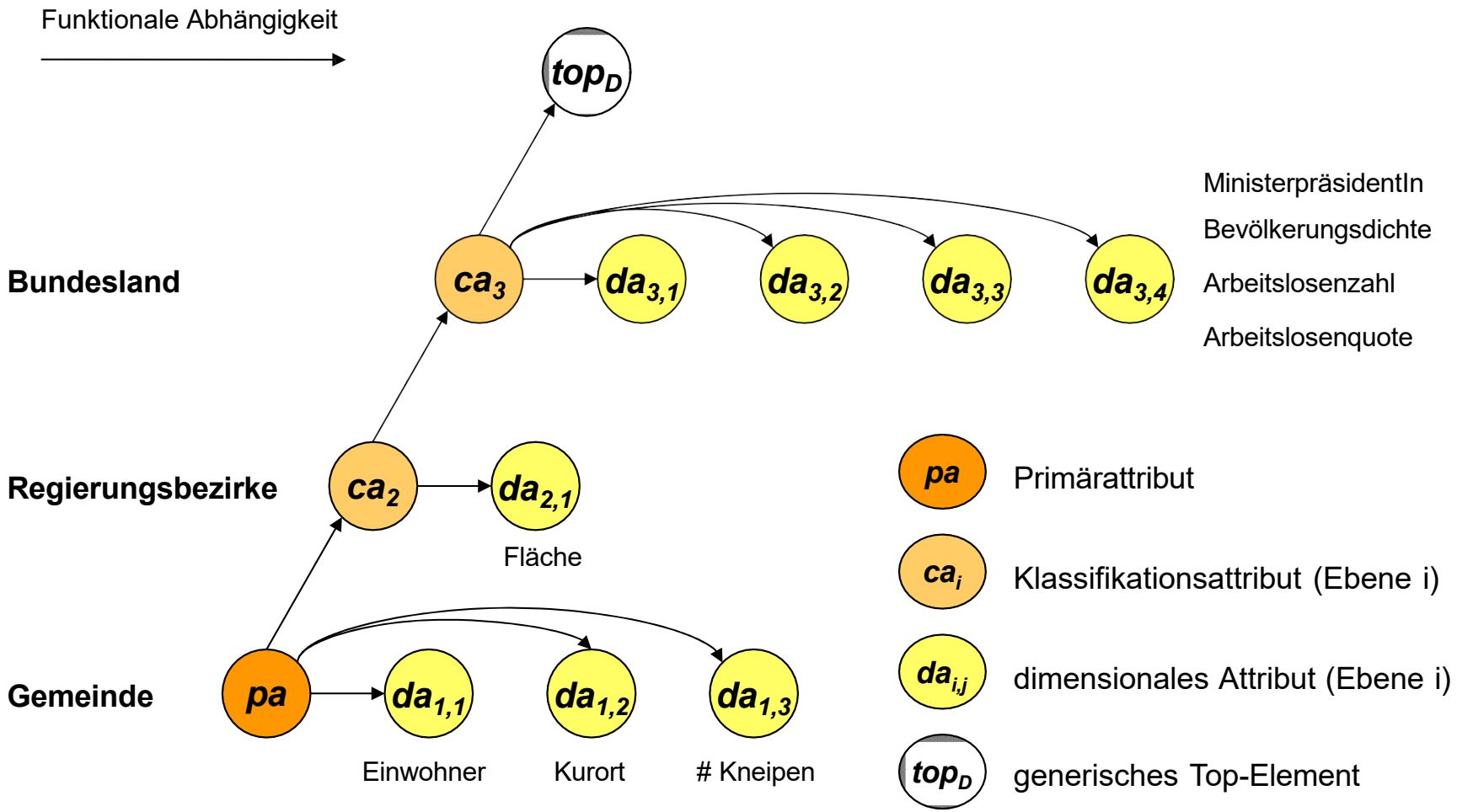
- kommt in fast jeder Anfrage vor:
  - Vergleich Quartal mit Vorjahresquartal
- hat linearen Charakter
  - 1., 2., 3., …
  - Januar, Februar, März, …
- hat Wiederholungscharakter
  - immer montags
  - immer im 4. Quartal
- verschiedene Klassifikationspfade denkbar

→ muss von jedem Analysesystem unterstützt werden

Funktionale Abhängigkeiten

- Funktionale Abhängigkeit
  = functional dependency (FD)
  - Zwischen zwei Attributen A und B existiert eine funktionale Abhängigkeit (**A☐ B**) genau dann, wenn **für jedes a aus A genau ein b aus B** existiert.
  - Für Attribute A, B, C:
    - Man schreibt AB☐ C, wenn für jedes Paar (a,b) mit a aus A und b aus B genau ein c aus C existiert.
    - Man schreibt A☐ BC, wenn A☐ B und A☐ C gilt.

Dimension: Kategorieattribute



alfatraining

Data Warehousing

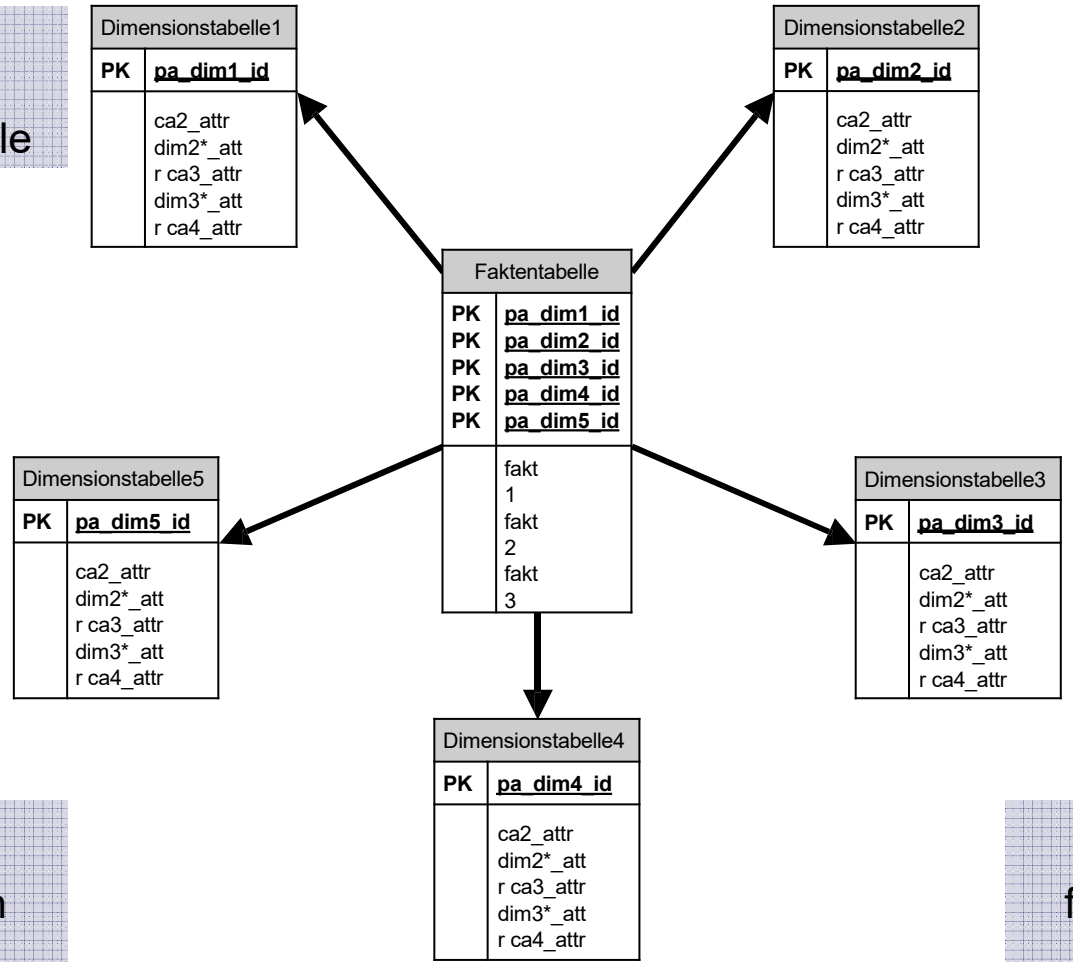
Data-Warehouse-Entwurf
- Logischer Datenbankentwurf -

Relationale Abbildung

- Wie lassen sich Kennzahlen und Dimensionen gut in einem relationalen Schema abbilden?
  - möglichst gut verständlich
  - auch relationales Schema sollte Zusammenhänge (z. B. Fakt- und Klassifikationsbeziehungen) abbilden
  - möglichst **performante Anfragebearbeitung**
  - möglichst redundanzfrei

Star-Schema

Dimensionen **sternförmig** um Faktentabelle



1 Tabelle pro Dimension

1 Tabelle für alle Fakten

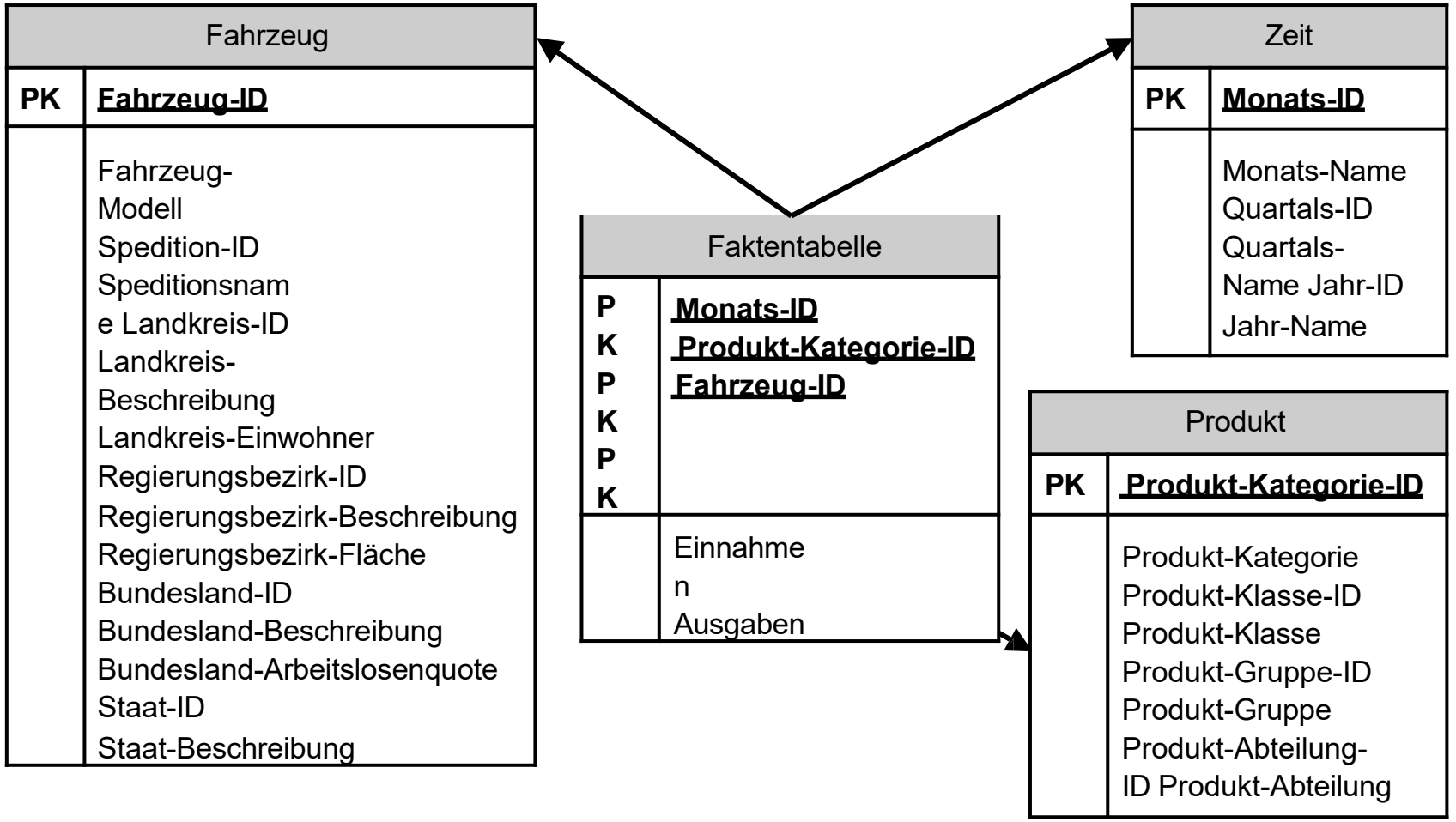
Faktentabelle

- Faktentabelle = fact table
- enthält Analysegegenstand, eigentliche Geschäftsdaten
- Tupel besteht aus
  - Zeigern auf Dimensionstabellen (Fremdschlüssel), die den Kontext des Tupels eindeutig bestimmen
  - den eigentlichen Kennzahlen
- Schlüssel der Faktentabelle =Gesamtheit der Dimensionszeiger
- Faktentabelle kann viele Millionen Tupel enthalten

Dimensionstabelle

- Dimensionstabelle = dimensional table
- pro Dimension existiert eine Dimensionstabelle
- Dimensionstabelle enthält
  - einen eindeutigen Schlüssel (Primärattribut), z.B. Produktnummer, Datums-ID, ...
  - weitere (beschreibende) Attribute der Dimension (Klassifikations- und dimensionale Attribute)
- Dimensionstabelle deutlich kleiner als Faktentabelle

Star-Schema: Beispiel



Star-Schema

- Eigenschaften
  - sehr übersichtlich
  - einfach erweiterbar
  - recht performante Anfragebearbeitung
- Probleme
  - **denormalisierte Dimensionstabellen**
    eventuell immer noch groß und redundant
  - **Dimensionsklassifikationen nicht** explizit **modelliert**

alfatraining

Data Warehousing

Data-Warehouse-Entwurf
- Physischer Datenbankentwurf -

Speicherorganisation

- ROLAP
  relationale Speicherorganisation
- MOLAP
  multidimensionale Speicherorganisation
- HOLAP
  hybride Speicherorganisation

ROLAP

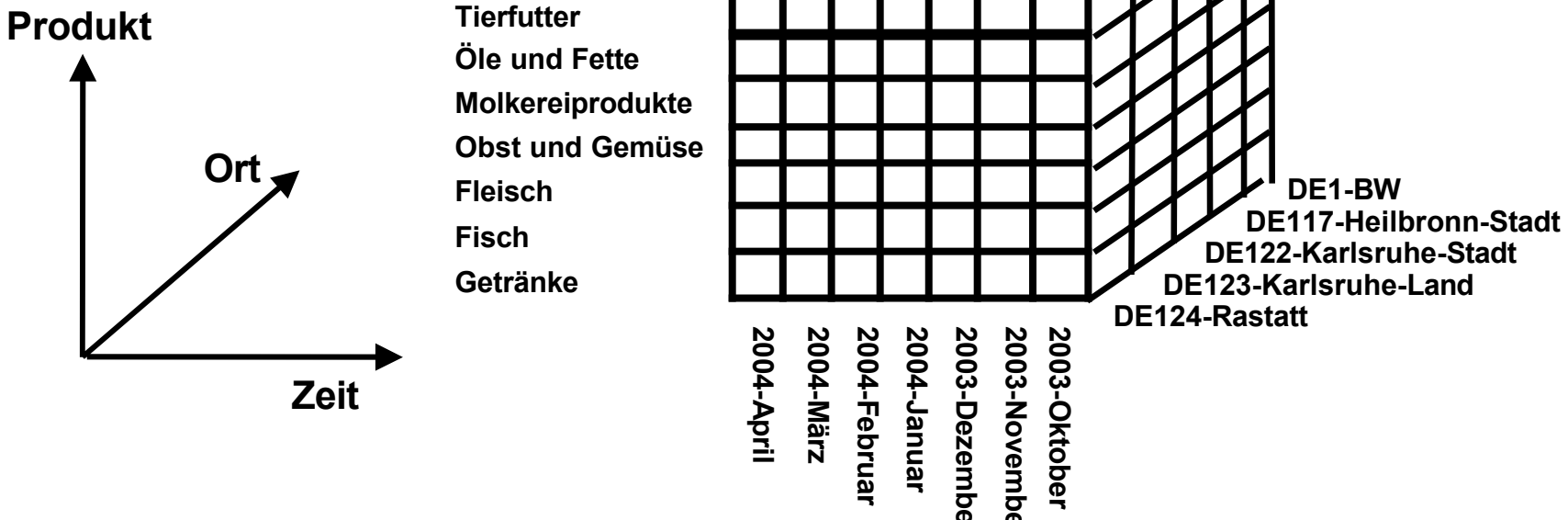
- ROLAP = Relational OLAP
- relationale Speicherung des multidimensionalen Datenmodells
- Speicherung der aggregierten Daten in Relationen
- geringere Performanz
- „unbegrenzt“ skalierbar

MOLAP

- MOLAP = Multidimensional OLAP
- multidimensionale Speicherung des multidimensionalen Datenmodells
- Speicherung der aggregierten Daten in speziellen multidimensionalen Datenstrukturen
- hohe Performanz
- begrenzte Skalierbarkeit

Multidimensionale Speicherung

- Würfel („Hypercube“)
- Dimensionen:
  - Produkt, Ort, Zeit



Multidimensionale Speicherung

- Probleme
  - Reihenfolge der Dimensionen spielt eine Rolle
  - Nachbarschaftsbeziehung geht verloren
  - bei dünn besetztem Würfel hoher Speicherbedarf
    - dicht besetzt (dense) bei hohem Füllgrad
    - dünn besetzt (sparse) bei niedrigem Füllgrad
    - Füllgrad = Anzahl der nicht leeren Zellen / Anzahl aller Zellen
    - Eine Zelle mit dem Wert „0“ ist nicht leer!

ROLAP vs. MOLAP

	ROLAP	MOLAP
	relationales OLAP	multidimensionales OLAP
Speicherung	relationale Speicherstruktur (Tabellen, Tupel)	spezielle multidimensionale Speicherstruktur (Arrays)
Speicherzugriff	über Schlüssel, über Indizes, …	über Berechnungs-vorschrift
Performanz	mittel	hoch
Skalierbarkeit	„unbegrenzt“	begrenzt

HOLAP

- Hybrides OLAP
- Kombination von ROLAP und MOLAP
- Problem
  - MOLAP gut für dicht besetzte Würfel (leere Zellen belegen unnötig Speicher)
  - ROLAP gut für dünn besetzte Würfel (jedes Tupel speichert Schlüsselattribute)

  → Zwei-Ebenen-Datenstruktur

Zwei-Ebenen-Datenstruktur

alfatraining

- Unterscheidung in dünn- (sparse) und dichtbesetzte (dense) Dimensionen
- speichere Relation für dünn besetzte Dimensionen mit Zeiger auf einen Block
- Block ist Würfel aus dicht besetzten Dimensionen
- falls alle Dimensionen dünn besetzt sind und relational gespeichert werden:
  - → Block besteht aus einer Zelle
  - → äquivalent zu ROLAP
- falls alle Dimensionen dicht besetzt sind und in einem Block gespeichert werden:
  - → Relation ist leer bzw. enthält Zeiger auf genau einen Block
  - → äquivalent zu MOLAP

Weitere Elemente

- Es existieren weitere Elemente des physische Datenbankentwurfs
  - Indexstrukturen
  - Materialisierung
  - Partitionierung

alfatraining

Data Warehousing

Data-Warehouse-Entwurf
- Indexstrukturen -

Indexstrukturen

- Wie werden die gewünschten Tupel einer Tabelle gefunden?
  - lineare Suche?
  - → linearer Aufwand
  - Binärsuche?
  - → logarithmischer Such-Aufwand, aber teure Sortierung nötig
  - geht‘s besser?
  - → durch Einsatz von Indexstrukturen(=Indizes)

Indexstrukturen

- Aufgabe einer Indexstruktur
  - effizienter, wertbasierter Zugriff auf
    - einzelne Tupel
      - EMQ=Exact Match Query (assoziativer/wertbasierter Zugriff)
    - mehrere Tupel über Wertebereichen
      - RQ = Range Query (Assoziativer Zugriff und sequentielles Lesen)
      - Beispiel: WHERE Jahrgang = 1998

Indexstrukturen

- Ziel:
  - Minimierung des zu übertragenden Datenvolumens (Seiten)
    - Festplattenzugriff hat den größten Einfluss auf Antwortzeit
    - auf Festplatte erfolgt Zugriff in Blöcken
    - Indexstrukturen arbeiten auf Seiten (4kB – 8kB) (hier kann 1:1-Zuordnung zu Blöcken angenommen werden)
  - Minimierung der wahlfreien Zugriffe
    - wahlfreier Zugriff
      Zugriff auf Blöcke unabhängig von der Reihenfolge der Speicherung
    - sequentieller Zugriff
      Zugriff auf Blöcke in der Reihenfolge der Speicherung

Indexarten

- Primärindex
  Index über eindeutiges Attribut (ID=Schlüsselattribut) der Datenkollektion
- Sekundärindex
  Index über nicht eindeutiges Attribut (ID≠Schlüsselattribut) der Datenkollektion
- Clusterindex
  Datenkollektion ist physisch in der Reihenfolge der Indexwerte angeordnet

→ geclusterte Primärindizes

→ geclusterte Sekundärindizes

Indizes im Data Warehouse

alfatraining

- Welche Indexstrukturen unterstützen Data-Warehouse-Anfragen besonders gut?
- Problem
  - Filterung auf der Faktentabelle
- Anforderung
  - Effiziente Verknüpfung von Einschränkungen der Faktentabelle auf disjunkten Attributmengen

Multidimensionale Indizes

- Immer Kompromiss zwischen
  - Füllgrad
  - Überlappungsfreiheit
  - Balancierung des Baumes
  - Clusterung benachbarter Daten

alfatraining

Data Warehousing

Data-Warehouse-Entwurf
- Materialisierung -

Materialisierung

- Faktentabelle durchsuchen und Berechnung von Aggregationen kostet Zeit
- Idee
  - häufig gebrauchte aggregierte Zwischenergebnisse materialisieren (physisch ablegen, speichern)
- Systematisches Vorgehen
  - Dimensionshierarchien aufstellen
  - Häufigkeit der Zugriffe auf die einzelnen Aggregationsstufen protokollieren
  - Modell aufstellen, welche Ergebnisse von welchem Zwischenergebnis profitieren
  - kostenbasierte Entscheidung treffen (Speicherplatz ↔ Lesezugriff ↔ Schreibzugriff)

Materialisierung

- Kostenmodell
  - Speicherplatzkosten
  - Update-Kosten
    - also Kosten für Anpassung materialisierter Zwischenergebnisse bei Datenänderung
    - wobei Update auch von Materialisierung profitieren kann
- Nichttriviales Problem: Query Rewriting
  - also Umschreiben von Anfragen, so dass bestehende Materialisierungen genutzt werden können

Materialisierung: Beispiel

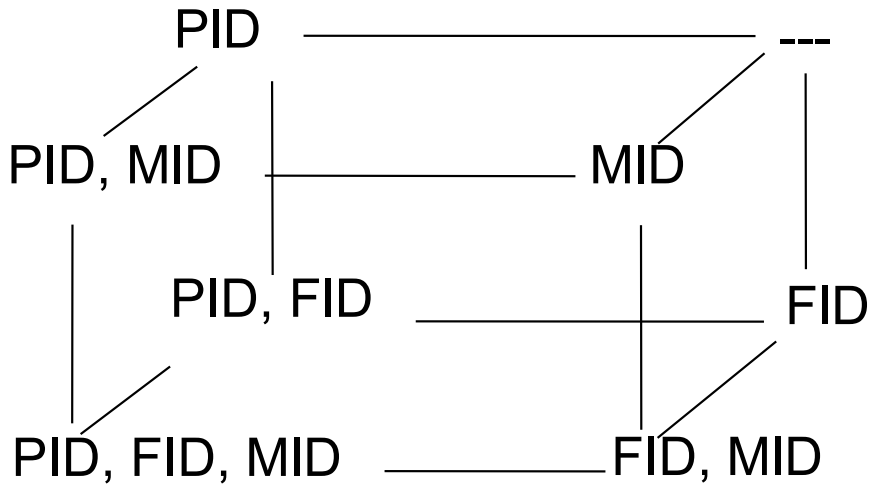
Monats-ID	Produkt-ID	Fahrzeug-ID	Einnahmen	Ausgaben
2004-Jan	5	1	1235	879
2004-Jan	7	3	5321	6345
2004-Feb	3	2	543	367
2004-Mar	4	1	235	198
2004-Apr	3	3	5432	5399
2004-Jan	2	2	745	4536
2004-Feb	2	3	346	636
2004-Jan	4	1	6246	3677
2004-Apr	1	2	326	436
2004-Apr	2	2	6436	7858
2004-May	4	3	8658	6356
2004-Jun	5	1	568	456
2004-May	4	3	5868	3167
2004-Jun	5	2	8762	6788

Quartals-ID	Produkt-ID	Einnahmen
2004-I	2	1091
2004-I	3	543
2004-I	4	6481
2004-I	5	1235
2004-I	7	5321
2004-II	1	326
2004-II	2	6436
2004-II	3	5432
2004-II	4	14526
2004-II	5	9330

- Wie sieht eine Anfrage nach den Einnahmen bei Lebensmittel-Transporten (Produkt-ID=3) pro Quartal mit und ohne Einsatz der aggregierten Tabelle aus?
- Star- oder Snowflake-Schema?

Materialisierung: Beispiel

- Aggregationsdimensionen:
  - Produkt (PID), Fahrzeug (FID), Monat (MID)



Materialisierung

- Erweiterung um mehrstufige Aggregationshierarchien (wie z.B. Monat ☐ Quartal im Beispiel) und gegebenenfalls Dimensionseinschränkungen führen zu explosionsartigem Wachstum der Möglichkeiten!

alfatraining

Data Warehousing

Data-Warehouse-Entwurf
- Data-Warehouse-Werkzeuge -

Data-Warehouse-Werkzeuge

alfatraining

- ETL-Werkzeuge
  - Werkzeuge zur **Unterstützung des Aufbaus und der Wartung**
  - eines Data Warehouse
  - Beispiele:
    - DataJunction
    - DataStage (Ascential)
    - …
- OLAP-Werkzeuge
  - Werkzeuge zur **multidimensionalen Datenanalyse**
  - Beispiele:
    - Hyperion Essbase
    - BusinessObjects
    - Cognos
    - MicroStrategy
    - …

Vergleich DW-Werkzeuge

Funktionsumfang der Business-Intelligence-Werkzeuge

	Daten-integration	Datenbank-management 1)	Online-analyse 2)	Webapplikations-aufbau
Arcplan				●
Ascential	●			
Brio				●
Business Objects	+			●
Cognos	●		●	●
Crystal Decisions			●	●
Hyperion	◗		●	+
IBM	◗	●		
Informatica	●			+
Microsoft	◗	●	●	
Microstrategy			●	
MIK			●	
MIS			●	◗
NCR Teradata		●		
Oracle	◗	●	●	
SAP		◗	●	+
SAS	◗	●	●	◗

1) Relationales Datenbank-Management-System (RDBMS)
2) Online Analytical Processing (OLAP), Datenbank oder explizites Werkzeug

Anfragebearbeitung: SQL

- Beispiel SQL

```
SELECT <Felder>
FROM <Tabellen>
WHERE <Einschränkungen>
GROUP BY <Gruppierungsfelder>
HAVING <Gruppeneinschränkungen>
```

- Oder

```
UPDATE <Tabelle>
SET <Feld> = <Wert>
WHERE <Einschränkungen>
```

Anfragebearbeitung: SQL

- Zum Aufwärmen:
  - SELECT Name, Vname, AVG(Note) AS Schnitt
  - FROM Klausuren, Schüler
  - FROM Klausuren INNER JOIN Schüler ON Schüler.ID = Klausuren.SID
  - WHERE Jahrgang = 1998
  - AND Schüler.ID = Klausuren.SID
  - GROUPBY Name, Vname
  - HAVING Schnitt > 4.0
- Was macht diese Anfrage?

… = kaProduct

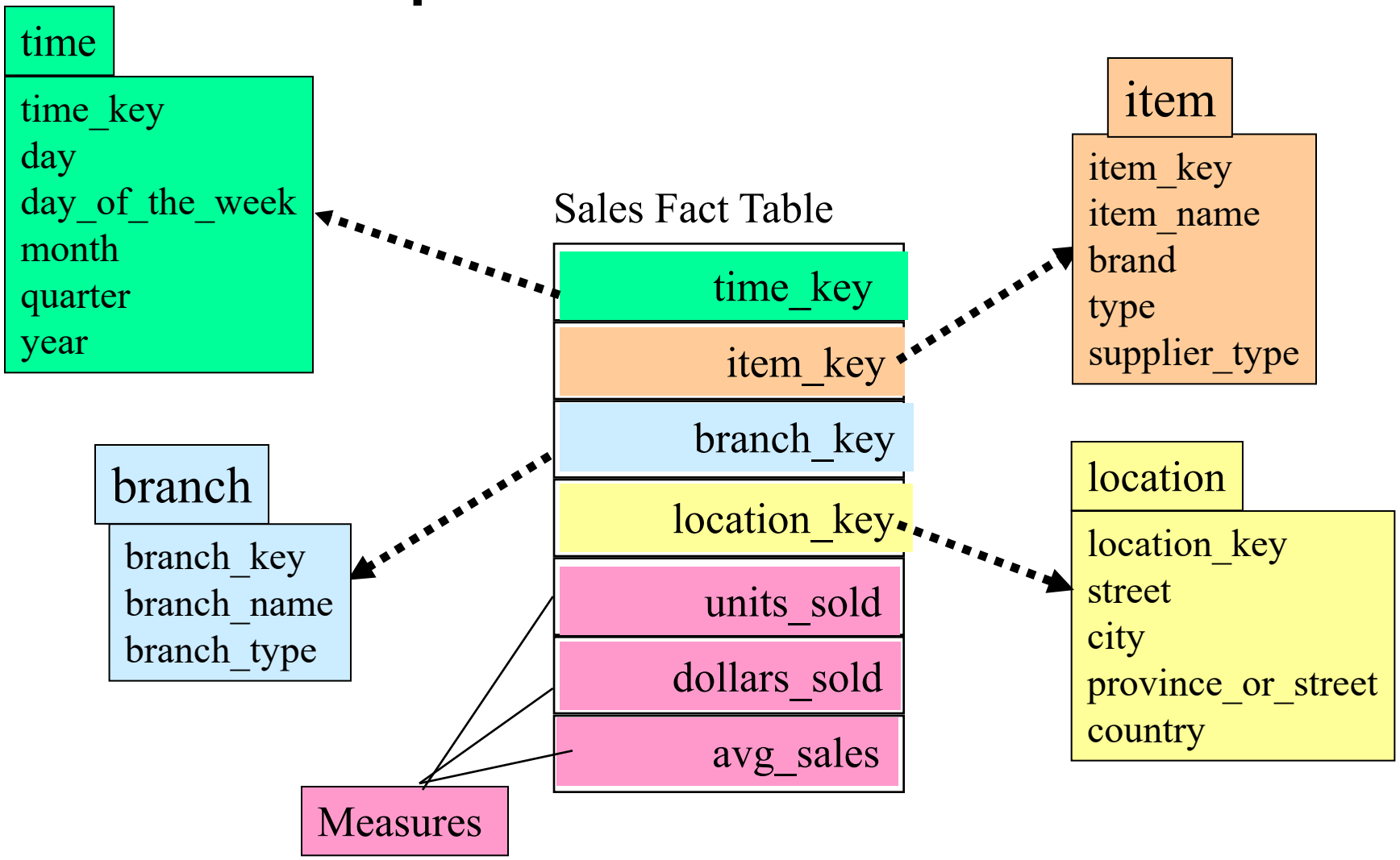
vs. Inner join

….

ANSI SQL

- https://plus-it.de/mdx-dax-sql-die-richtige-sprache-fur-jedes-datenmodell/

Example of Star Schema



Conceptual Modeling of Data Warehouses

Modeling data warehouses: dimensions & measures

- Star schema: A fact table in the middle connected to a set of dimension tables
- Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
- Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

Star

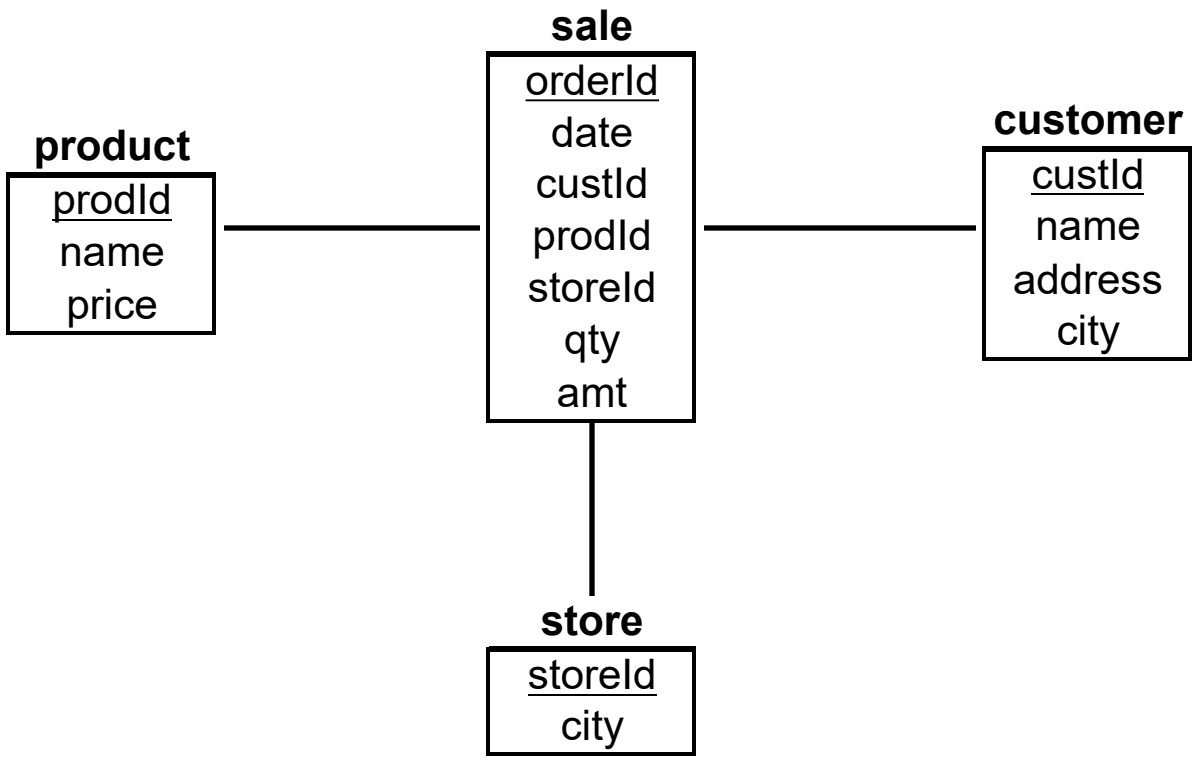
product	prodId	name	price
	p1	bolt	10
	p2	nut	5

store	storeId	city
	c1	nyc
	c2	sfo
	c3	la

sale	oderId	date	custId	prodId	storeId	qty	amt
	o100	1/7/97	53	p1	c1	1	12
	o102	2/7/97	53	p2	c1	2	11
	105	3/8/97	111	p1	c3	5	50

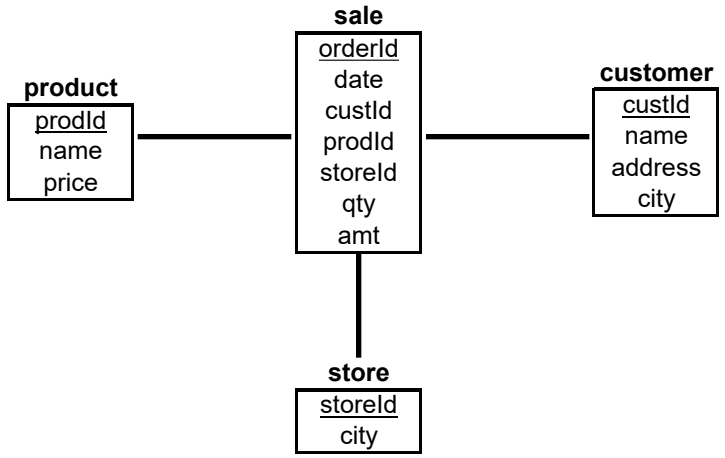
customer	custId	name	address	city
	53	joe	10 main	sfo
	81	fred	12 main	sfo
	111	sally	80 willow	la

Star Schema


sale
orderId
date
custId
prodId
storeId
qty
amt
product
prodId
name
price
customer
custId
name
address
city
store
storeId
city

Terms

- Fact table
- Dimension tables
- Measures



Cube

Fact table view:

sale	prodId	storeId	amt
	p1	c1	12
	p2	c1	11
	p1	c3	50
	p2	c2	8

Multi-dimensional cube:

	c1	c2	c3
p1	12		50
p2	11	8	

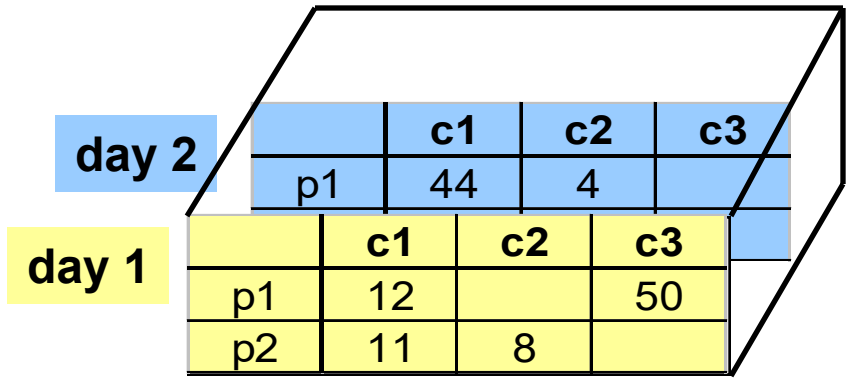
dimensions = 2

3-D Cube

Fact table view:

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

Multi-dimensional cube:

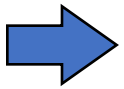


dimensions = 3

Aggregates

- Add up amounts for day 1
- In SQL: SELECT sum(amt) FROM SALE
  WHERE date = 1

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

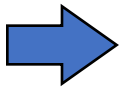


81

Aggregates

- Add up amounts by day
- In SQL: SELECT date, sum(amt) FROM SALE
  GROUP BY date

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

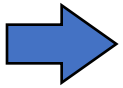


ans	date	sum
	1	81
	2	48

Another Example

- Add up amounts by day, product
- In SQL: SELECT date, sum(amt) FROM SALE
  GROUP BY date, prodId

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4



sale	prodId	date	amt
	p1	1	62
	p2	1	19
	p1	2	48

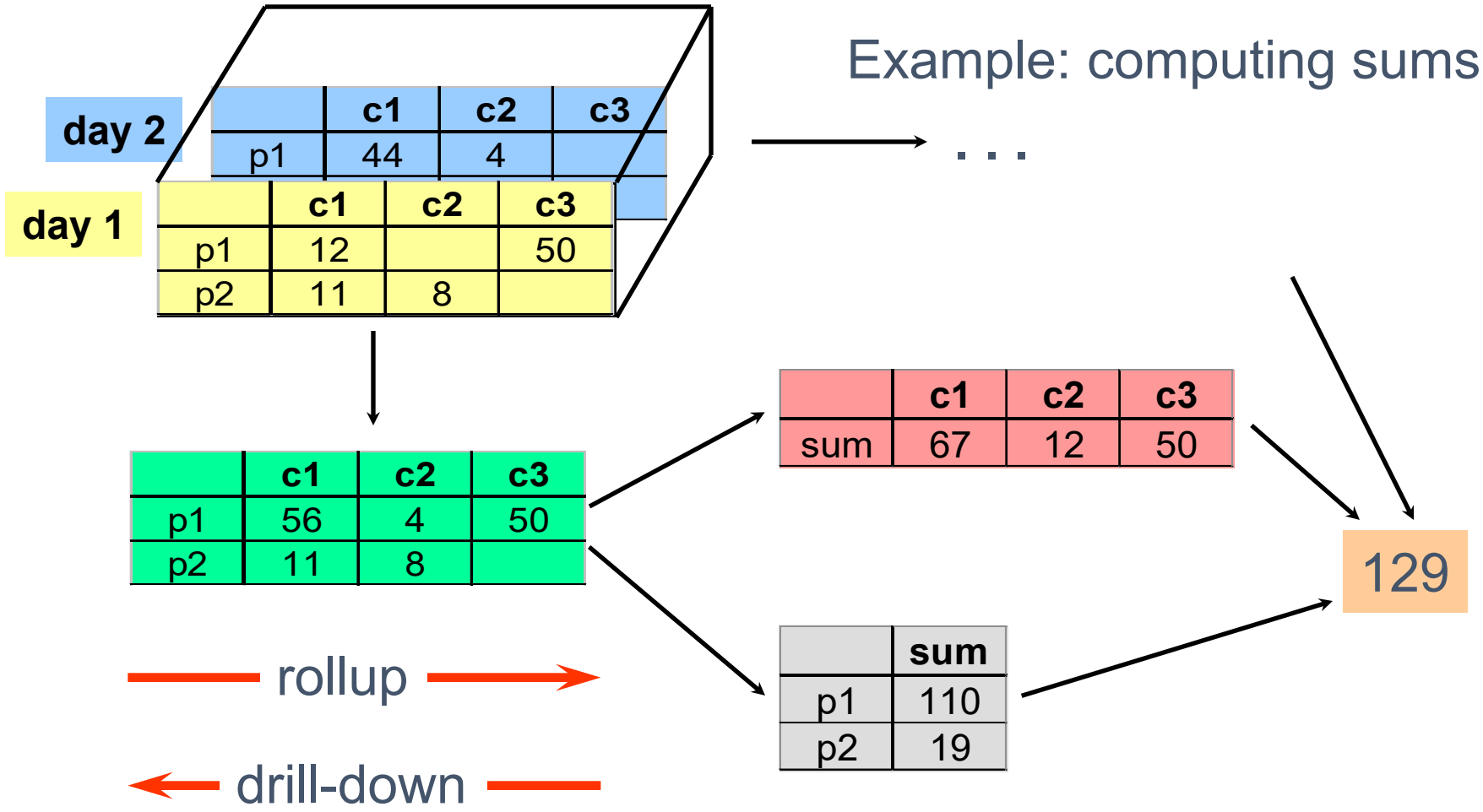
rollup →

← drill-down

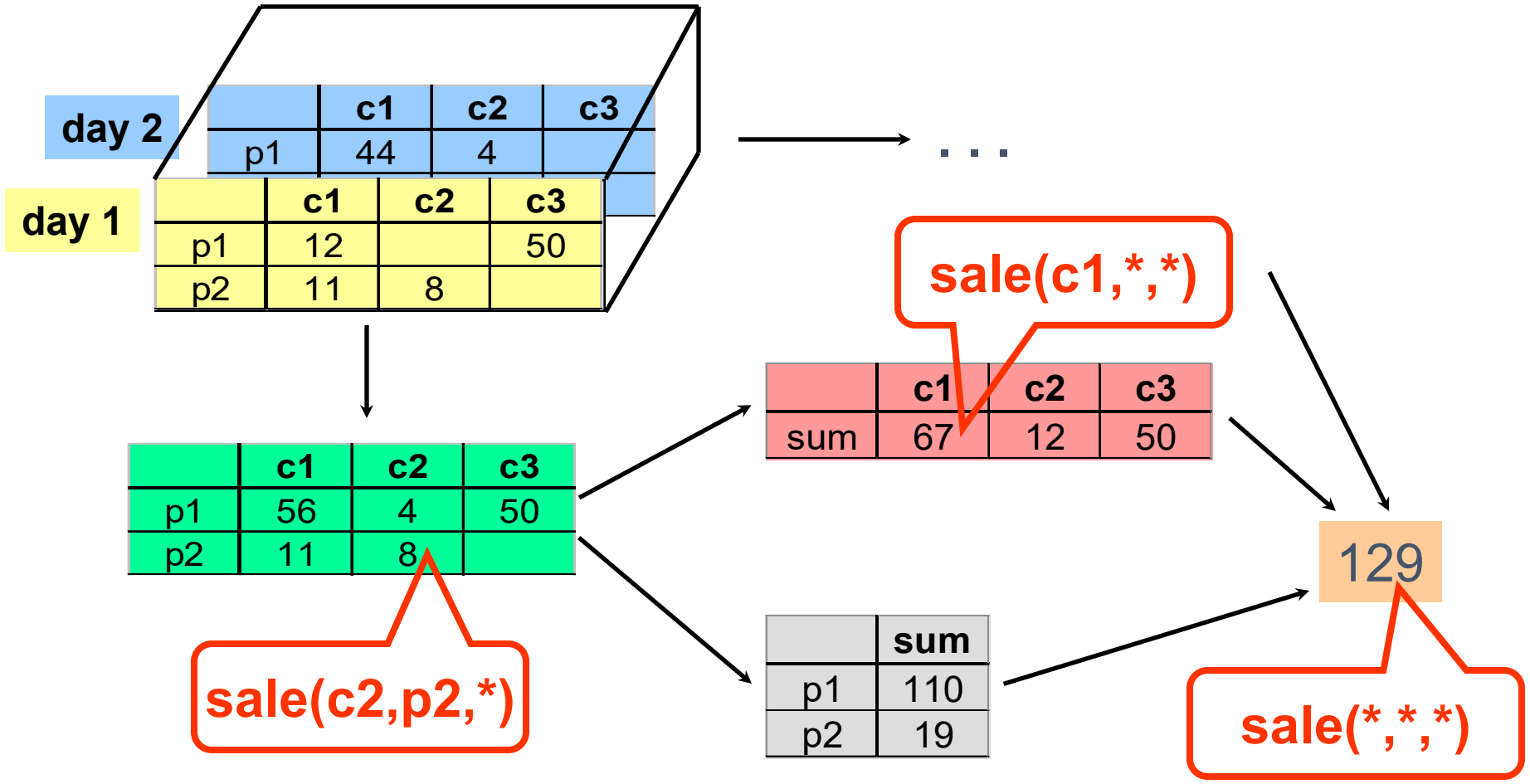
Aggregates

- Operators: sum, count, max, min, median, ave
- “Having” clause
- Using dimension hierarchy
  - average by region (within store)
  - maximum by month (within date)

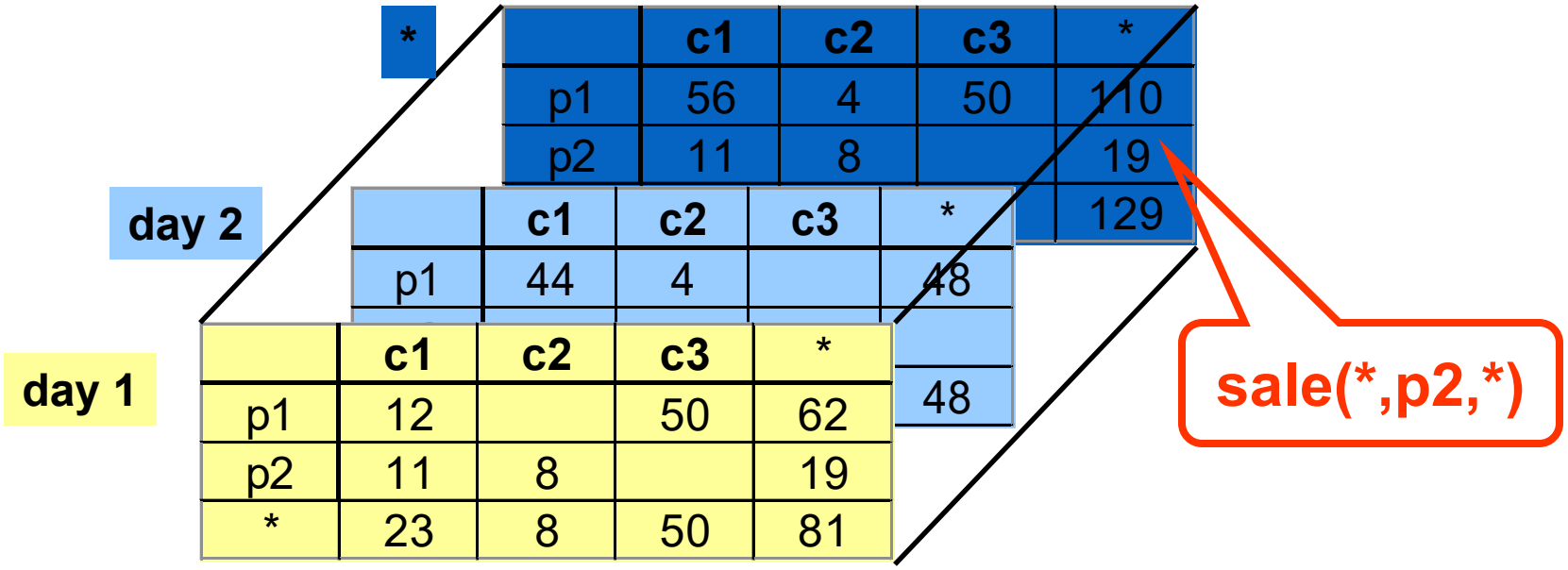
Cube Aggregation



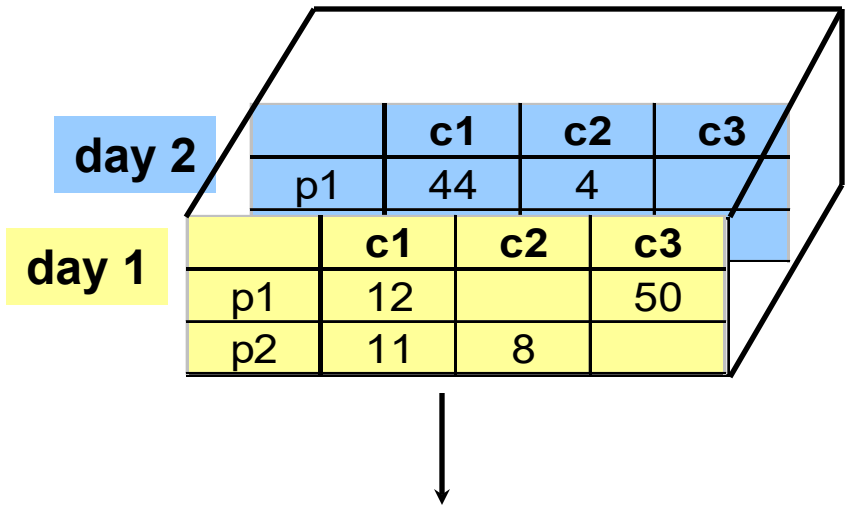
Cube Operators



Extended Cube



Aggregation Using Hierarchies



day 2	c1	c2	c3
p1	44	4	

day 1	c1	c2	c3
p1	12		50
p2	11	8	

	region A	region B
p1	56	54
p2	11	8

customer
|
region
|
country

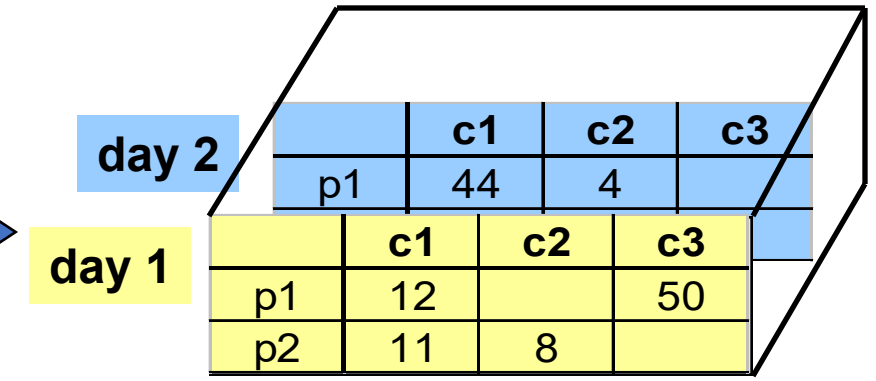
(customer c1 in Region A;
customers c2, c3 in Region B)

Pivoting

Fact table view:

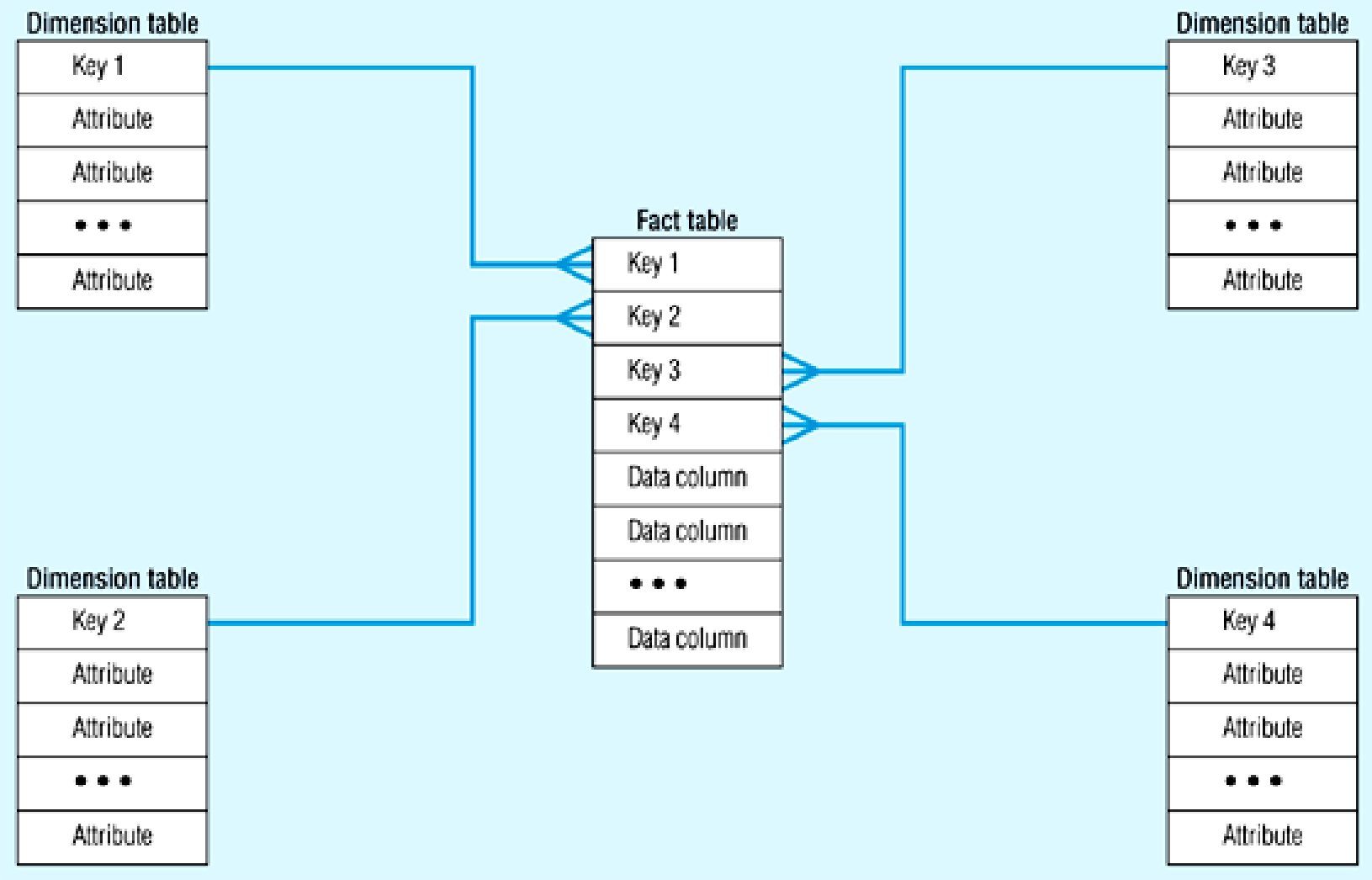
sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

Multi-dimensional cube:

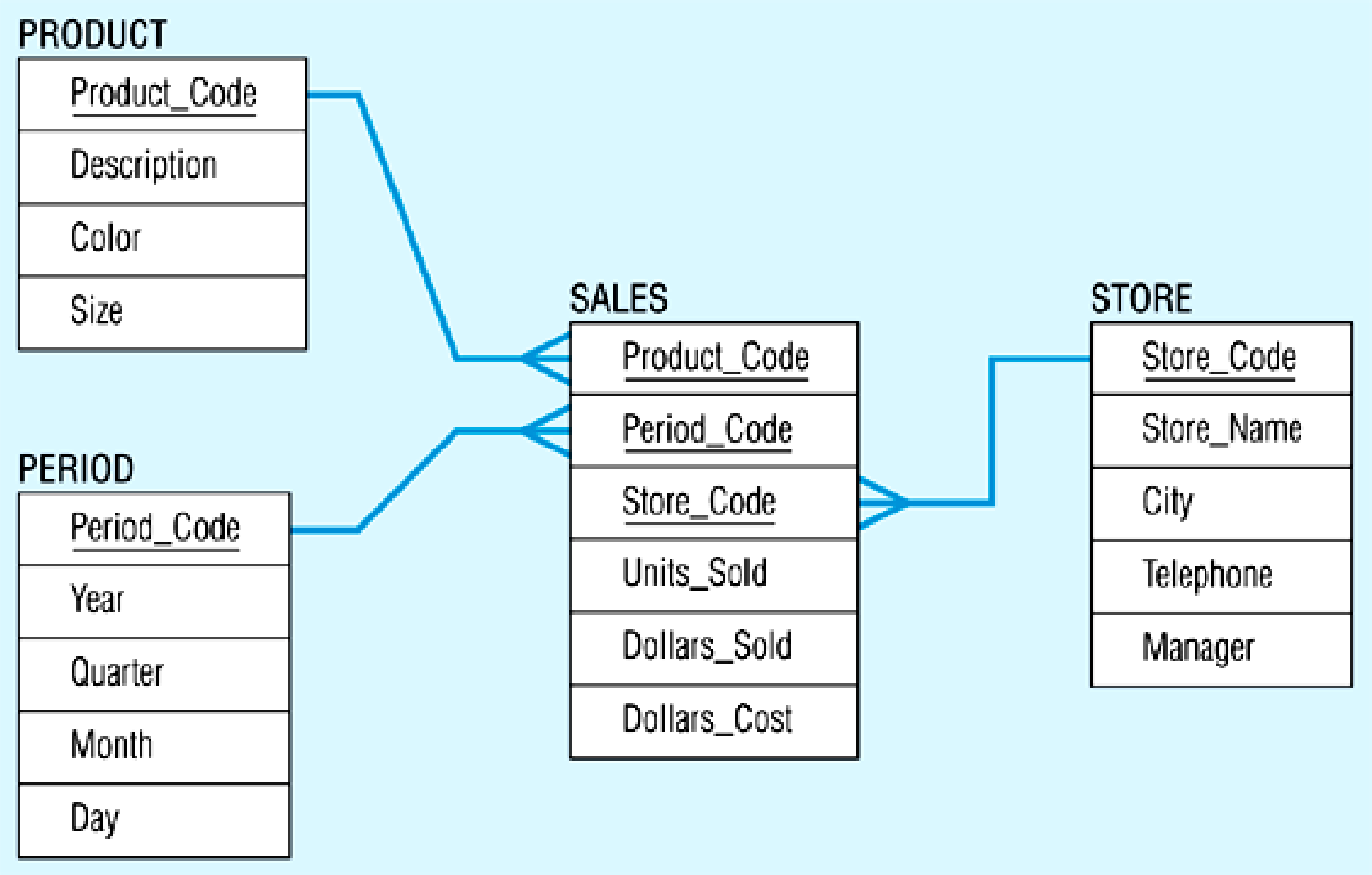


	c1	c2	c3
p1	56	4	50
p2	11	8	

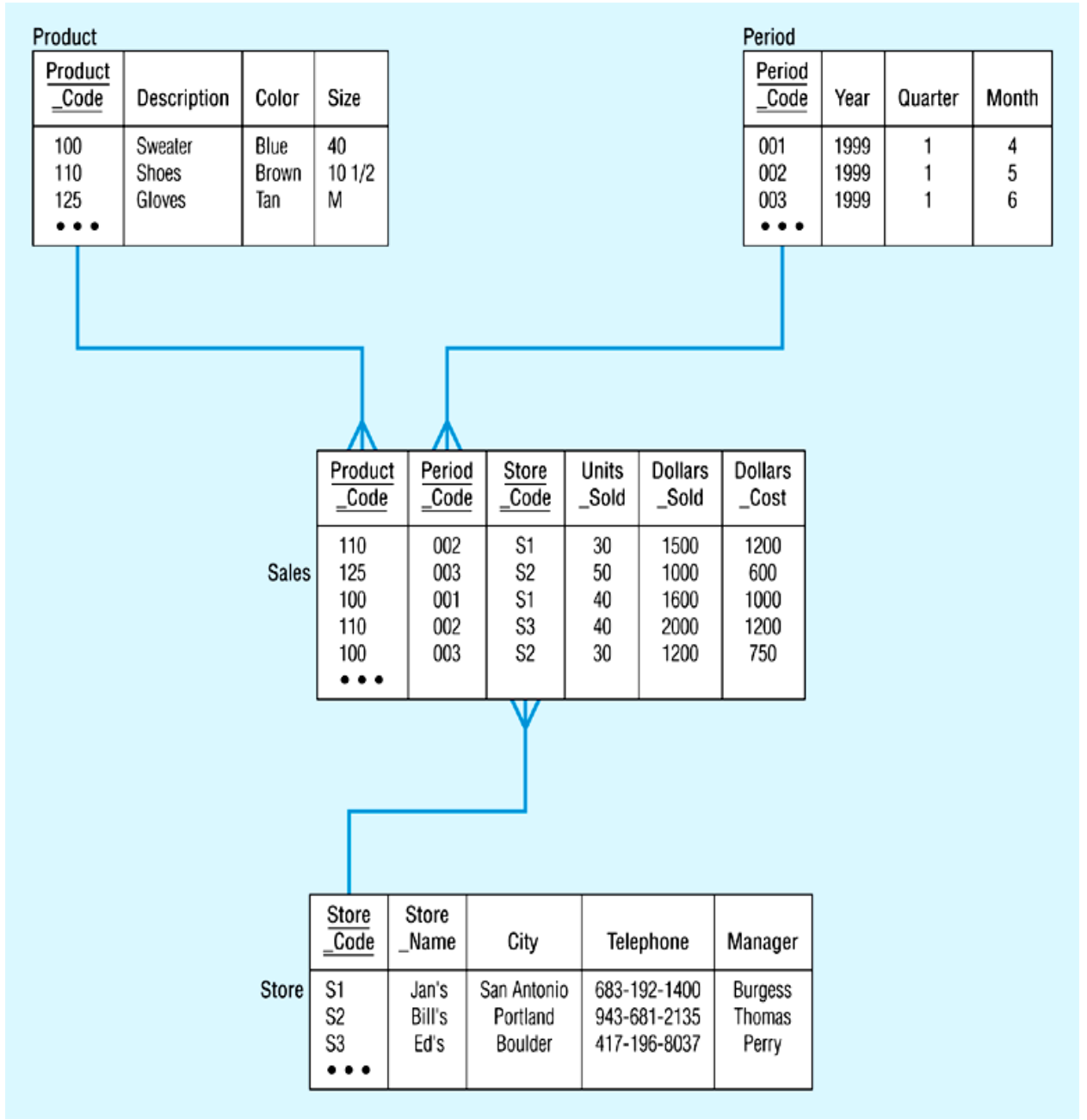
Star Schema (in RDBMS)



Star Schema Example



Star Schema with Sample Data



The “Classic” Star Schema

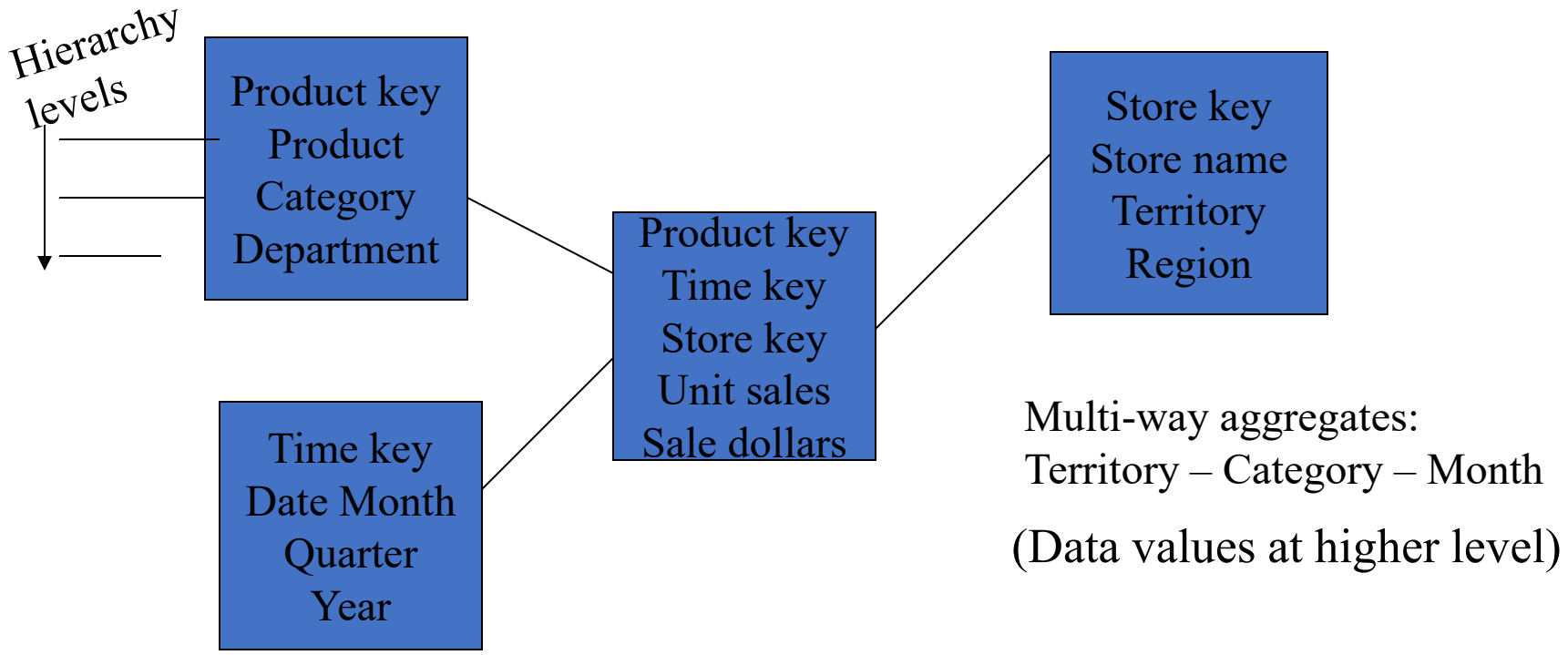
- A relational model with a one-to-many relationship between dimension table and fact table.
- A single fact table, with detail and summary data
- Fact table primary key has only one key column per dimension
- Each dimension is a single table, highly denormalized
- **Benefits**: Easy to understand, intuitive mapping between the business entities, easy to define hierarchies, reduces # of physical joins, low maintenance, very simple metadata
- **Drawbacks**: Summary data in the fact table yields poorer performance for summary levels, huge dimension tables a problem

Need for Aggregates

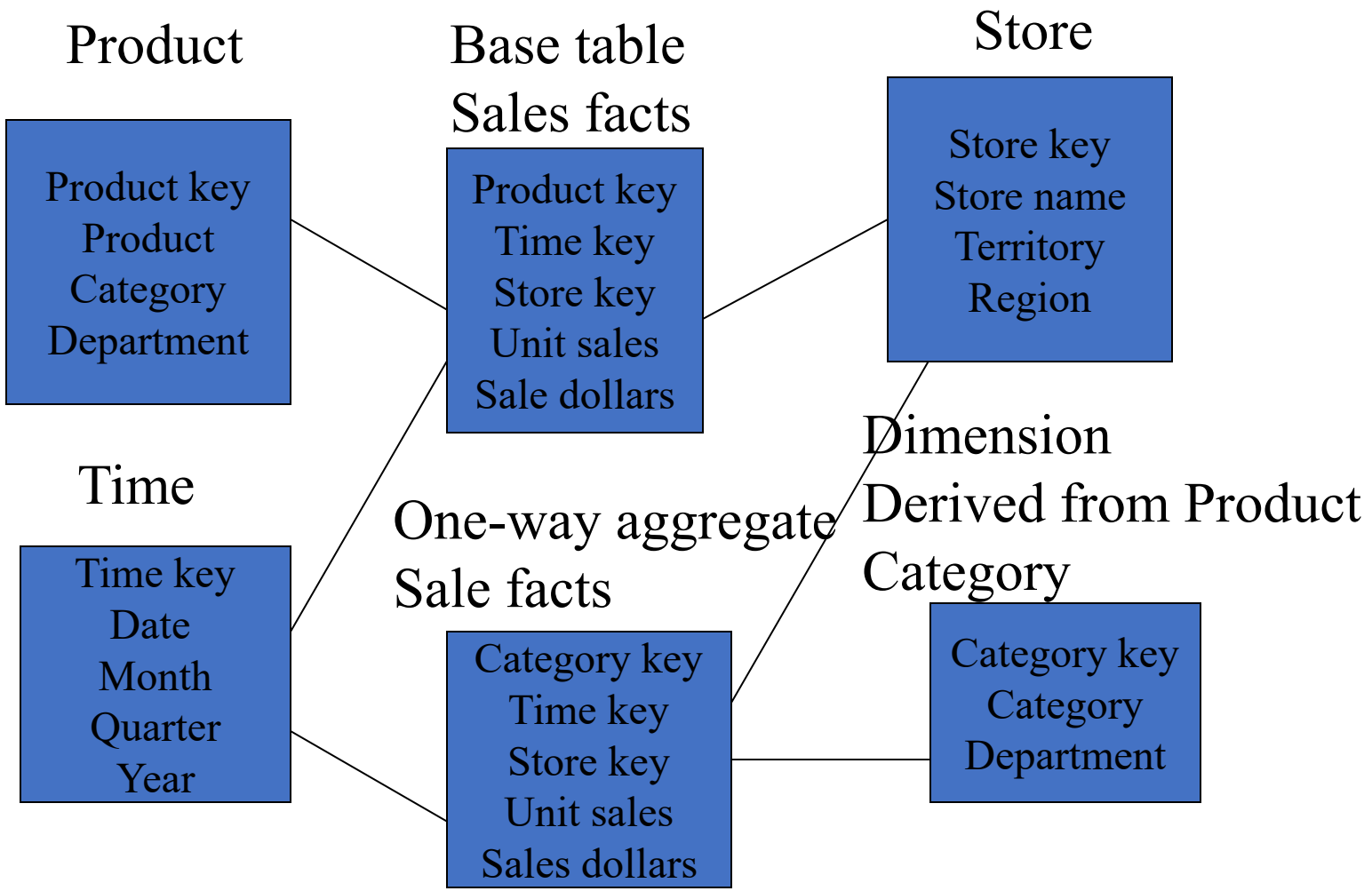
- Sizes of typical tables:
  - Time dimension: 5 years x 365 days = 1825
  - Store dimension: 300 stores reporting daily sales
  - Production dimension: 40,000 products in each store (about 4000 sell in each store daily)
  - Maximum number of base fact table records: 2 billion (lowest level of detail)
- A query involving 1 brand, all store, 1 year: retrieve/summarize over 7 million fact table rows.

Aggregating Fact Tables

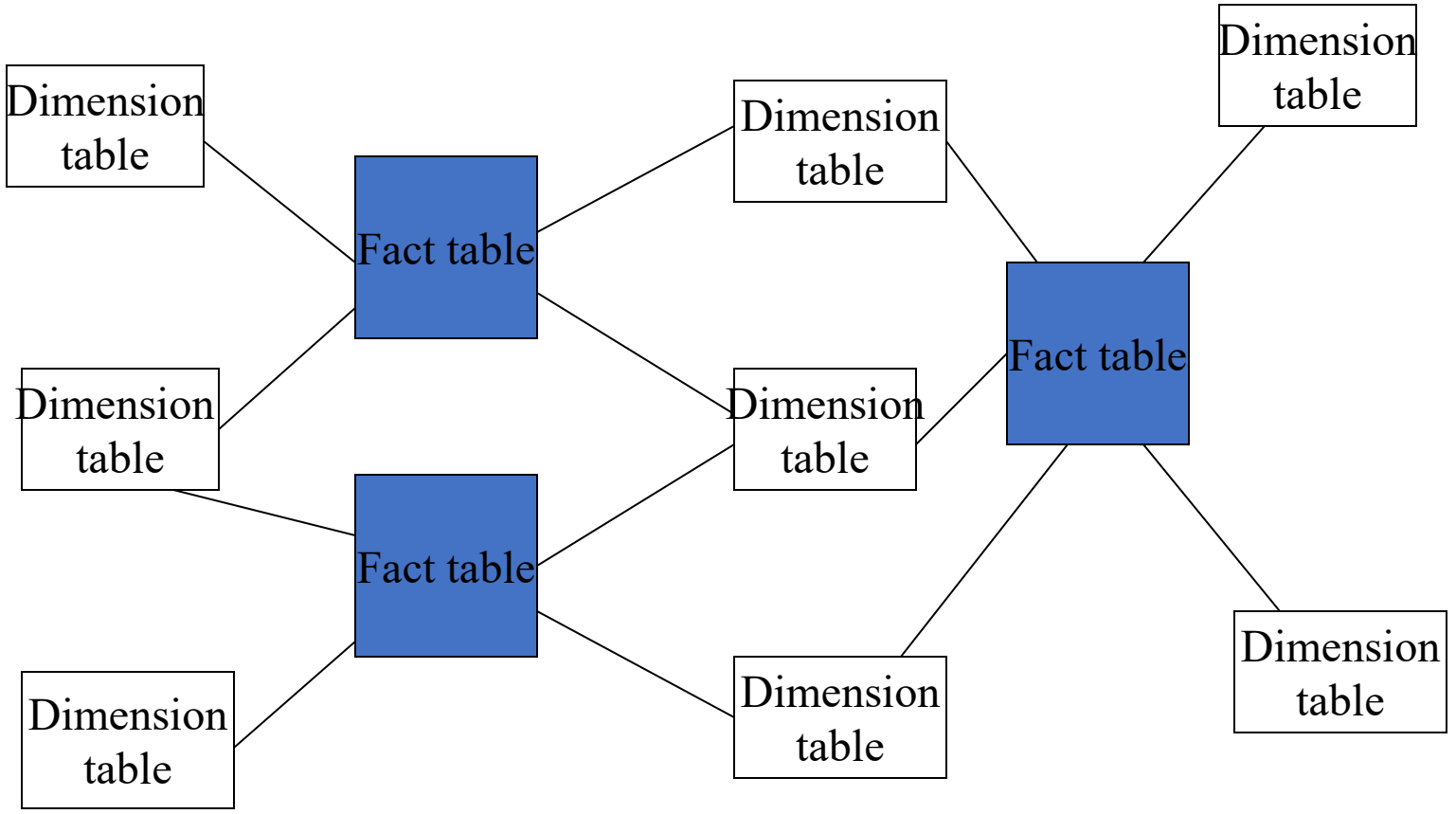
- Aggregate fact tables are summaries of the most granular data at higher levels along the dimension hierarchies.



Aggregate Fact Tables



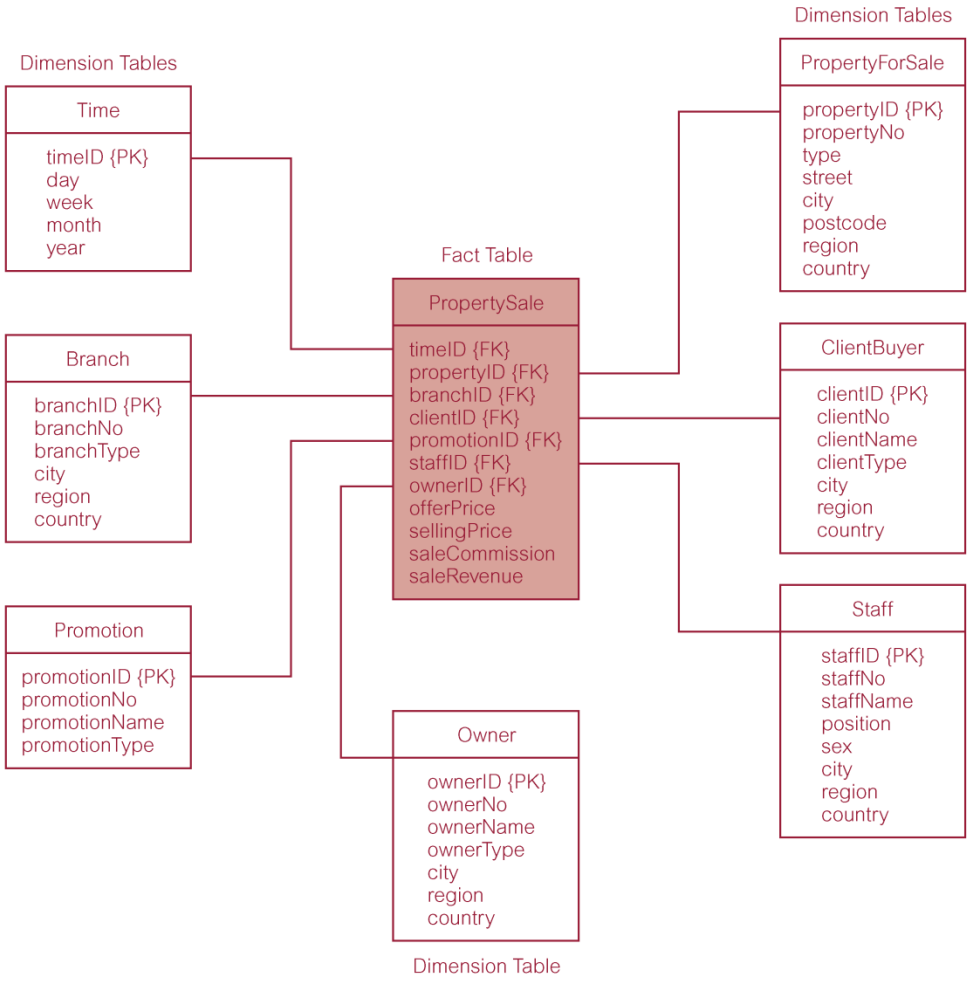
Families of Stars



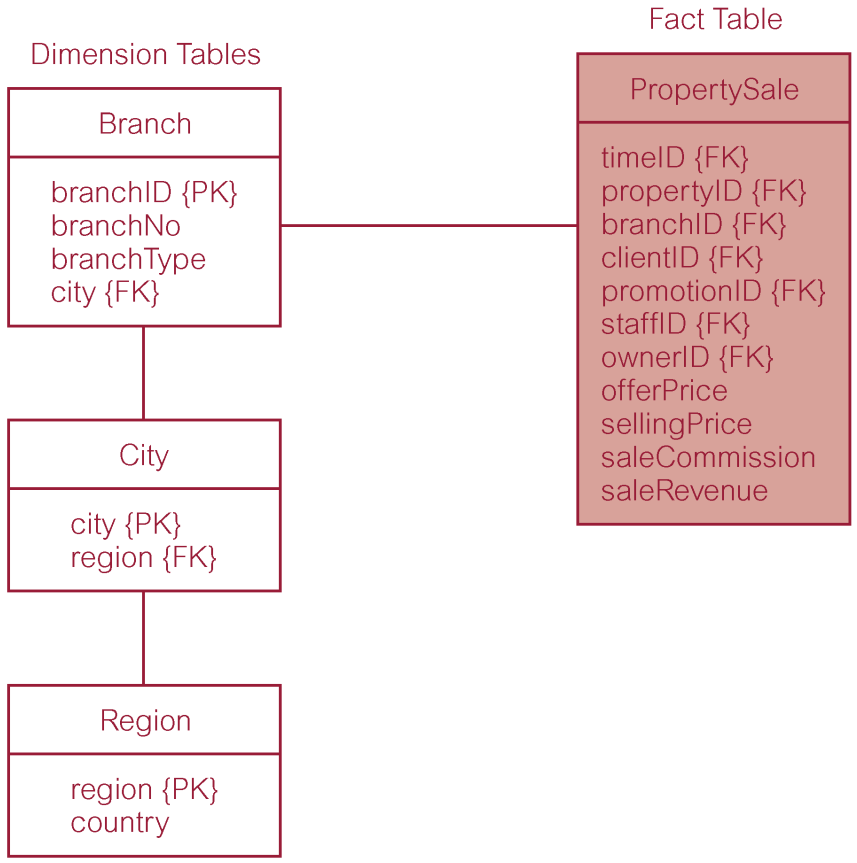
What is the Best Design?

- Performance benchmarking can be used to determine what is the best design.
- Snowflake schema: easier to maintain dimension tables when dimension tables are very large (reduce overall space). It is not generally recommended in a data warehouse environment.
- Star schema: more effective for data cube browsing (less joins): can affect performance.

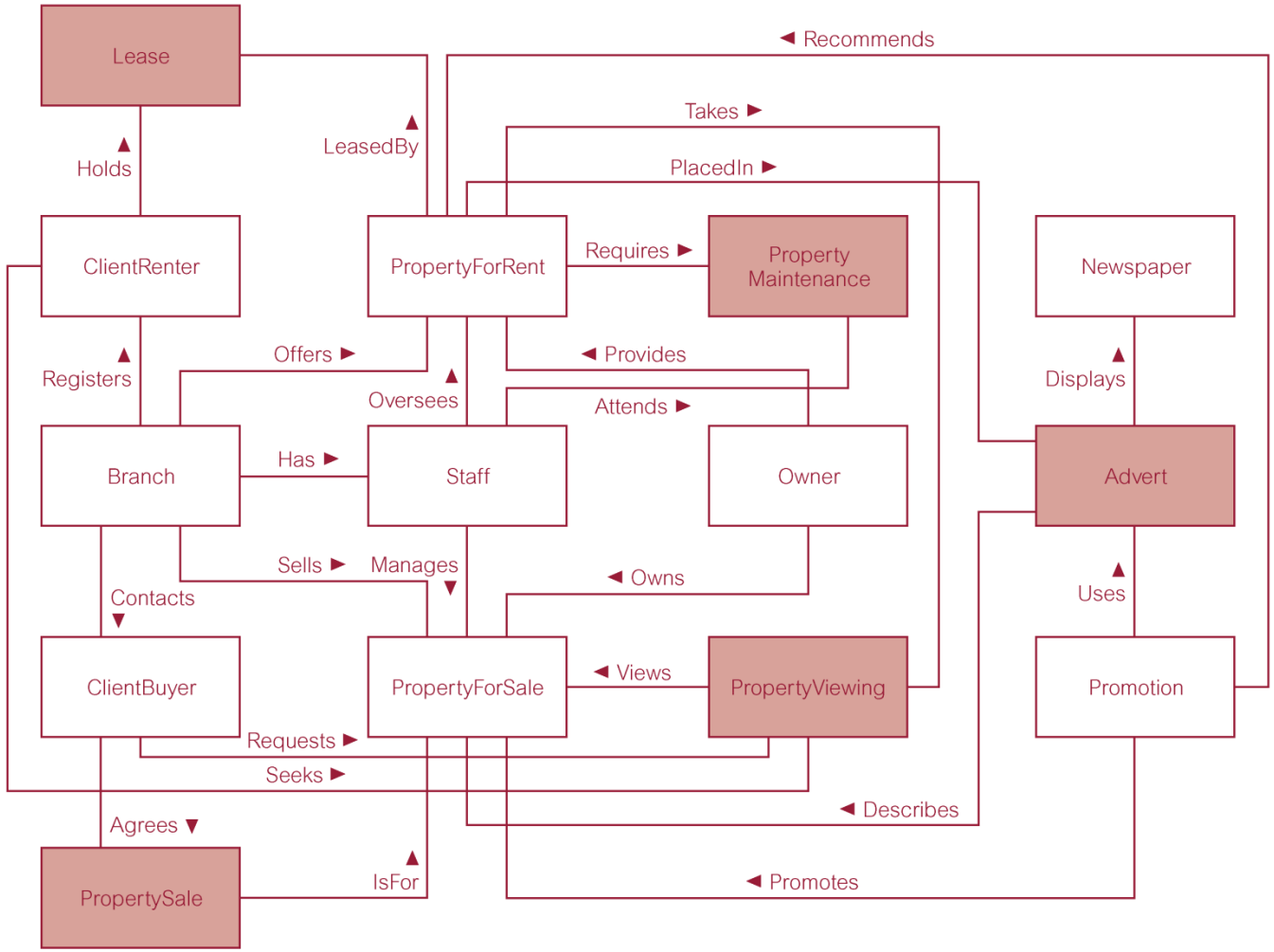
Star schema (dimensional model) for property sales of *DreamHome*



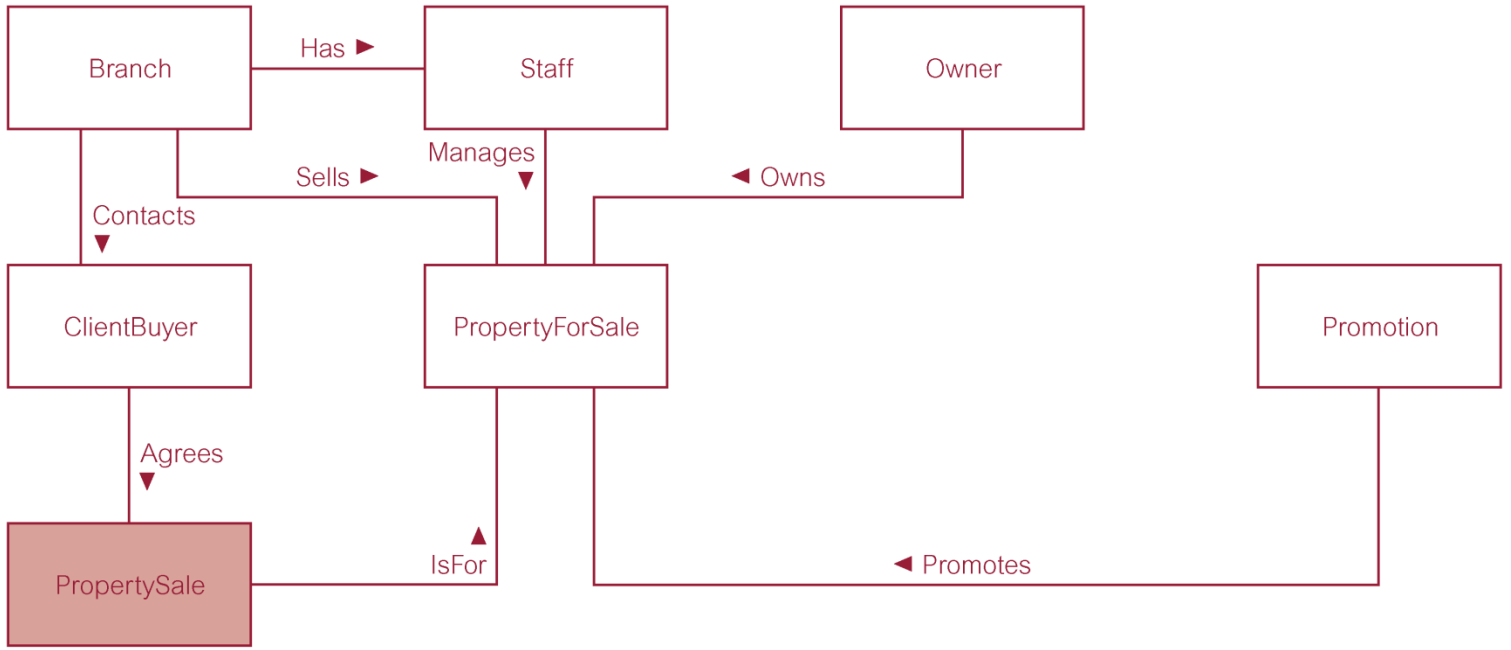
Property sales with normalized version of Branch dimension table



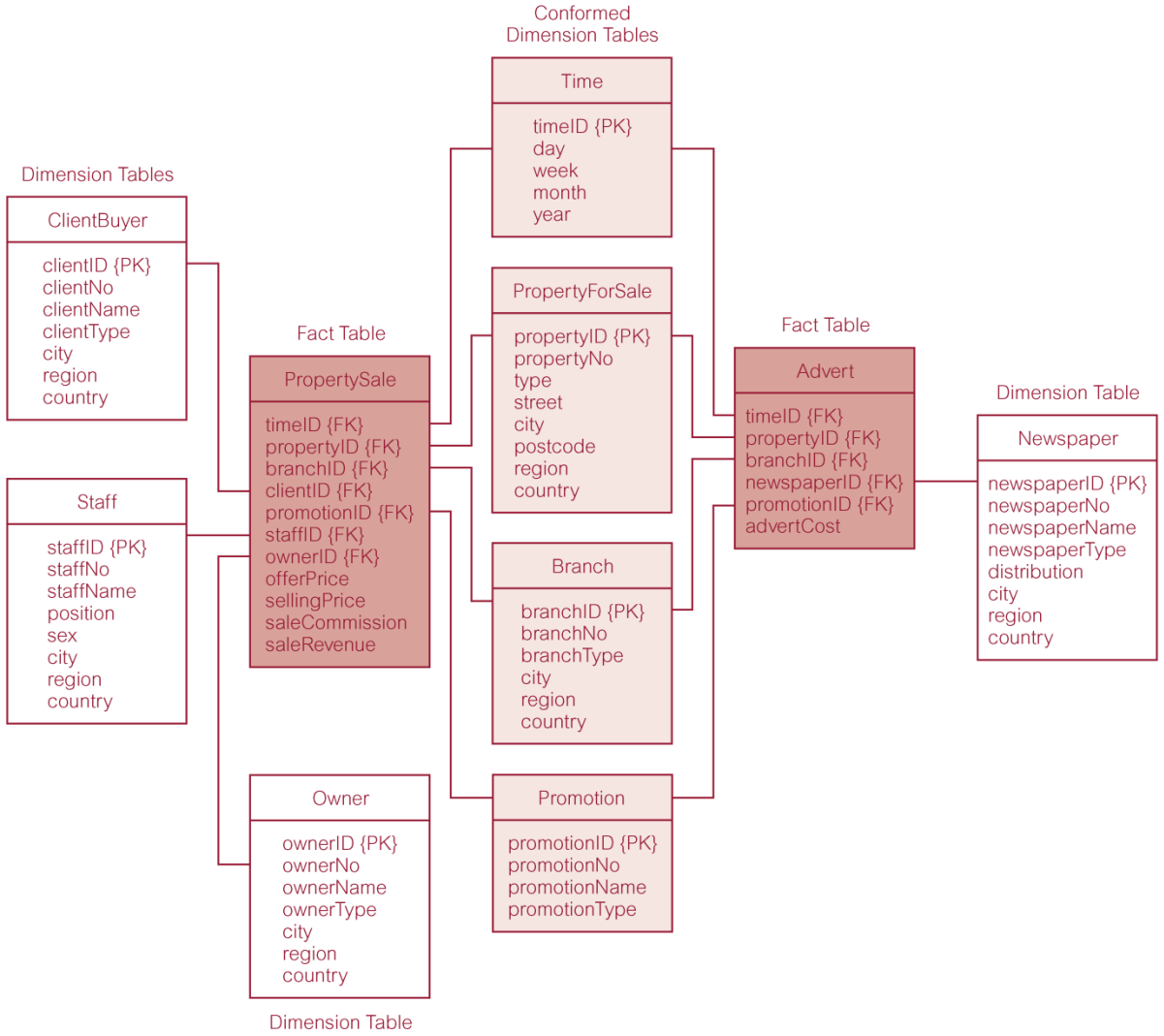
ER model of an extended version of *DreamHome*



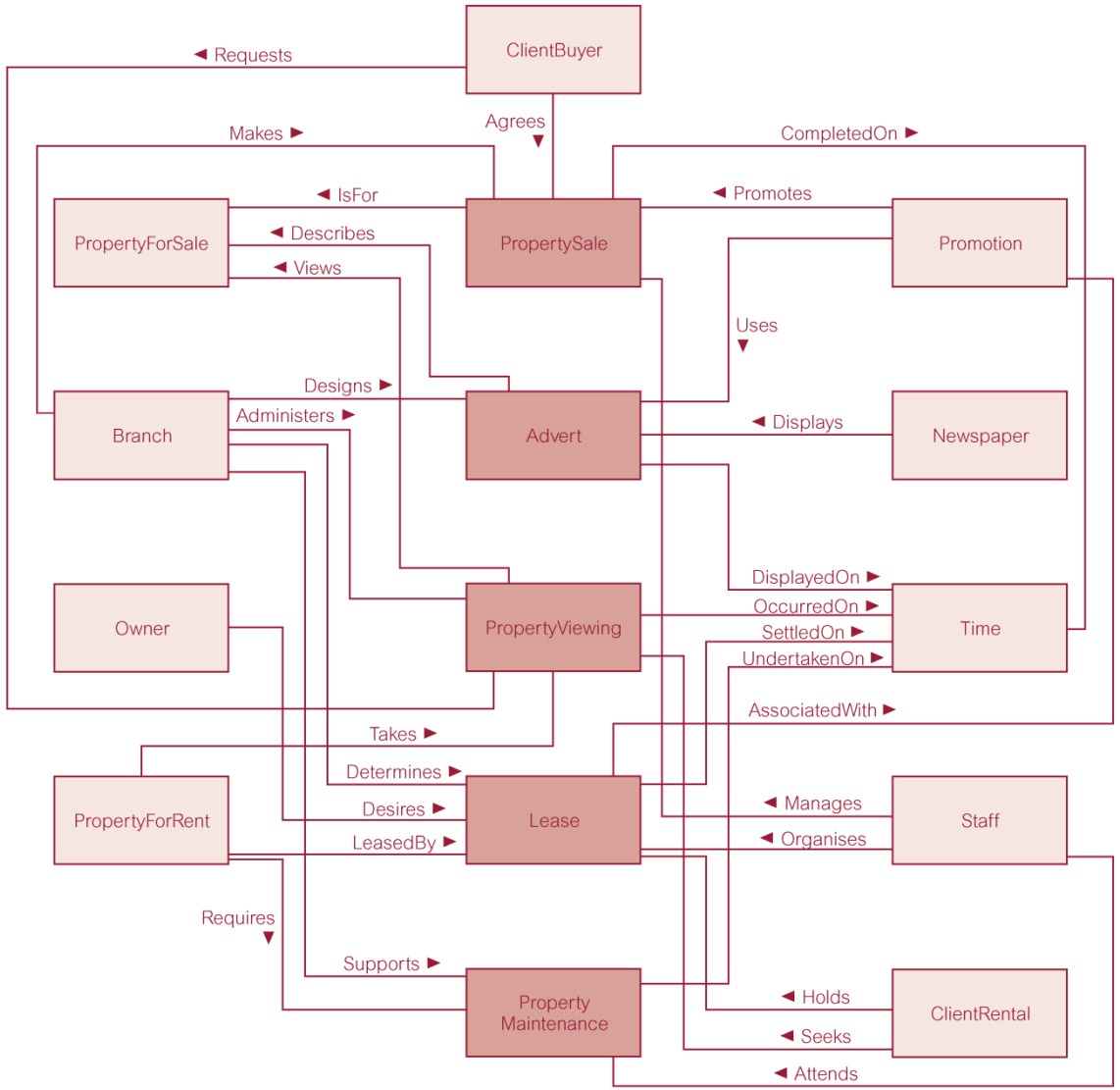
ER model of property sales business process of *DreamHome*



Star schemas for property sales and property advertising



Dimensional model (fact constellation) for the *DreamHome* data warehouse



Fragen?