

Betriebliche Anwendungen

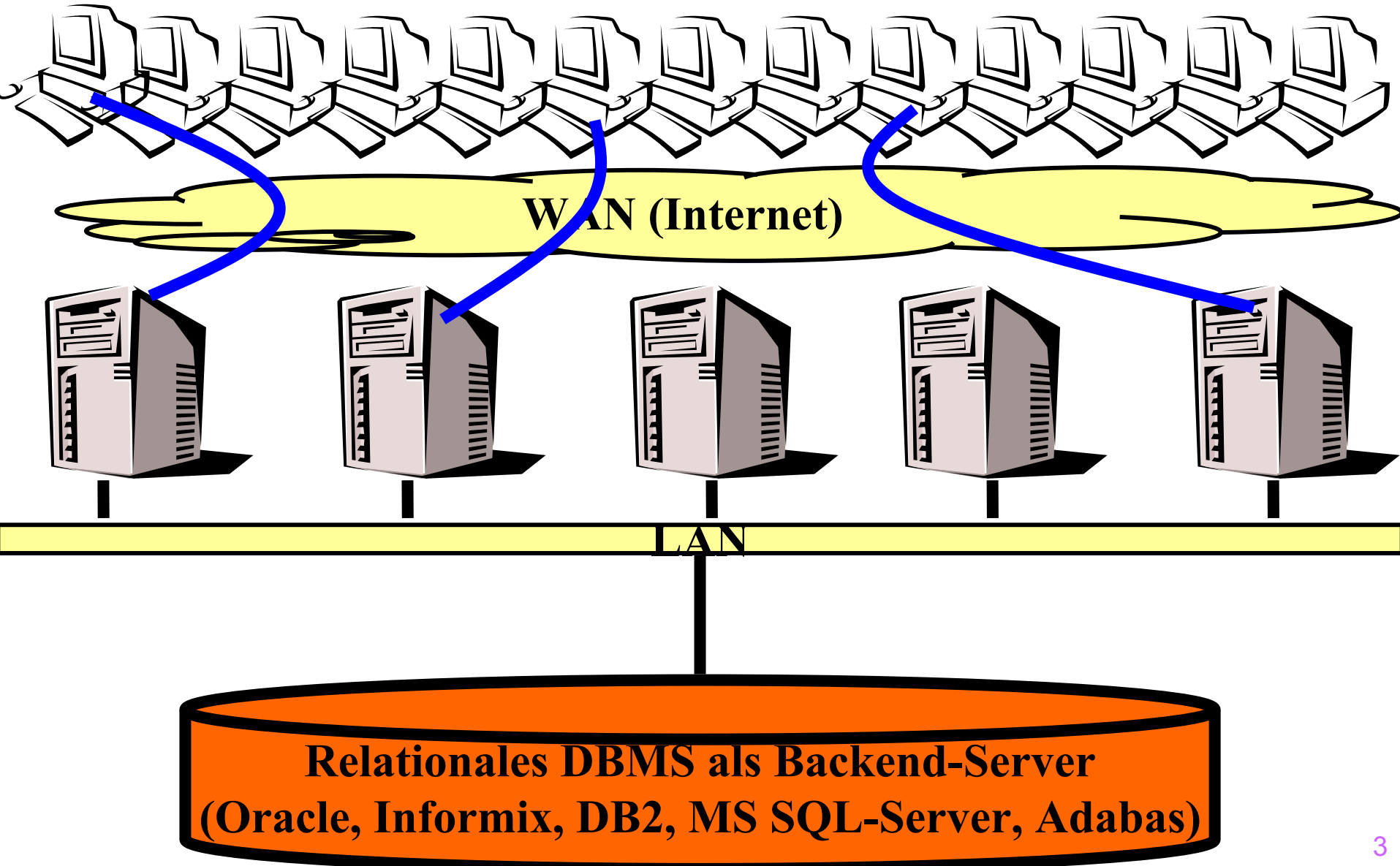
- OLTP
- Data Warehouse
- Data Mining



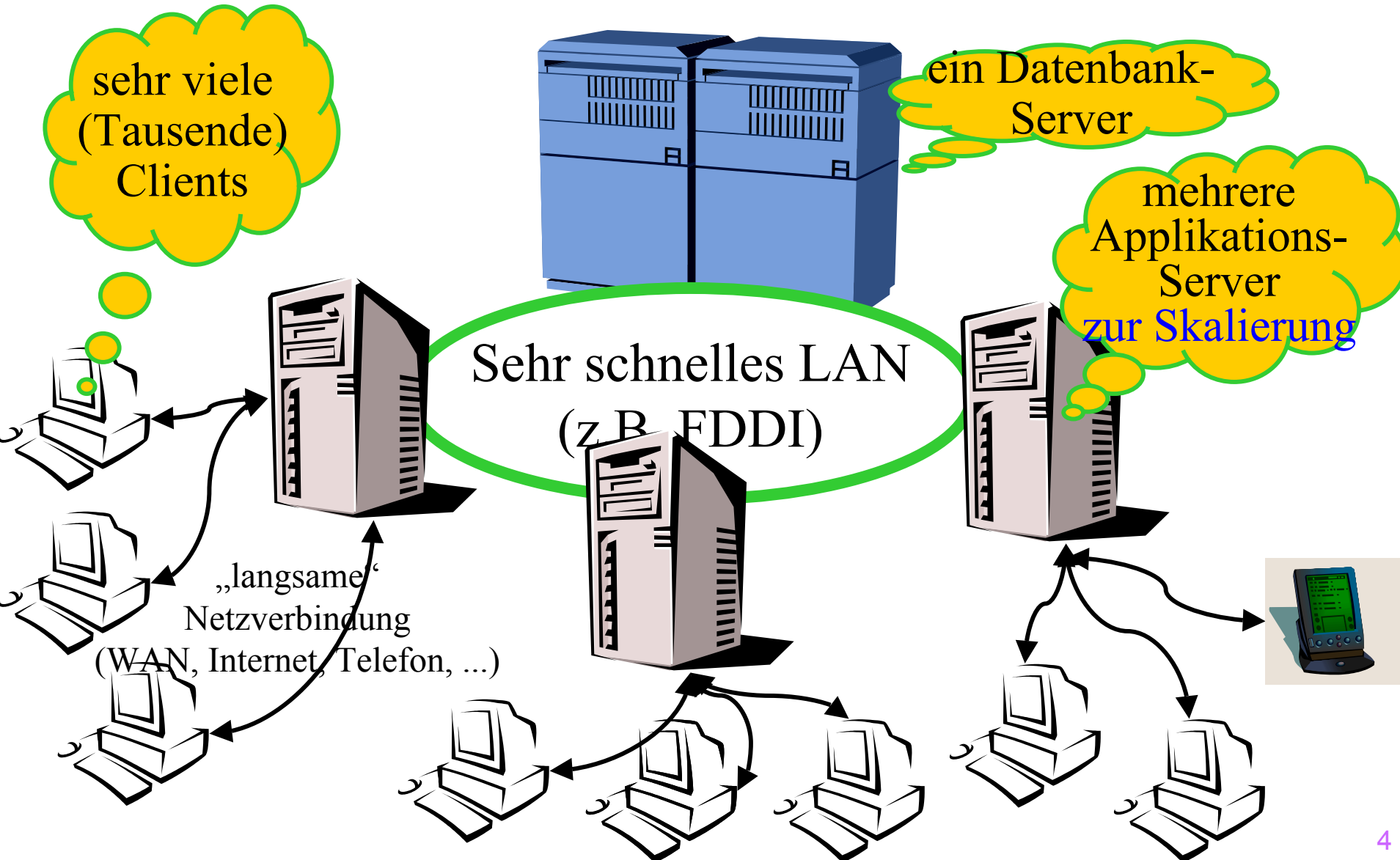
OLTP: Online Transaction Processing

- Beispiele
 - Flugbuchungssystem
 - Bestellungen in einem Handelsunternehmen
- Charakterisierung
 - Hoher Parallelitätsgrad
 - Viele (Tausende pro Sekunde) kurze Transaktionen
 - TAs bearbeiten nur ein kleines Datenvolumen
 - „mission-critical“ für das Unternehmen
 - Hohe Verfügbarkeit muss gewährleistet sein
- Normalisierte Relationen (möglichst wenig Update-Kosten)
- Nur wenige Indexe (wegen Fortschreibungskosten)

SAP R/3: Enterprise Resource Modelling (ERP-System)

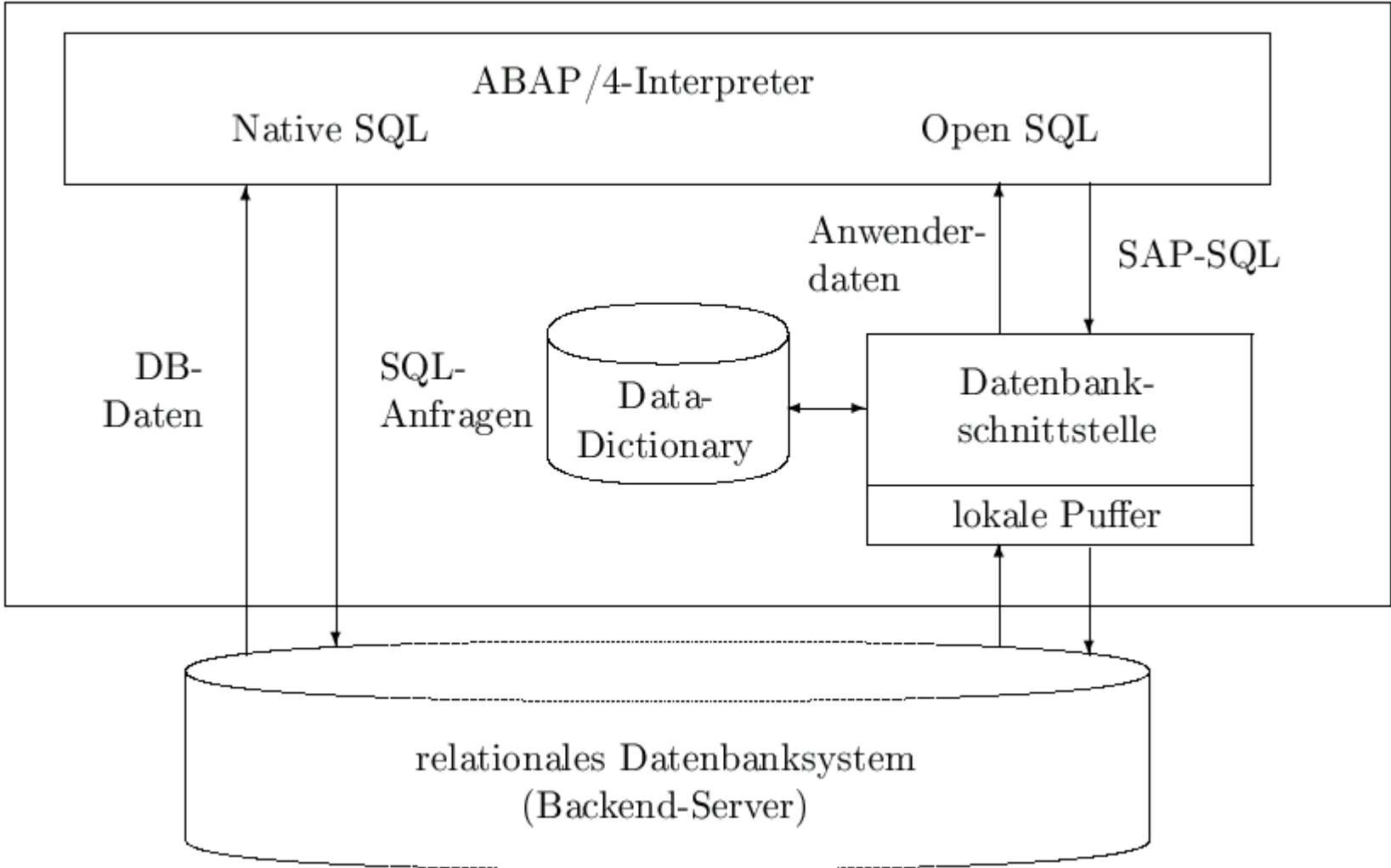


Dreistufige Client/Server-Architektur (3 Tier, SAP R/3)

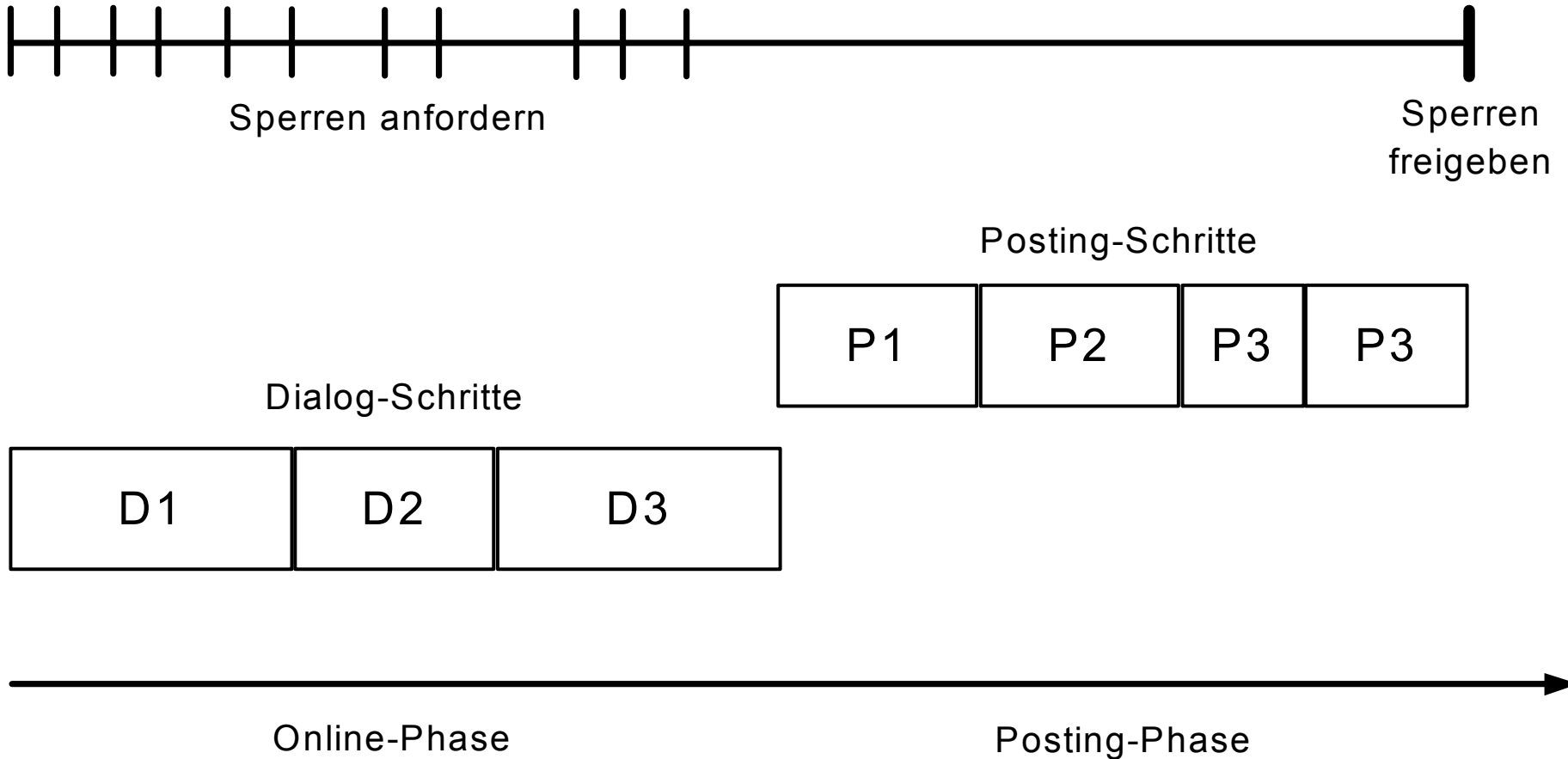


Interne Architektur von SAP R/3

SAP R/3



Transaktionsverarbeitung in SAP R/3



Data Warehouse-Anwendungen: OLAP~Online Analytical Processing

- Wie hat sich die Auslastung der Transatlantikflüge über die letzten zwei Jahre entwickelt?

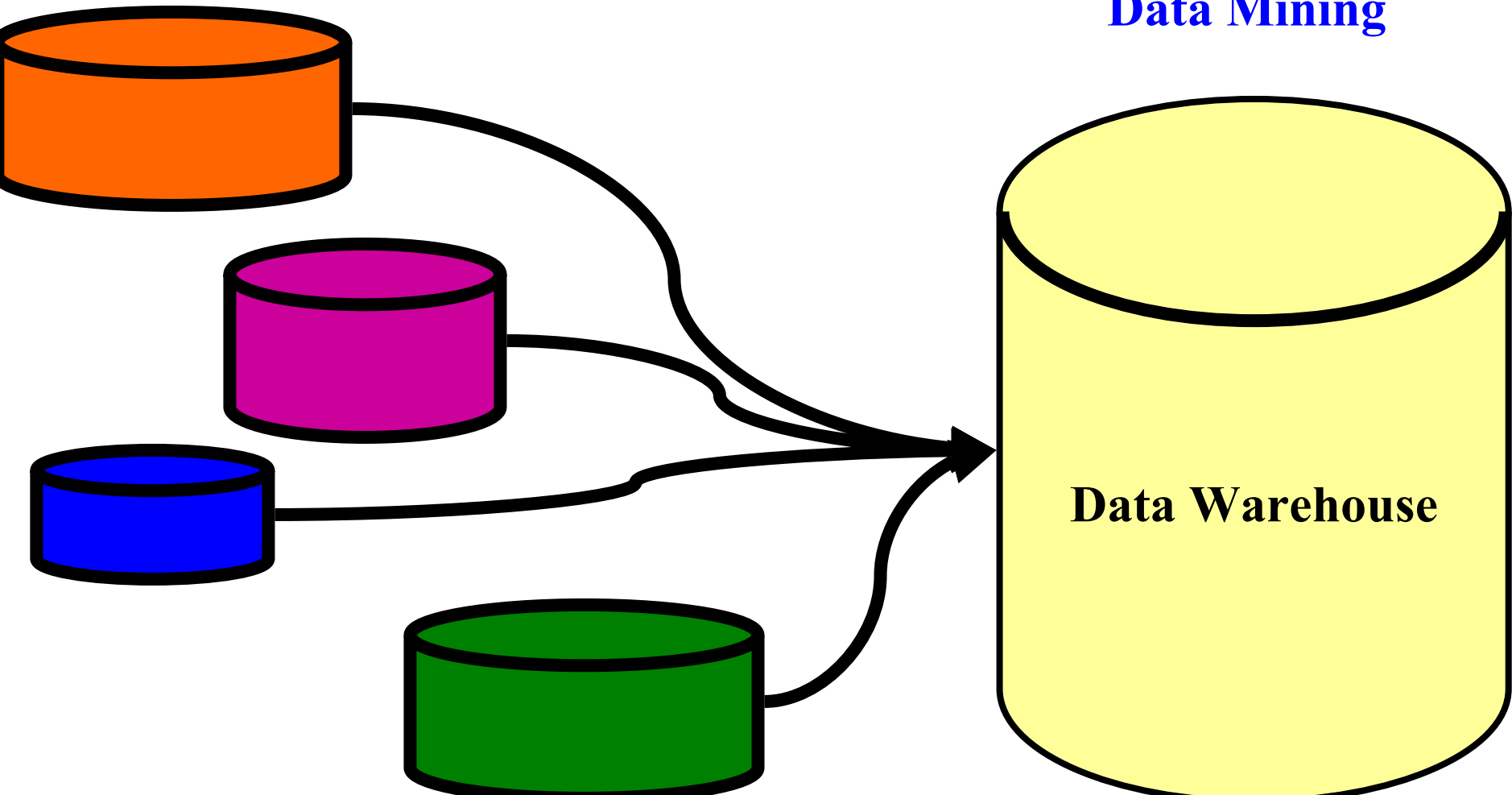
oder

- Wie haben sich besondere offensive Marketingstrategien für bestimmte Produktlinien auf die Verkaufszahlen ausgewirkt?

Sammlung und periodische Auffrischung der Data Warehouse-Daten

OLTP-Datenbanken
und andere Datenquellen

OLAP-Anfragen
Decision Support
Data Mining



Das Stern-Schema

Verkäufe					
VerkDatum	Filiale	Produkt	Anzahl	Kunde	Verkäufer
25-Jul-00	Passau	1347	1	4711	825
...

Filialen			
Filialenkennung	Land	Bezirk	...
Passau	D	Bayern	...
...

Kunden			
KundenNr	Name	wiealt	...
4711	Kemper	43	...
...

Verkäufer					
VerkäuferNr	Name	Fachgebiet	Manager	wiealt	...
825	Handyman	Elektronik	119	23	...
...

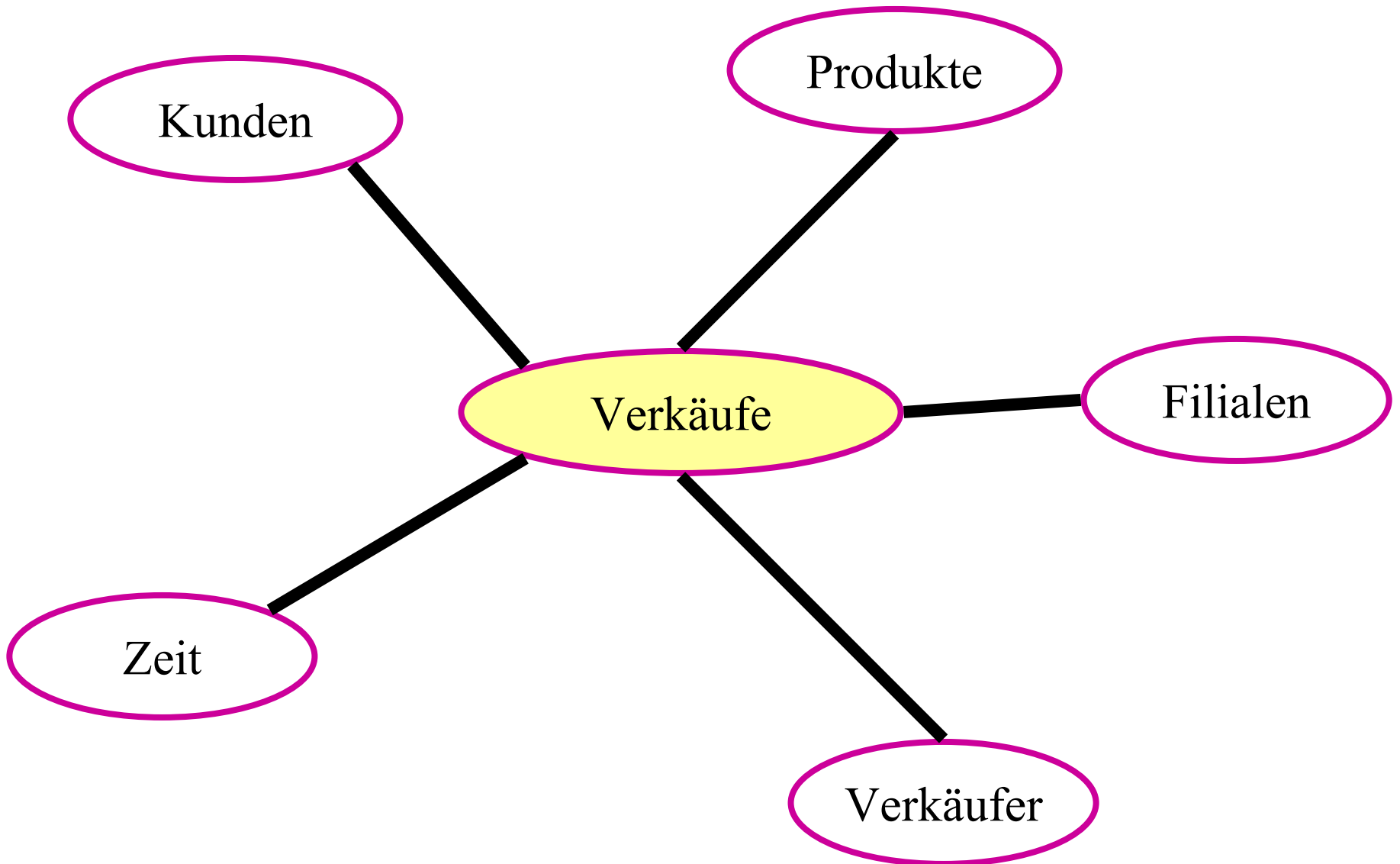
Zeit								
Datum	Tag	Monat	Jahr	Quartal	KW	Wochentag	Saison	...
...
25-Jul-00	25	Juli	2000	3	30	Dienstag	Hochsommer	...
...
18-Dec-01	18	Dezember	2001	4	52	Dienstag	Weihnachten	...
...

Produkte					
ProduktNr	Produkttyp	Produktgruppe	Produkthauptgruppe	Hersteller	...
1347	Handy	Mobiltelekom	Telekom	Siemens	...
...

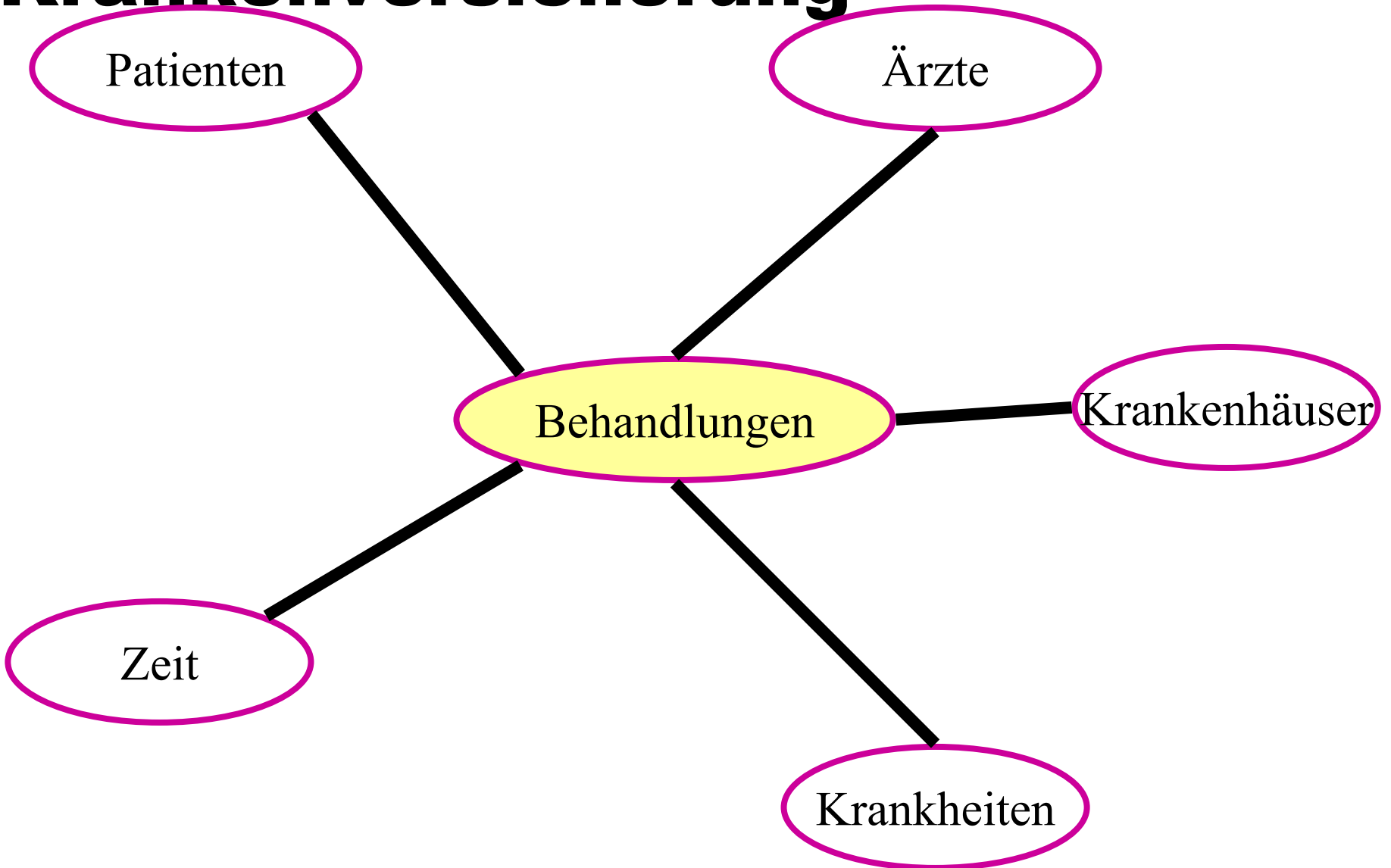
Stern-Schema bei Data Warehouse-Anwendungen

- Eine sehr große Faktentabelle
 - Alle Verkäufe der letzten drei Jahre
 - Alle Telefonate des letzten Jahres
 - Alle Flugreservierungen der letzten fünf Jahre
 - normalisiert
- Mehrere Dimensionstabellen
 - Zeit
 - Filialen
 - Kunden
 - Produkt
 - Oft nicht normalisiert

Das Stern-Schema: Handelsunternehmen



Das Stern-Schema: Krankenversicherung



Stern-Schema

Verkäufe

VerkDatum	Filiale	Produkt	Anzahl	Kunde	Verkäufer
25-Jul-00	Passau	1347	1	4711	825
...

Faktentabelle (SEHR groß)

Filialen

FilialenKennung	Land	Bezirk	...
Passau	D	Bayern	...
...

Kunden

KundenNr	Name	wieAlt	...
4711	Kemper	43	...
...

Dimensionstabellen (relativ klein)

Verkäufer

VerkäuferNr	Name	Fachgebiet	Manager	wieAlt	...
825	Handyman	Elektronik	119	23	...
...

Stern-Schema (cont'd)

Zeit							
Datum	Tag	Monat	Jahr	Quartal	KW	Wochentag	Saison
25-Jul-00	25	7	2000	3	30	Dienstag	Hochsommer
...		
18-Dec-01	18	12	2001	4	52	Dienstag	Weihnachten
...

Produkte					
ProduktNr	Produkttyp	Produktgruppe	Produkthauptgruppe	Hersteller	..
1347	Handy	Mobiltelekom	Telekom	Siemens	..
...

Nicht-normalisierte Dimensionstabellen: effizientere Anfrageauswertung

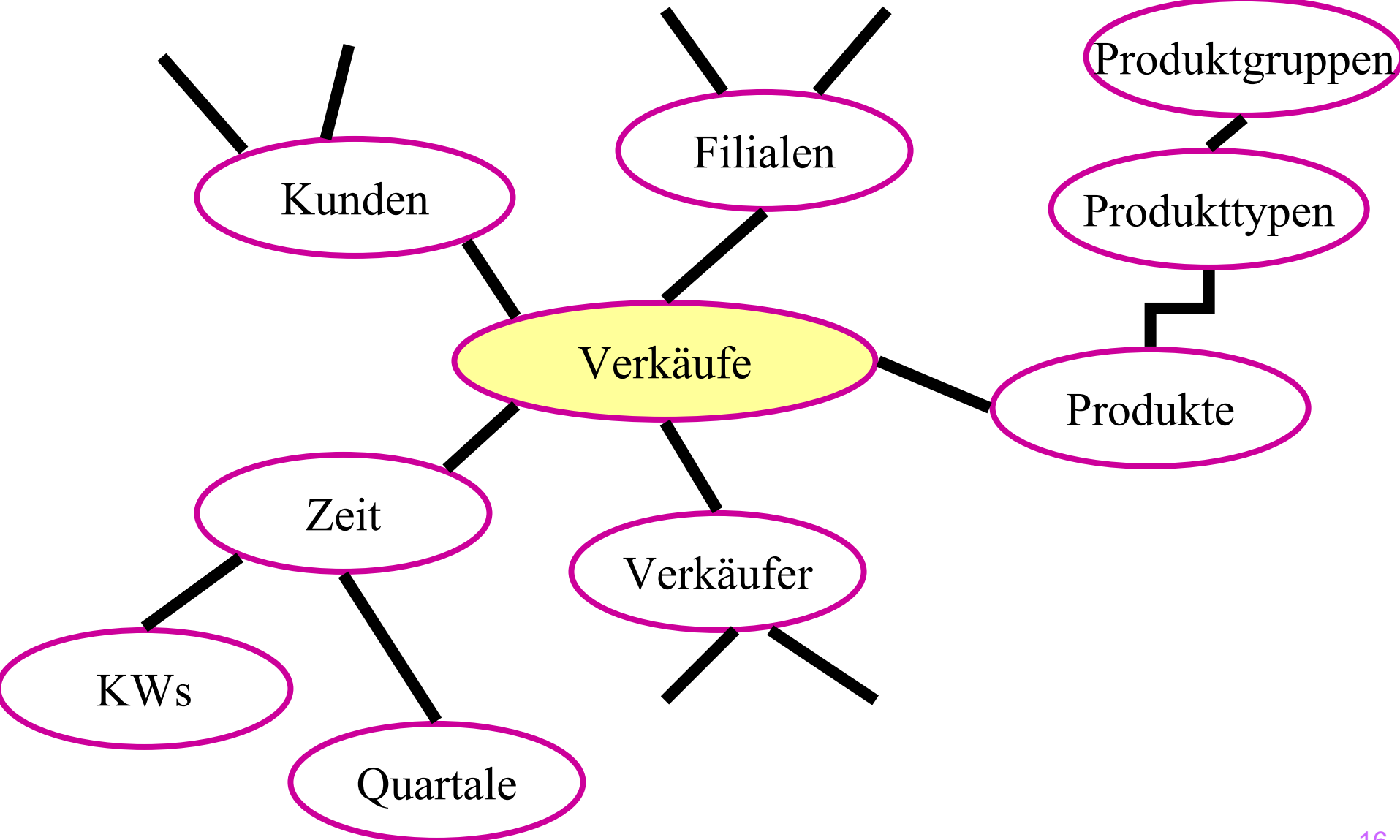
Zeit							
Datum	Tag	Monat	Jahr	Quartal	KW	Wochentag	Saison
25-Jul-00	25	7	2000	3	30	Dienstag	Hochsommer
...		
18-Dec-01	18	12	2001	4	52	Dienstag	Weihnachten
...

Datum → Monat → Quartal

Produkte					
ProduktNr	Produkttyp	Produktgruppe	Produkthauptgruppe	Hersteller	..
1347	Handy	Mobiltelekom	Telekom	Siemens	..
...

ProduktNr → Produkttyp → Produktgruppe → Produkthauptgruppe

Normalisierung führt zum Schneeflocken-Schema



Anfragen im Sternschema

select sum(v.Anzahl), p.Hersteller

from Verkäufe v, Filialen f, Produkte p, Zeit z, Kunden k

where z.Saison = 'Weihnachten' and

z.Jahr = 2001 and k.wieAlt < 30 and

p.Produkttyp = 'Handy' and f.Bezirk = 'Bayern' and

v.VerkDatum = z.Datum and v.Produkt = p.ProduktNr and

v.Filiale = f.FilialenKennung and v.Kunde = k.KundenNr

group by p.Hersteller;

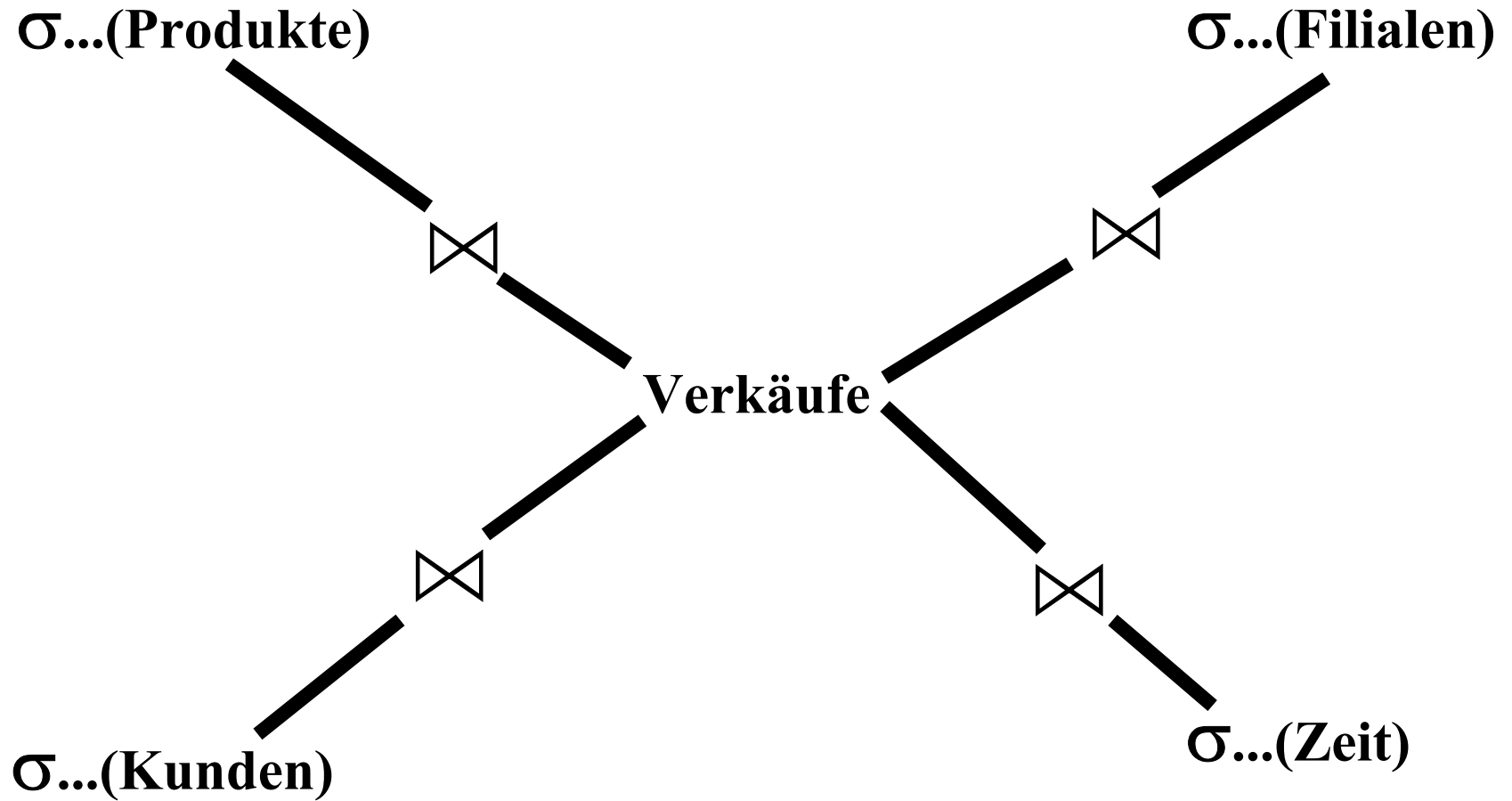


Einschränkung
der Dimensionen



Join-Prädikate

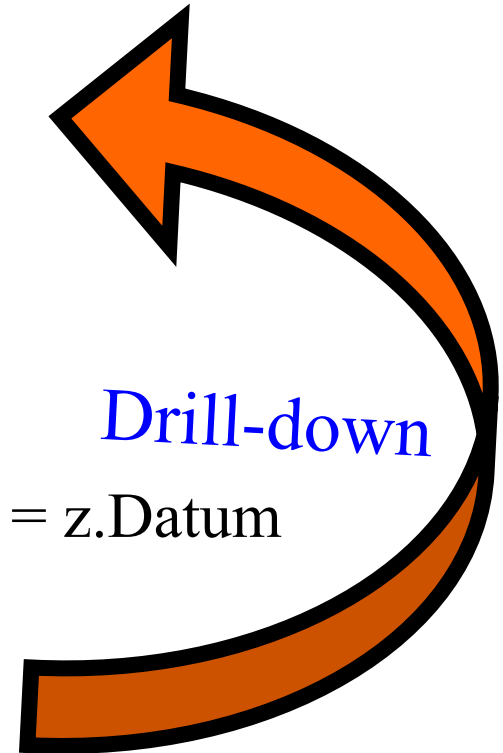
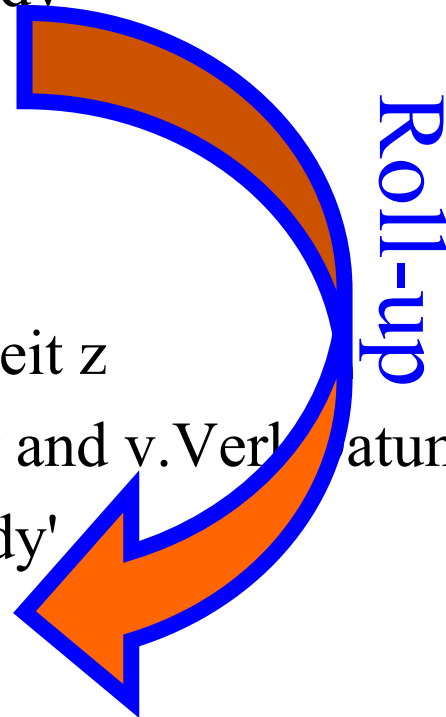
Algebra-Ausdruck



Roll-up/Drill-down-Anfragen

```
select Jahr, Hersteller, sum(Anzahl)
from Verkäufe v, Produkte p, Zeit z
where v.Produkt = p.ProduktNr and v.VerkDatum = z.Datum
      and p.Produkttyp = 'Handy'
group by p.Hersteller, z.Jahr;
```

```
select Jahr, sum(Anzahl)
from Verkäufe v, Produkte p, Zeit z
where v.Produkt = p.ProduktNr and v.VerkDatum = z.Datum
      and p.Produkttyp = 'Handy'
group by z.Jahr;
```



Ultimative Verdichtung

select sum(Anzahl)

from Verkäufe v, Produkte p

where v.Produkt = p.ProduktNr and p.Produkttyp = 'Handy';

Handyverkäufe nach Hersteller und Jahr		
Hersteller	Jahr	Anzahl
Siemens	1999	2.000
Siemens	2000	3.000
Siemens	2001	3.500
Motorola	1999	1.000
Motorola	2000	1.000
Motorola	2001	1.500
Bosch	1999	500
Bosch	2000	1.000
Bosch	2001	1.500
Nokia	1999	1.000
Nokia	2000	1.500
Nokia	2001	2.000

Handyverkäufe nach Jahr	
Jahr	Anzahl
1999	4.500
2000	6.500
2001	8.500

Handyverkäufe nach Hersteller	
Hersteller	Anzahl
Siemens	8.500
Motorola	3.500
Bosch	3.000
Nokia	4.500

Abb. 17.7: Analyse der Handyverkaufszahlen nach unterschiedlichen Dimensionen

Hersteller \ Jahr	1999	2000	2001	Σ
Siemens	2.000	3.000	3.500	8.500
Motorola	1.000	1.000	1.500	3.500
Bosch	500	1.000	1.500	3.000
Nokia	1.000	1.500	2.000	4.500
Σ	4.500	6.500	8.500	19.500

Abb. 17.8: Handyverkäufe nach Jahr und Hersteller

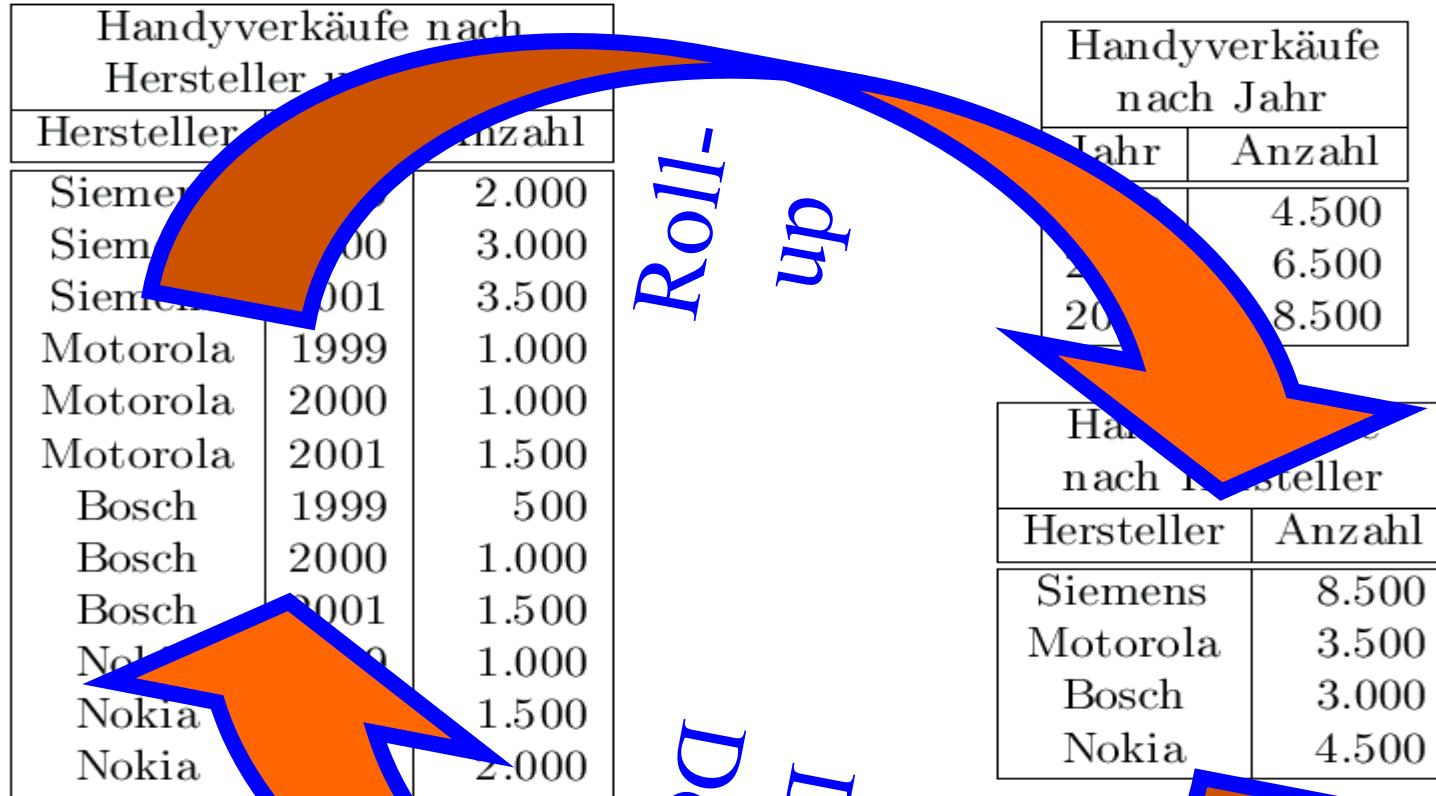
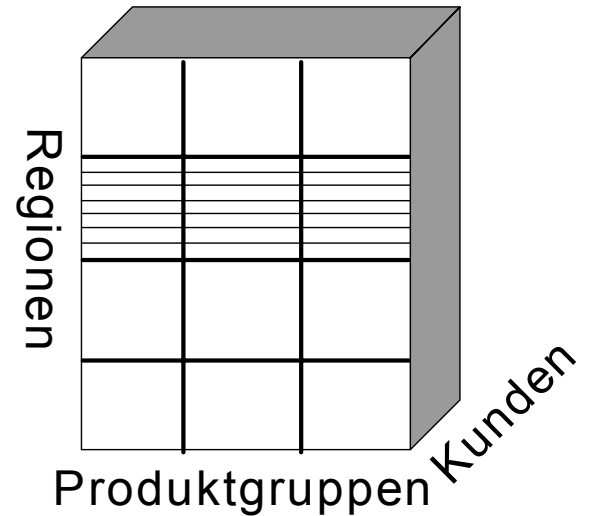
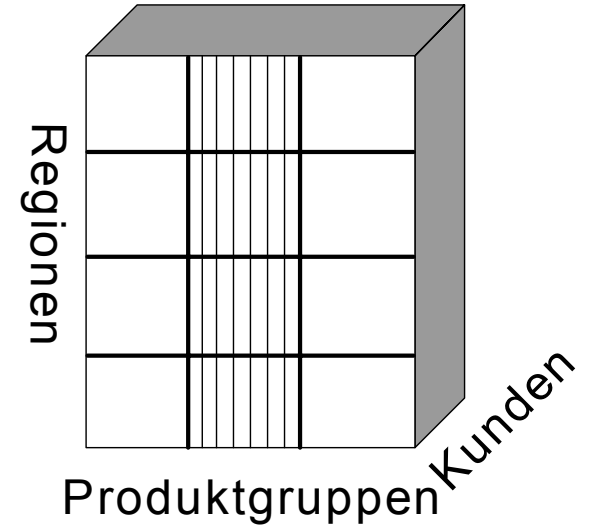
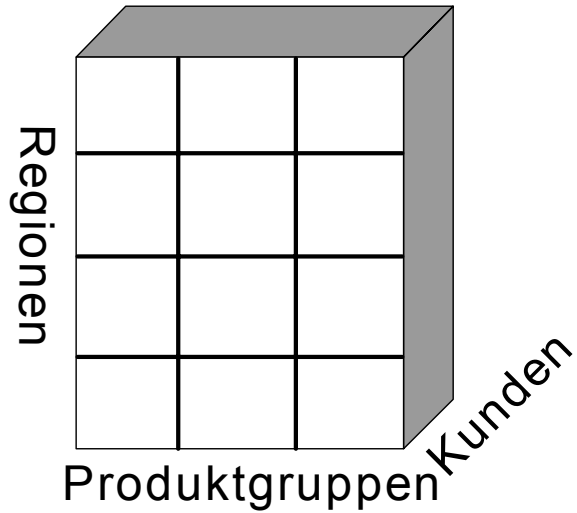


Abb. 17.7: Analyse von Handyverkaufszahlen nach unterschiedlichen Dimensionen

Hersteller \ Jahr	1999	2000	2001	Σ
Siemens	2.000	3.000	3.500	8.500
Motorola	1.000	1.000	1.500	3.500
Bosch	500	1.000	1.500	3.000
Nokia	1.000	1.500	2.000	4.500
Σ	4.500	6.500	8.500	19.500

Abb. 17.8: Handyverkäufe nach Jahr und Hersteller

Flexible Auswertungsmethoden: slice and dice



Materialisierung von Aggregaten

insert into Handy2DCube

```
( select p.Hersteller, z.Jahr, sum(v.Anzahl)
from Verkäufe v, Produkte p, Zeit z
where v.Produkt = p.ProduktNr and p.Produkttyp = 'Handy'
and v.VerkDatum = z.Datum
group by z.Jahr, p.Hersteller ) union
```

```
( select p.Hersteller, to_number(null), sum(v.Anzahl)
from Verkäufe v, Produkte p
where v.Produkt = p.ProduktNr and p.Produkttyp = 'Handy'
group by p.Hersteller ) union
```

```
( select null, z.Jahr, sum(v.Anzahl)
from Verkäufe v, Produkte p, Zeit z
where v.Produkt = p.ProduktNr and p.Produkttyp = 'Handy'
and v.VerkDatum = z.Datum
group by z.Jahr ) union
```

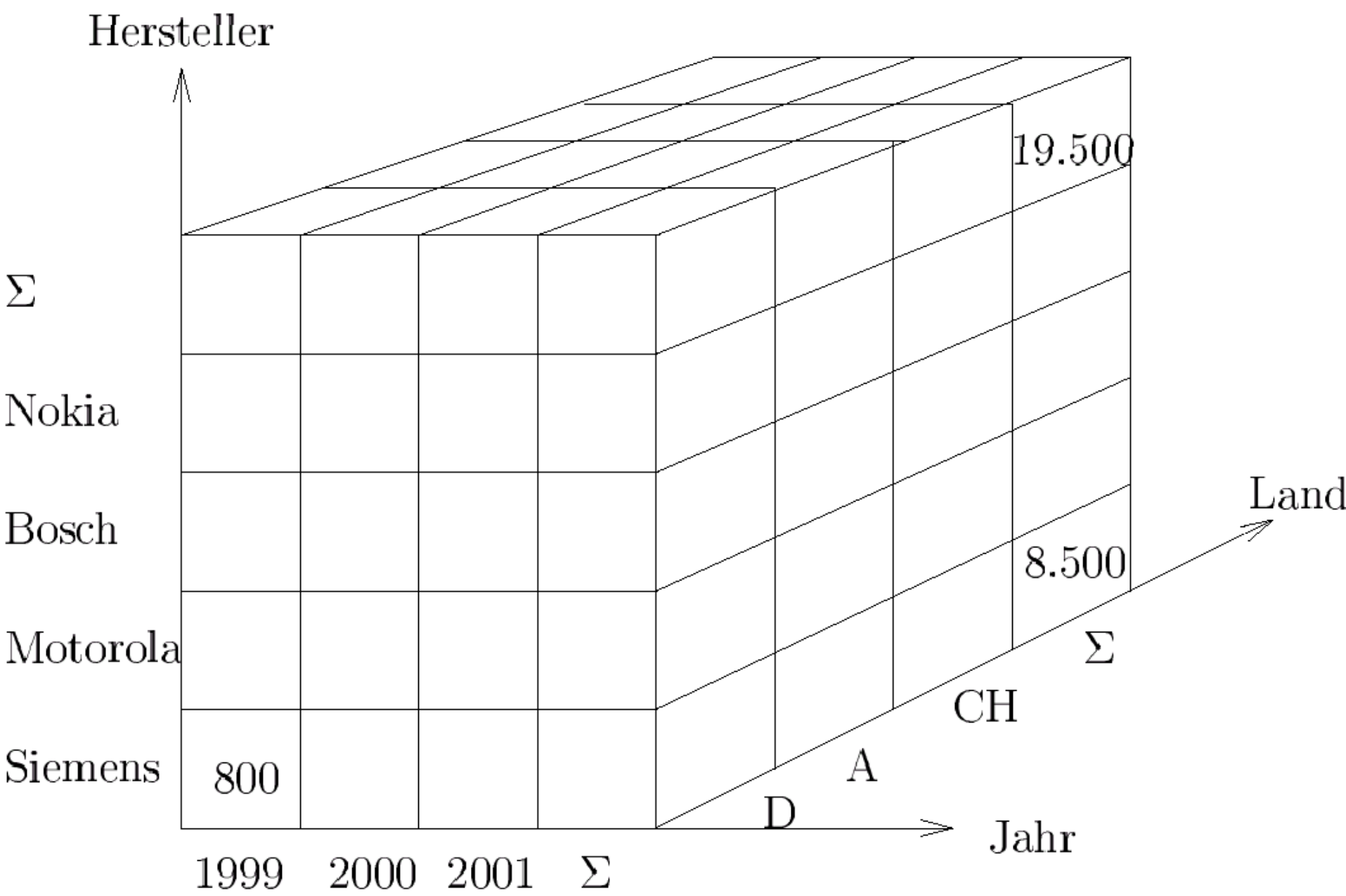
```
( select null, to_number(null), sum(v.Anzahl)
from Verkäufe v, Produkte p
where v.Produkt = p.ProduktNr and p.Produkttyp = 'Handy' );
```

Relationale Struktur der Datenwürfel

Handy2DCube		
Hersteller	Jahr	Anzahl
Siemens	1999	2.000
Siemens	2000	3.000
Siemens	2001	3.500
Motorola	1999	1.000
Motorola	2000	1.000
Motorola	2001	1.500
Bosch	1999	500
Bosch	2000	1.000
Bosch	2001	1.500
Nokia	2000	1.000
Nokia	2001	1.500
Nokia	2001	2.000
null	1999	4.500
null	2000	6.500
null	2001	8.500
Siemens	null	8.500
Motorola	null	3.500
Bosch	null	3.000
Nokia	null	4.500
null	null	19.500

Handy3DCube			
Hersteller	Jahr	Land	Anzahl
Siemens	1999	D	800
Siemens	1999	A	600
Siemens	1999	CH	600
Siemens	2000	D	1.200
Siemens	2000	A	800
Siemens	2000	CH	1.000
Siemens	2001	D	1.400
...
Motorola	1999	D	400
Motorola	1999	A	300
Motorola	1999	CH	300
...
Bosch
...
null	1999	D	...
null	2000	D	...
...
Siemens	null	null	8.500
...
null	null	null	19.500

Würfeldarstellung



Der **cube**-Operator

```
select p.Hersteller, z.Jahr, f.Land, sum(v.Anzahl)
from Verkäufe v, Produkte p, Zeit z, Filialen f
where v.Produkt = p.ProduktNr and p.Produkttyp = 'Handy'
      and v.VerkDatum = z.Datum and v.Filiale = f.Filialenkennung
group by cube (z.Jahr, p.Hersteller, f.Land);
```

Wiederverwendung von Teil-Aggregaten

```
insert into VerkäufeProduktFilialeJahr
```

```
( select v.Produkt, v.Filiale, z.Jahr, sum(v.Anzahl)
```

```
from Verkäufe v, Zeit z
```

```
where v.VerkDatum = z.Datum
```

```
group by v.Produkt, v.Filiale, z.Jahr );
```

Wie kann die folgende Anfrage nun effizient beantwortet werden?

```
select v.Produkt, v.Filiale, sum(v.Anzahl)
```

```
from Verkäufe v
```

```
group by v.Produkt, v.Filiale
```

Wiederverwendung von Teil-Aggregaten

```
select v.Produkt, v.Filiale, sum(v.Anzahl)
from Verkäufe v
group by v.Produkt, v.Filiale
```



```
select v.Produkt, v.Filiale, sum(v.Anzahl)
from VerkäufeProduktFilialeJahr v
group by v.Produkt, v.Filiale
```

Wiederverwendung von Teil-Aggregaten

```
select v.Produkt, z.Jahr, sum(v.Anzahl)
```

```
from Verkäufe v, Zeit z
```

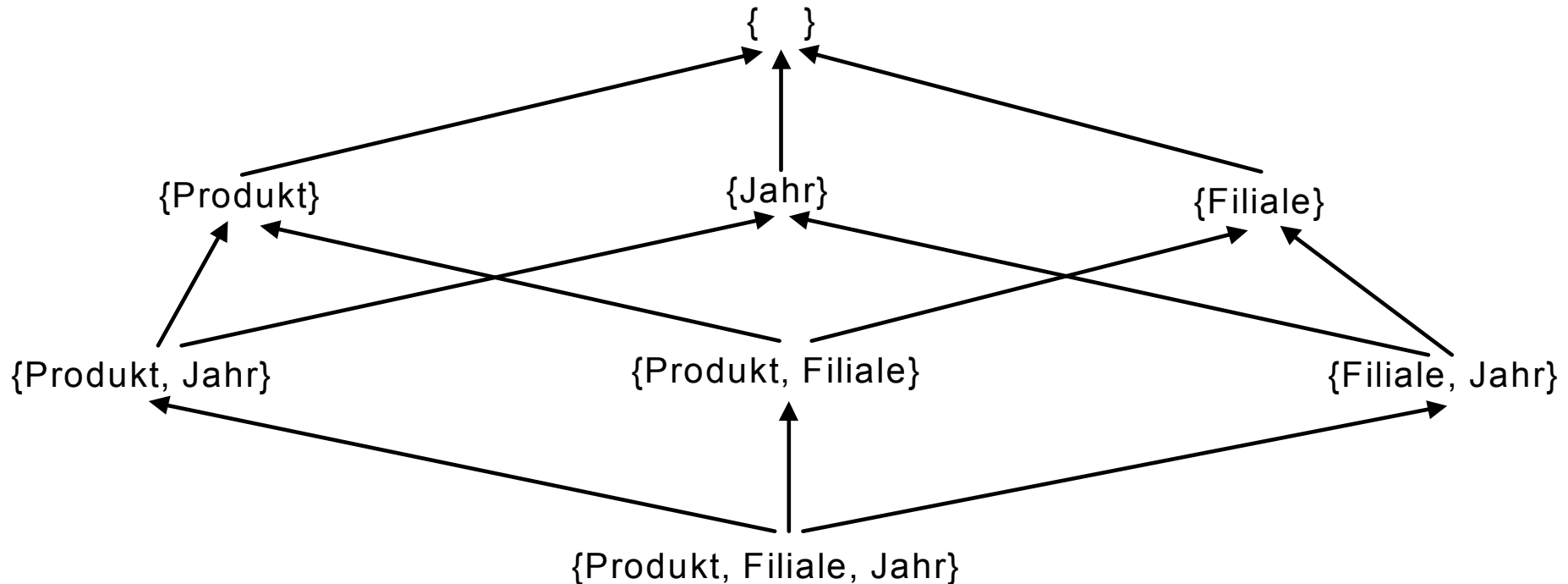
```
where v.VerkDatum = z.Datum
```

```
group by v.Produkt, z.Jahr
```

... kann ähnlich mit VerkäufeProduktFilialeJahr rationalisiert werden.

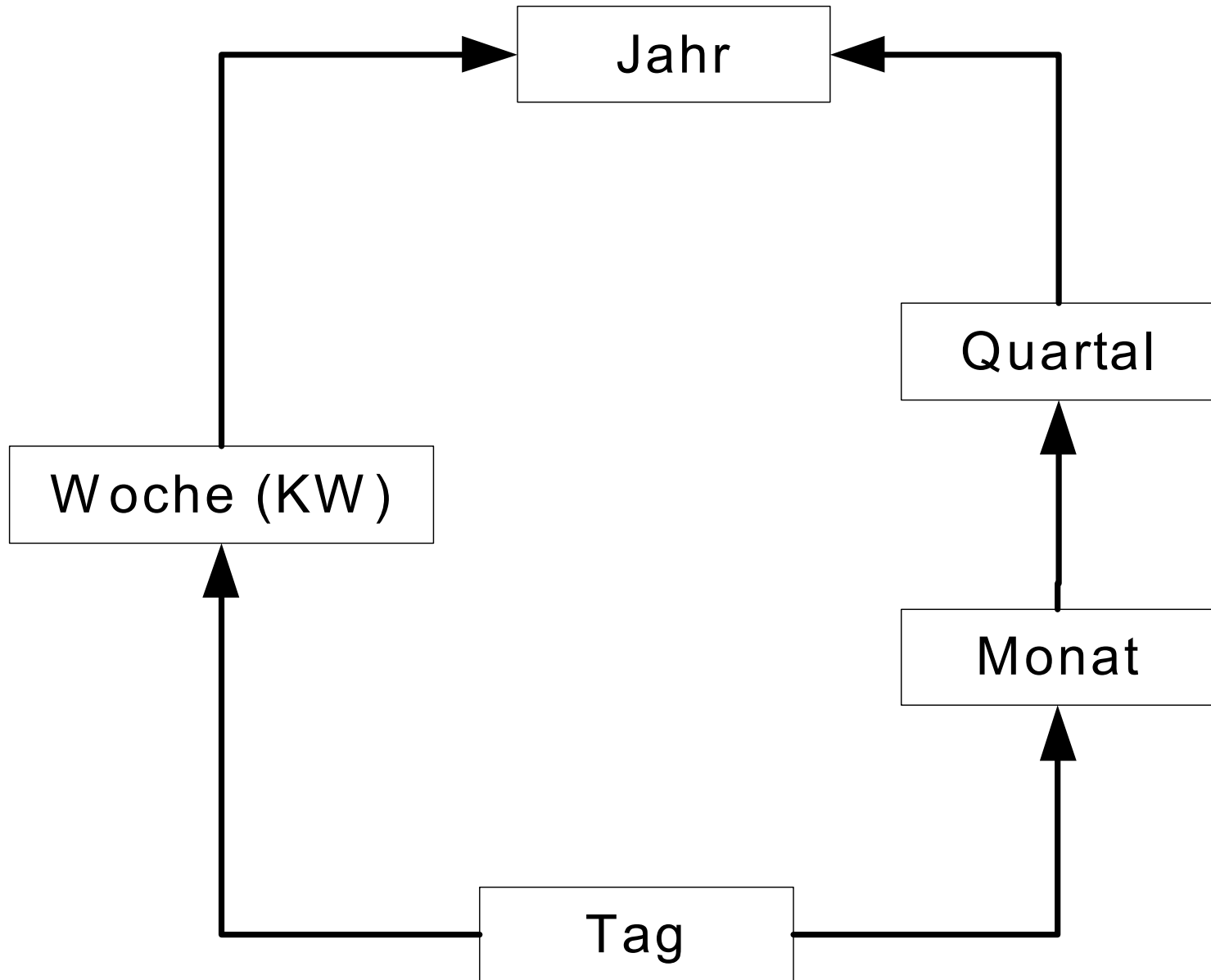
(Wie? → gute Übung!)

Die Materialisierungs-Hierarchie



- Teilaggregate T sind für eine Aggregation A wiederverwendbar wenn es einen gerichteten Pfad von T nach A gibt
- Also $T \rightarrow \dots \rightarrow A$
- Man nennt diese Materialisierungshierarchie auch einen Verband (Engl. *Lattice*)

Die Zeit-Hierarchie



Weitere Decision-Support Anfrage-Typen

- Top N-Anfragen
 - Ich will nur die N besten Treffer erhalten und nicht alle 5 Millionen
 - Muss bei der Anfrageoptimierung berücksichtigt werden
- Online Aggregation
 - Man berechnet das Ergebnis approximativ
 - Je länger die Anfrage läuft, desto genauer wird das Ergebnis

Top N-Anfragen

Select A.*

From Angestellte A, Abteilungen abt

Where A.Abteilung = abt.AbteilungsNr and abt.Ort = Passau

Order by A.Gehalt

Stop after 20

Online-Aggregation

Select abt.Ort, avg(A.Gehalt)

From Angestellte A, Abteilungen abt

Where A.Abteilung = abt.AbteilungsNr

Group by abt.Ort

Data Mining



Klassifikation

Assoziationsregeln

Clustering

Klassifikationsregeln

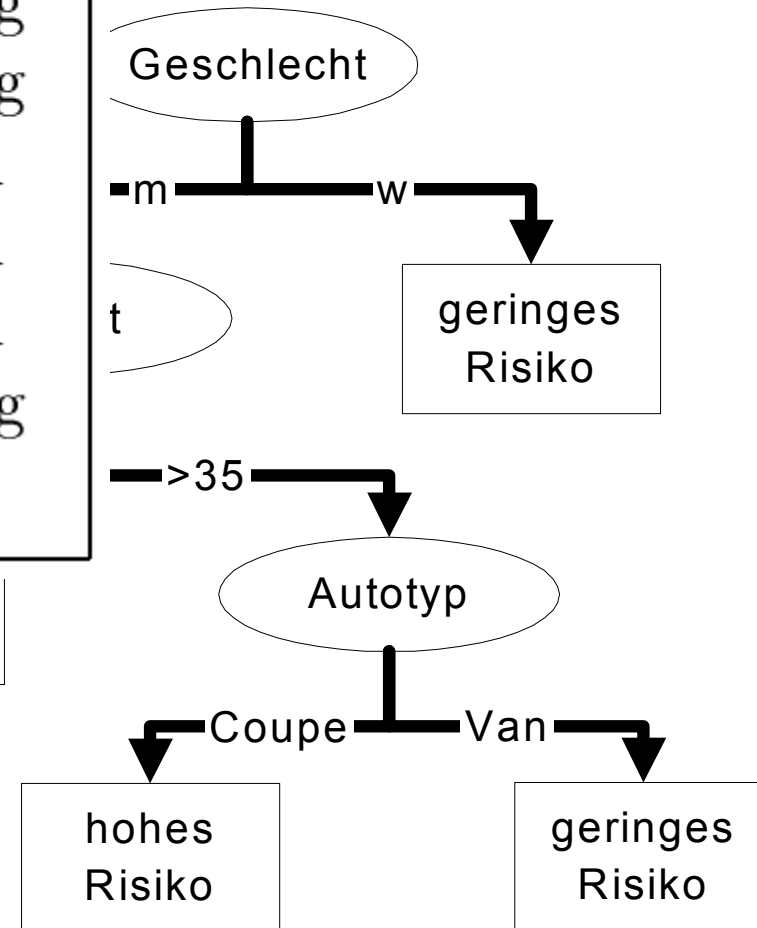
- Vorhersageattribute
 - V_1, V_2, \dots, V_n
- Vorhergesagtes Attribut A
- Klassifikationsregel
 - $P_1(V_1) \wedge P_2(V_2) \wedge \dots \wedge P_n(V_n) \rightarrow A = c$
 - Prädikate P_1, P_2, \dots, P_n
 - Konstante c
- Beispielregel

$(\text{wieAlt} > 35) \wedge (\text{Geschlecht} = \text{'m'}) \wedge (\text{Autotyp} = \text{'Coupé'}) \rightarrow (\text{Risiko} = \text{'hoch'})$

Klassifikations/Entscheidungsbaum

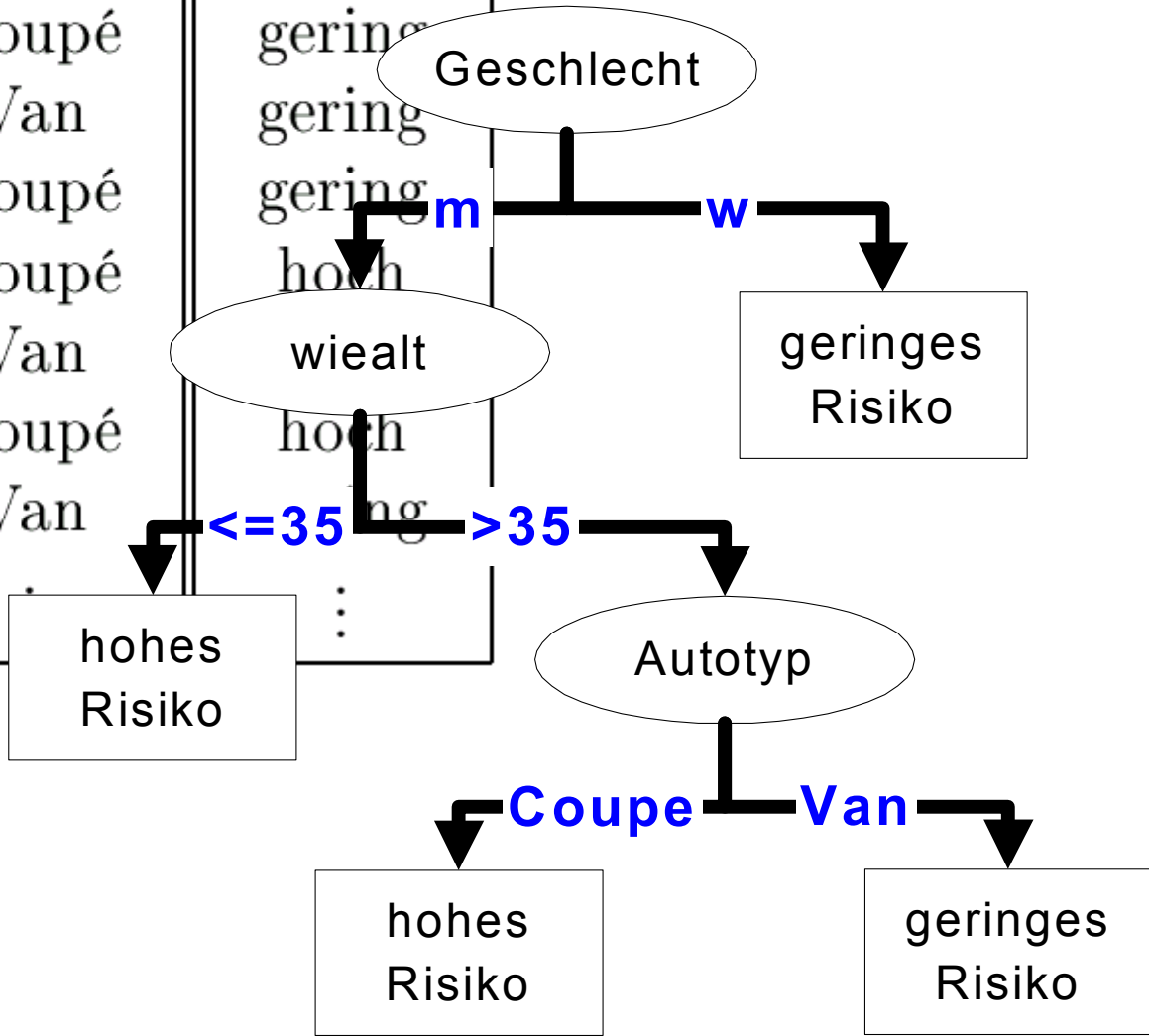
Schadenshöhe			
wiealt	Geschlecht	Autotyp	Schäden
45	w	Van	gering
18	w	Coupé	gering
22	w	Van	gering
38	w	Coupé	gering
19	m	Coupé	hoch
24	m	Van	hoch
40	m	Coupé	hoch
40	m	Van	gering
⋮	⋮	⋮	⋮

hohes
Risiko



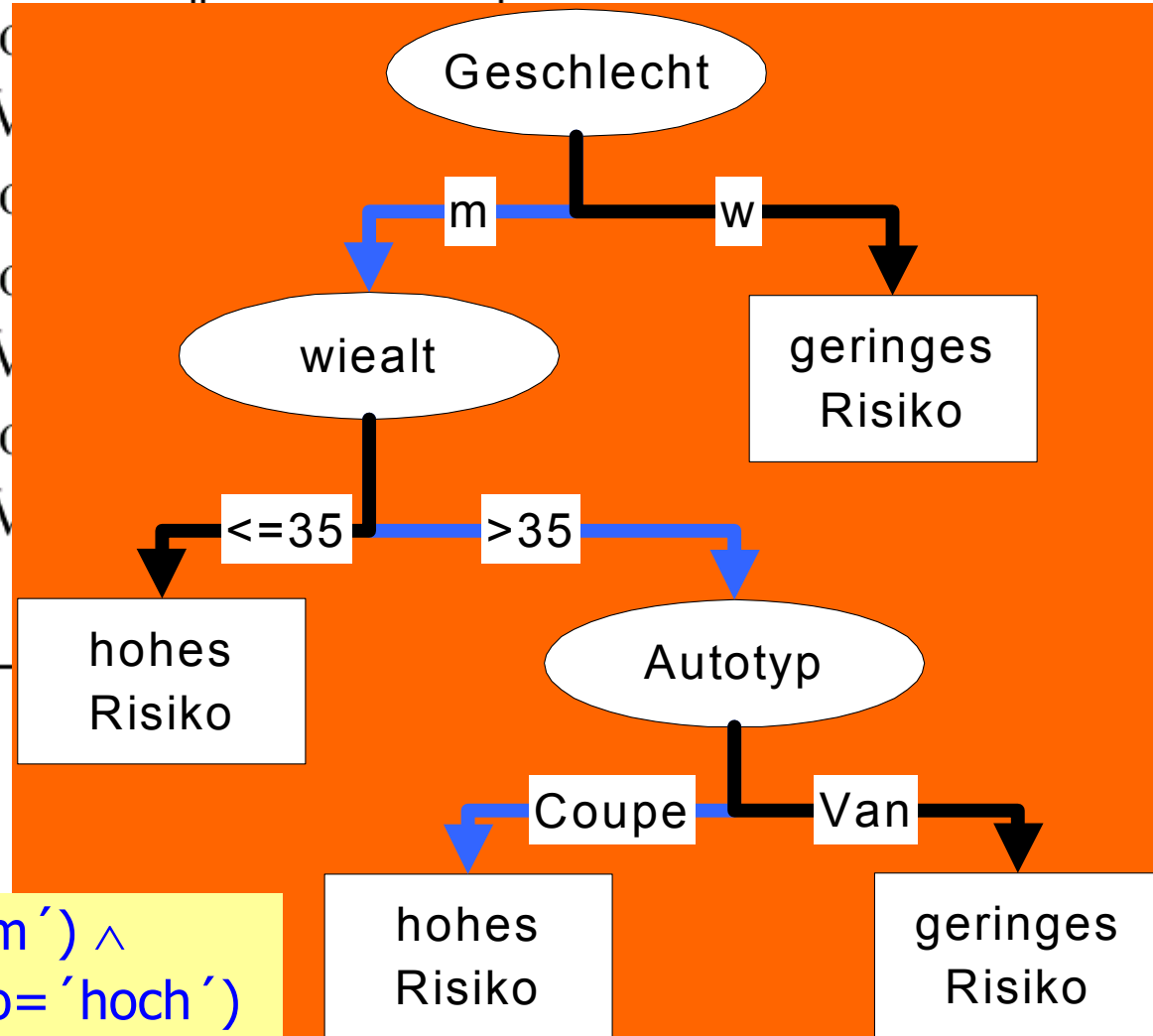
Klassifikations/Entscheidungsbaum

Schadenshöhe			
wiealt	Geschlecht	Autotyp	Schäden
45	w	Van	gering
18	w	Coupé	gering
22	w	Van	gering
38	w	Coupé	gering
19	m	Coupé	hoch
24	m	Van	gering
40	m	Coupé	hoch
40	m	Van	gering
⋮	⋮	⋮	⋮



Klassifikations/Entscheidungsbaum

Schadenshöhe			
wiealt	Geschlecht	Autotyp	Schäden
45	w	Van	gering
18	w	Coupe	gering
22	w	Van	gering
38	w	Coupe	gering
19	m	Coupe	gering
24	m	Van	gering
40	m	Coupe	gering
40	m	Van	gering
⋮	⋮		



$(\text{wieAlt} > 35) \wedge (\text{Geschlecht} = \text{'m'}) \wedge (\text{Autotyp} = \text{'Coupé'}) \rightarrow (\text{Risiko} = \text{'hoch'})$

Wie werden Entscheidungs-/Klassifikationsbäume erstellt

- Trainingsmenge
 - Große Zahl von Datensätzen, die in der Vergangenheit gesammelt wurden
 - Sie dient als Grundlage für die Vorhersage von „neu ankommenden“ Objekten
 - Beispiel: neuer Versicherungskunde wird gemäß dem Verhalten seiner „Artgenossen“ eingestuft
- Rekursives Partitionieren
 - Fange mit einem Attribut an und spalte die Tupelmengen
 - Jede dieser Teilmengen wird rekursiv weiter partitioniert, bis nur noch gleichartige Objekte in der jeweiligen Partition sind

Top-Down Klassifikationsbaum-Aufbau

- Eingabe: Knoten n , Partition D , Zerlegungsmethode S
- Ausgabe: Klassifikationsbaum für D , Wurzel n

- $\text{BuildTree}(n, D, S)$
 - Wende S auf D an und finde die richtige Zerlegung
 - Wenn eine gute Partitionierung gefunden ist
 - Kreiere zwei Kinder n_1 und n_2
 - Partitioniere D in D_1 und D_2
 - $\text{BuildTree}(n_1, D_1, S)$
 - $\text{BuildTree}(n_2, D_2, S)$

Assoziationsregeln

- Beispielregel
 - Wenn jemand einen PC kauft, dann kauft er/sie auch einen Drucker.
- Konfidenz
 - Dieser Wert legt fest, bei welchem Prozentsatz der Datenmenge, bei der die Voraussetzung (linke Seite) erfüllt ist, die Regel (rechte Seite) auch erfüllt ist.
 - Eine Konfidenz von 80% für unsere Beispielregel sagt aus, dass vier Fünftel der Leute, die einen PC gekauft haben, auch einen Drucker dazu gekauft haben.
- Support
 - Dieser Wert legt fest, wieviele Datensätze überhaupt gefunden wurden, um die Gültigkeit der Regel zu verifizieren.
 - Bei einem Support von 1% wäre also jeder Hundertste Verkauf ein PC zusammen mit einem Drucker.

VerkaufsTransaktionen	
TransID	Produkt
111	Drucker
111	Papier
111	PC
111	Toner
222	PC
222	Scanner
333	Drucker
333	Papier
333	Toner
444	Drucker
444	PC
555	Drucker
555	Papier
555	PC
555	Scanner
555	Toner

Verkaufstransaktionen

Warenkörbe

- Finde alle Assoziationsregeln $L \rightarrow R$
 - mit einem Support größer als **minsupp** und
 - einer Konfidenz von mindestens **minconf**
- Dazu sucht man zunächst die sogenannten frequent itemsets, also Produktmengen, die in mindestens minsupp der Einkaufswägen/Transaktionen enthalten sind
- Der A Priori-Algorithmus basiert auf der Erkenntnis, dass alle Teilmengen eines FI auch FIs sein müssen

A Priori Algorithmus

für alle Produkte

überprüfe ob es ein frequent itemset ist, also in mindestens minsupp Einkaufswägen enthalten ist

$k:=1$

iteriere solange

für jeden *frequent itemset* I_k mit k Produkten

generiere alle *itemsets* I_{k+1} mit $k+1$ Produkten und $I_k \subset I_{k+1}$

lies alle Einkäufe einmal (sequentieller Scan auf der Datenbank)

und überprüfe, welche der $(k+1)$ -elementigen *itemset*-Kandidaten mindestens minsupp mal vorkommen

$k:=k+1$

bis keine neuen frequent itemsets gefunden werden

A Priori-Algorithmus

Minsupp=3

Zwischenergebnisse

VerkaufsTransaktionen	
TransID	Produkt
111	Drucker
111	Papier
111	PC
111	Toner
222	PC
222	Scanner
333	Drucker
333	Papier
333	Toner
444	Drucker
444	PC
555	Drucker
555	Papier
555	PC
555	Scanner
555	Toner

Disqualifiziert

FI-Kandidat	Anzahl
{Drucker}	4
{Papier}	3
{PC}	4
{Scanner}	2
{Toner}	3
{Drucker, Papier}	3
{Drucker, PC}	3
{Drucker, Scanner}	
{Drucker, Toner}	3
{Papier, PC}	2
{Papier, Scanner}	
{Papier, Toner}	3
{PC, Scanner}	
{PC, Toner}	2
{Scanner, Toner}	

A Priori-Algorithmus

VerkaufsTransaktionen	
TransID	Produkt
111	Drucker
111	Papier
111	PC
111	Toner
222	PC
222	Scanner
333	Drucker
333	Papier
333	Toner
444	Drucker
444	PC
555	Drucker
555	Papier
555	PC
555	Scanner
555	Toner

Zwischenergebnisse	
FI-Kandidat	Anzahl
{Drucker, Papier}	3
{Drucker, PC}	3
{Drucker, Scanner}	
{Drucker, Toner}	3
{Papier, PC}	2
{Papier, Scanner}	
{Papier, Toner}	3
{PC, Scanner}	
{PC, Toner}	2
{Scanner, Toner}	
{Drucker, Papier, PC}	2
{Drucker, Papier, Toner}	3
{Drucker, PC, Toner}	2
{Papier, PC, Toner}	2

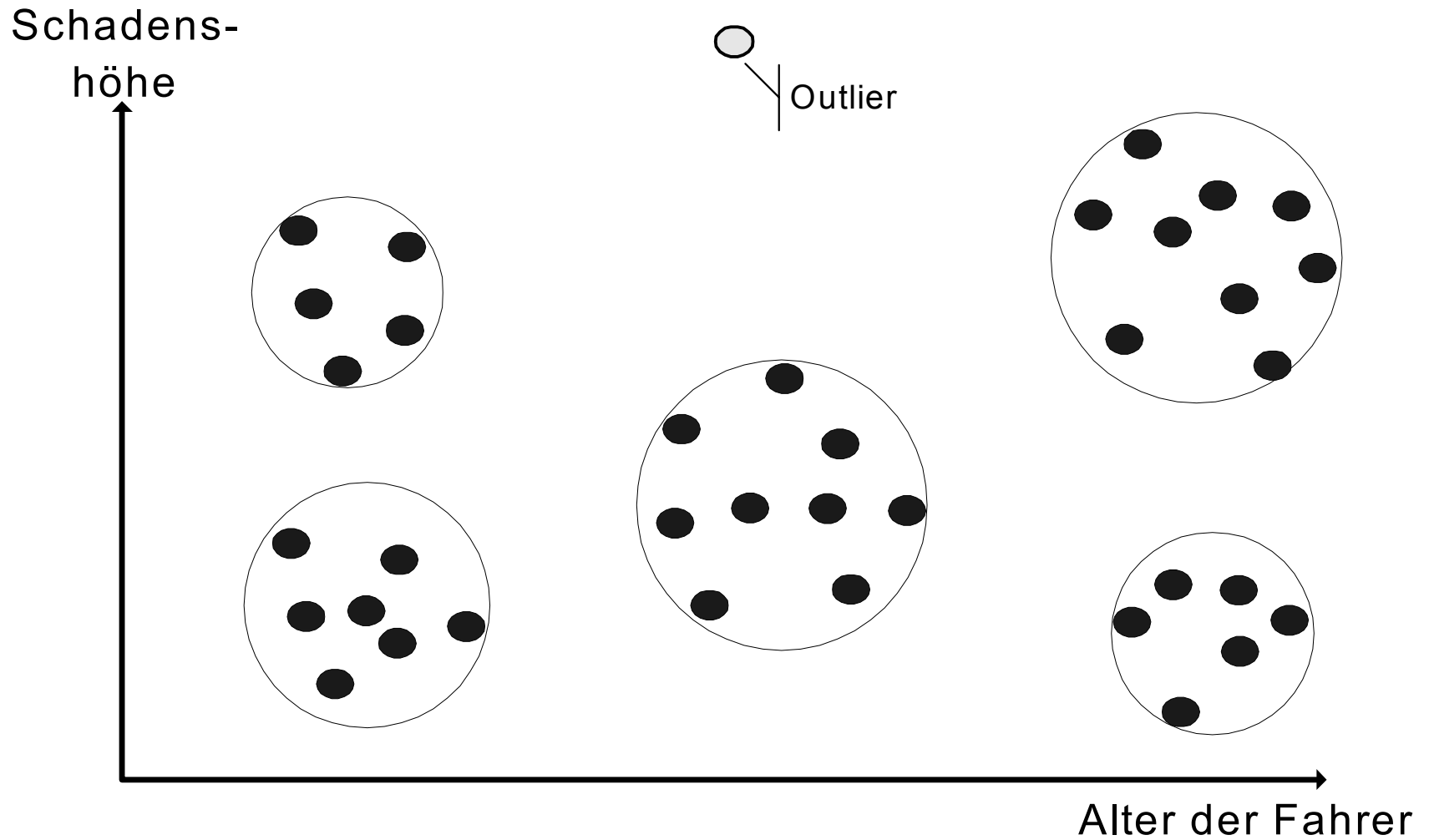
Ableitung von Assoziationsregeln aus den *frequent itemsets*

- Betrachte jeden FI mit hinreichend viel *support*
- Bilde alle nicht-leeren Teilmengen $L \subset \text{FI}$ und untersuche die Regel
 - $L \rightarrow \text{FI} - L$
 - Die Konfidenz dieser Regel berechnet sich als
 - $\text{Konfidenz}(L \rightarrow \text{FI} - L) = \text{support}(\text{FI}) / \text{support}(L)$
 - Wenn die Konfidenz ausreicht, also größer *minconf* ist, behalte diese Regel
- Betrachte $\text{FI} = \{\text{Drucker, Papier, Toner}\}$
 - $\text{Support} = 3$
- Regel: $\{\text{Drucker}\} \rightarrow \{\text{Papier, Toner}\}$
 - $\text{Konfidenz} = S(\{\text{Drucker, Papier, Toner}\}) / S(\{\text{Drucker}\})$
 $= (3/5) / (4/5)$
 $= 3/4 = 75 \%$

Erhöhung der Konfidenz

- Vergrößern der linken Seite (dadurch Verkleinern der rechten Seite) führt zur Erhöhung der Konfidenz
 - Formal: $L \subset L^+$, $R \subset R^-$
 - $\text{Konfidenz}(L \rightarrow R) \leq C(L^+ \rightarrow R^-)$
- Beispiel-Regel: $\{\text{Drucker}\} \rightarrow \{\text{Papier, Toner}\}$
 - $\text{Konfidenz} = S(\{\text{Drucker, Papier, Toner}\}) / S(\{\text{Drucker}\})$
 $= (3/5) / (4/5)$
 $= 3/4 = 75\%$
- Beispiel-Regel: $\{\text{Drucker, Papier}\} \rightarrow \{\text{Toner}\}$
 - $\text{Konf.} = S(\{\text{Drucker, Papier, Toner}\}) / S(\{\text{Drucker, Papier}\})$
 $= (3/5) / (3/5)$
 $= 1 = 100\%$

Clustering



Clustering-Algorithmus

- Greedy Heuristik
- Lese sequentiell alle Datensätze
- Für den nächsten Datensatz r bestimme
 - Für alle bisher existierenden Cluster denjenigen c , dessen Zentrum den kürzesten Abstand zu r hat
 - Wenn $\text{distance}(r, \text{center}(c)) \leq \text{epsilon}$
 - Füge r in c ein
 - Anderenfalls lege einen neuen Cluster c' an, der zunächst nur r enthält
- Funktioniert solange ganz gut, wie die Cluster in den Hauptspeicher passen

Beispiel-System: Microsoft® SQL Server 2000™ Analysis Services

```
CREATE MINING MODEL [MemberCards] (  
    [customer Id] LONG KEY ,  
    [Yearly Income] TEXT DISCRETE ,  
    [Member Card Type] TEXT DISCRETE PREDICT,  
    [Marital Status] TEXT DISCRETE )  
USING Microsoft_Ddecision_Trees
```