Datawarehouse Inhalt

- Datenbankentwurf für Data Warehouse
- Star Join
- Roll-Up/Drill-Down-Anfragen
- Materialisierung von Aggregaten
- Der Cube-Operator
- Data Warehouse-Architekturen
- Data Mining

Datawarehouse

- Zwei Arten von Datenbankanwendungen:
  - **OLTP** (*Online Transaction Processing*):
    zB: **Bestellungen in einem Handelsunternehmen**
    .
  - **OLAP** (*Online Analytical Processing*):
    zB: **Auswirkungen gewisser Marketingstrategien**.

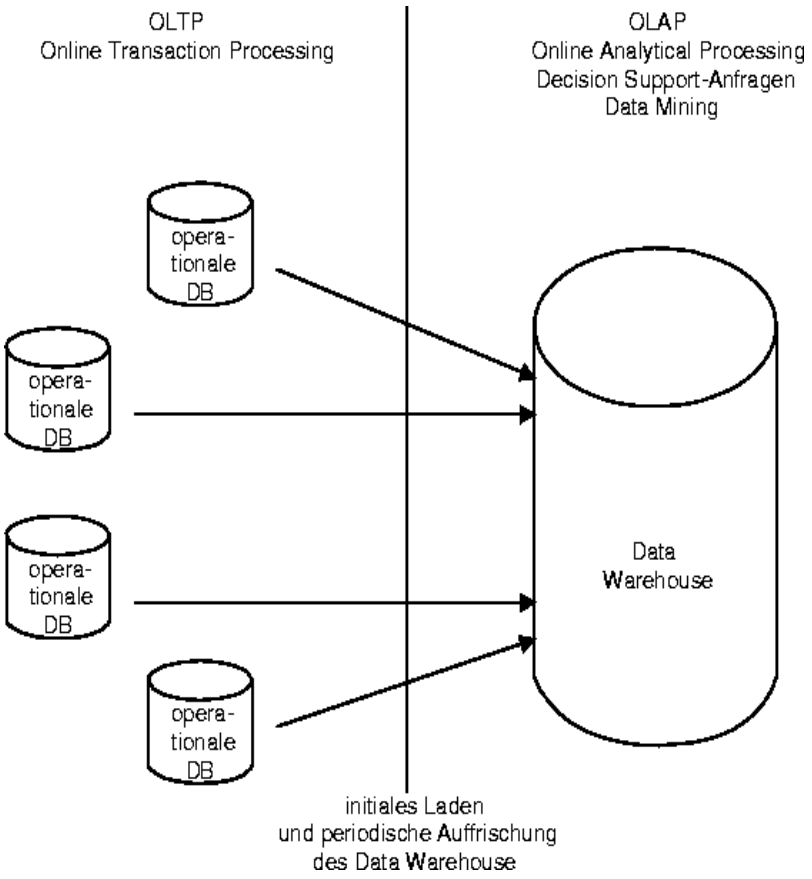
    - OLAP-Anwendungen verarbeiten **sehr große Datenmengen** und greifen auf historische Daten zurück.
    - Sie bilden die **Grundlage für Decision-Support-Systeme.**

Datawarehouse

- OLTP- und OLAP-Anwendungen sollten
  - **nicht auf demselben Datenbestand** arbeiten aus folgenden Gründen:
    - OLTP-Datenbanken sind auf **Änderungstransaktionen** mit begrenzten Datenmengen hin **optimiert**.
    - OLAP-Auswertungen benötigen **Daten aus verschiedenen Datenbanken** in konsolidierter, integrierter Form.
    - Typischerweise wird beim **Transferieren** der Daten aus den operationalen Datenbanken eine **Verdichtung** (sum,avg, …) durchgeführt, da nun nicht mehr einzelne Transaktionen im Vordergrund stehen, sondern ihre **Aggregation**.

Datawarehouse

- **Aufbau eines *Data Warehouse***,
  - in dem die für Decision-Support-Anwendungen notwendigen Daten in konsolidierter/**aggregierter Form** gesammelt werden.



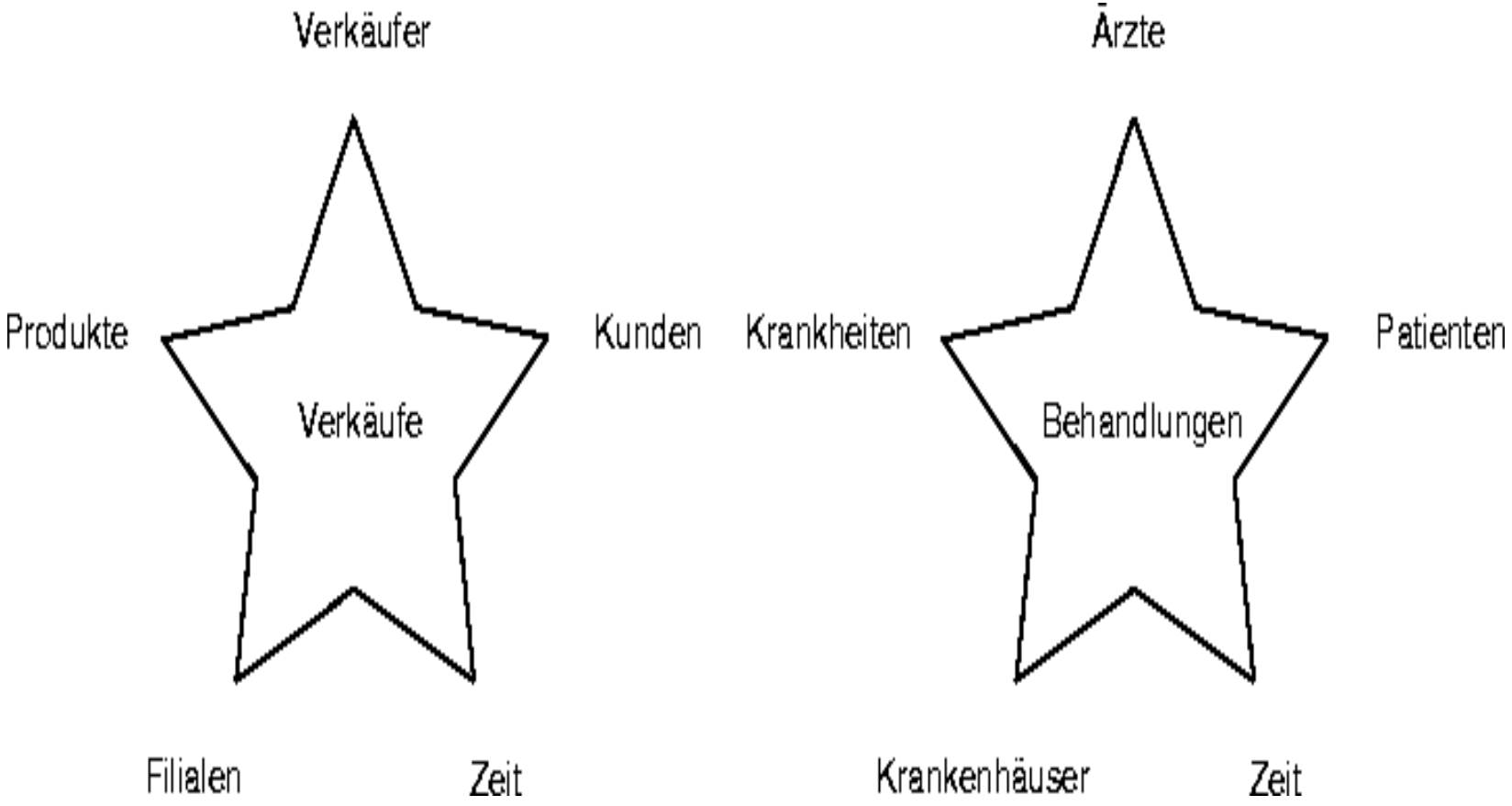
Datenbankentwurf für Data Warehouse

- Als Datenbankschema für Data Warehouse-Anwendungen hat sich das sogenannte **Sternschema** (engl.: *star scheme*) durchgesetzt.

- Dieses Schema besteht aus
  - **einer Faktentabelle** und

    mehreren Dimensionstabellen

Sternschemata



Fakten- und Dimensionstabellen

Faktentabelle

Verkäufe					
VerkDatum	Filiale	Produkt	Anzahl	Kunde	Verkäufer
30-Jul-96	Passau	1347	1	4711	825
...	...	...	...	...	...

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

Kunden			
KundenNr	Name	wiealt	...
4711	Kemper	38	...
...	...	...	...

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

Zeit								
Datum	Tag	Monat	Jahr	Quartal	KW	Wochentag	Saison	...
...	...	...	...	...	...	...	...	...
30-Jul-96	30	Juli	1996	3	31	Dienstag	Hochsommer	...
...	...	...	...	...	...	...	...	...
23-Dec-97	27	Dezember	1997	4	52	Dienstag	Weihnachten	...
...	...	...	...	...	...	...	...	...

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

Fakten- und Dimensionstabellen

- **Faktentabelle** *Verkäufe* können mehrere Millionen Tupel sein, während die

- **Dimensionstabelle** *Produkte* vielleicht 10.000 Einträge und die

  Dimensionstabelle *Zeit* vielleicht 1.000 Einträge (für die letzen drei Jahre) aufweist.

Dimensionstabellen nicht normalisiert

- *Zeit*-Dimension
  - Es lassen sich alle Attribute aus dem Schlüsselattribut *Datum* ableiten.
  - Trotzdem ist die explizite Speicherung dieser Dimension sinnvoll, da **Abfragen** nach Verkäufen in bestimmten Quartalen oder an bestimmten Wochentagen dadurch **effizienter** durchgeführt werden können

Dimensionstabellen nicht normalisiert

- Achtung:
  Dimensionstabellen nicht normalisiert:
  Tabelle: Produkte hat folg. funktionalen Abhängigkeiten:
  - ProduktNR ->Produkttyp ,
  - Produkttyp -> Produktgruppe und
  - Produktgruppe -> Produkthauptgruppe .

Prodnr,	Prodtyp,	Prodgruppe,	ProdHptgruppe,	Hersteller,…
123,	Handy,	Mobiltelekom,	Telekom,	Siemens,

....

Dimensionstabellen nicht normalisiert

- Die Verletzung der Normalformen in den Dimensionstabellen ist bei Decision-Support-Systemen nicht so gravierend,
  - Da die **Daten nur selten verändert werden** und
  - da der durch die **Redundanz** verursachte erhöhte Speicherbedarf bei den **relativ kleinen Dimensionstabellen** im Vergleich zu der großen (normalisierten) Faktentabelle nicht so sehr ins Gewicht fällt.

Star Join

- Sternschema führt bei typischen Abfragen zu sogenannten Star Joins:
  - Welche Handys (d.h. von welchen Herstellern)
  - haben junge Kunden
  - in den bayrischen Filialen
  - zu Weihnachten 1996 gekauft ?

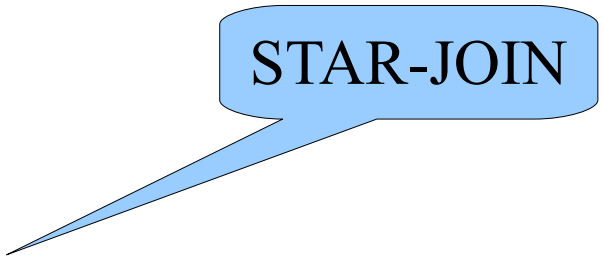
Star Join

Wieviele und welche Handys (d.h. von welchen Herstellern) haben junge Kunden in den bayrischen Filialen zu Weihnachten 1996 gekauft ?

```
select sum(v.Anzahl), p.Hersteller
from  Verkäufe v,
  Filialen f, Produkte p, Zeit z, Kunden k
where
  z.Saison = 'Weihnachten' and
  z.Jahr = 1996 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;
```



Roll-Up/Drill-Down-Anfragen

- Der **Verdichtungsgrad** bei einer SQL-Anfrage wird durch die **group by**-Klausel gesteuert.

- Werden **mehr** Attribute in die **group by**-Klausel aufgenommen, spricht man von einem **drill down**.

- Werden **weniger** Attribute in die **group by**-Klausel aufgenommen, spricht man von einem **roll up.**

Drill-Down-Anfragen

- Wieviel Handys wurden von welchem **Hersteller** in welchem **Jahr** verkauft ? (**drill down**)

```
select Hersteller, 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 p.Hersteller, z.Jahr;
```

Roll-Up -Anfragen (entlang der Dimension Datum)

- Durch das **Weglassen der Zeitangabe** aus der **group by**-Klausel (und der **select**-Klausel) entsteht ein **roll up entlang der Dimension z.Jahr:**

- Wieviel Handys wurden von welchem Hersteller verkauft ?

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

group  by p.Hersteller;
```

Roll-Up -Anfragen (entlang Dim Hersteller)

- Durch das Weglassen der Herstellerangabe aus der **group by**-Klausel (und der **select**-Klausel) entsteht ein **roll up entlang der Dimension p.Hersteller:**

- Wieviel Handys wurden in welchem Jahr verkauft ?

```
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;
```

Roll-Up -Anfragen
vollständiges ROLL-UP

- Die ultimative Verdichtung besteht im **vollständigen Weglassen der group-by-Klausel**. Das Ergebnis besteht aus einem Wert, nämlich 19.500:

- Wieviel Handys wurden verkauft ?

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

Roll-Up/Drill-Down-Anfragen

Drill down

Handyverkäufe nach Hersteller und Jahr		
Hersteller	Jahr	Anzahl
Siemens	1994	2.000
Siemens	1995	3.000
Siemens	1996	3.500
Motorola	1994	1.000
Motorola	1995	1.000
Motorola	1996	1.500
Bosch	1994	500
Bosch	1995	1.000
Bosch	1996	1.500
Nokai	1995	1.000
Nokai	1996	1.500
Nokai	1996	2.000

Roll up

Handyverkäufe nach Jahr	
Jahr	Anzahl
1994	4.500
1995	6.500
1996	8.500

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

n -dimensionales Spreadsheet

- Durch eine sogenannte **cross tabulation** (Kreuztabelle) können die Ergebnisse obiger 3 Anfragen in einem *n* -dimensionalen Spreadsheet (einem 2-dimensionalen Datenwürfel **data cube**.) zusammengefaßt werden.

Hersteller \ Jahr	1994	1995	1996	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
Nokai	1.000	1.500	2.000	4.500
Σ	4.500	6.500	8.500	19.500

Materialisierung von Aggregaten

- Da es sehr zeitaufwendig ist, die Aggregation(zB. **Sum**()) jedesmal neu zu berechnen, empfiehlt es sich, sie zu **materialisieren**, d.h.

- die **vorberechneten Aggregate verschiedener Detaillierungsgrade in einer Relation abzulegen.**

  - Es folgen einige SQL-Statements, welche die linke Tabelle der folg. Abbildung erzeugen. Mit dem **null**-Wert wird markiert, dass entlang dieser Dimension die Werte aggregiert wurden.

Materialisierung von Aggregaten

Handy2DCube		
Hersteller	Jahr	Anzahl
Siemens	1994	2.000
Siemens	1995	3.000
Siemens	1996	3.500
Motorola	1994	1.000
Motorola	1995	1.000
Motorola	1996	1.500
Bosch	1994	500
Bosch	1995	1.000
Bosch	1996	1.500
Nokai	1995	1.000
Nokai	1996	1.500
Nokai	1996	2.000
null	1994	4.500
null	1995	6.500
null	1996	8.500
Siemens	null	8.500
Motorola	null	3.500
Bosch	null	3.000
Nokai	null	4.500
null	null	19.500

Handy3DCube			
Hersteller	Jahr	Land	Anzahl
Siemens	1994	D	800
Siemens	1994	A	600
Siemens	1994	CH	600
Siemens	1995	D	1.200
Siemens	1995	A	800
Siemens	1995	CH	1.000
Siemens	1996	D	1.400
...	...	...	...
Motorola	1994	D	400
Motorola	1994	A	300
Motorola	1994	CH	300
...	...	...	...
Bosch	...	...	...
...	...	...	...
null	1994	D	...
null	1995	D	...
...	...	...	...
Siemens	null	null	8.500
...	...	...	...
null	null	null	19.500

Materialisierung von Aggregaten

```
create table Handy2DCube (
Hersteller      varchar(20),
Jahr            integer,
Anzahl          integer);
```

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 v.VerkDatum = z.Datum
 and p.Produkttyp = 'Handy'
 group  by z.Jahr, p.Hersteller
)

union
```

Materialisierung von Aggregaten

```
-- roll up
(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
```

Materialisierung von Aggregaten

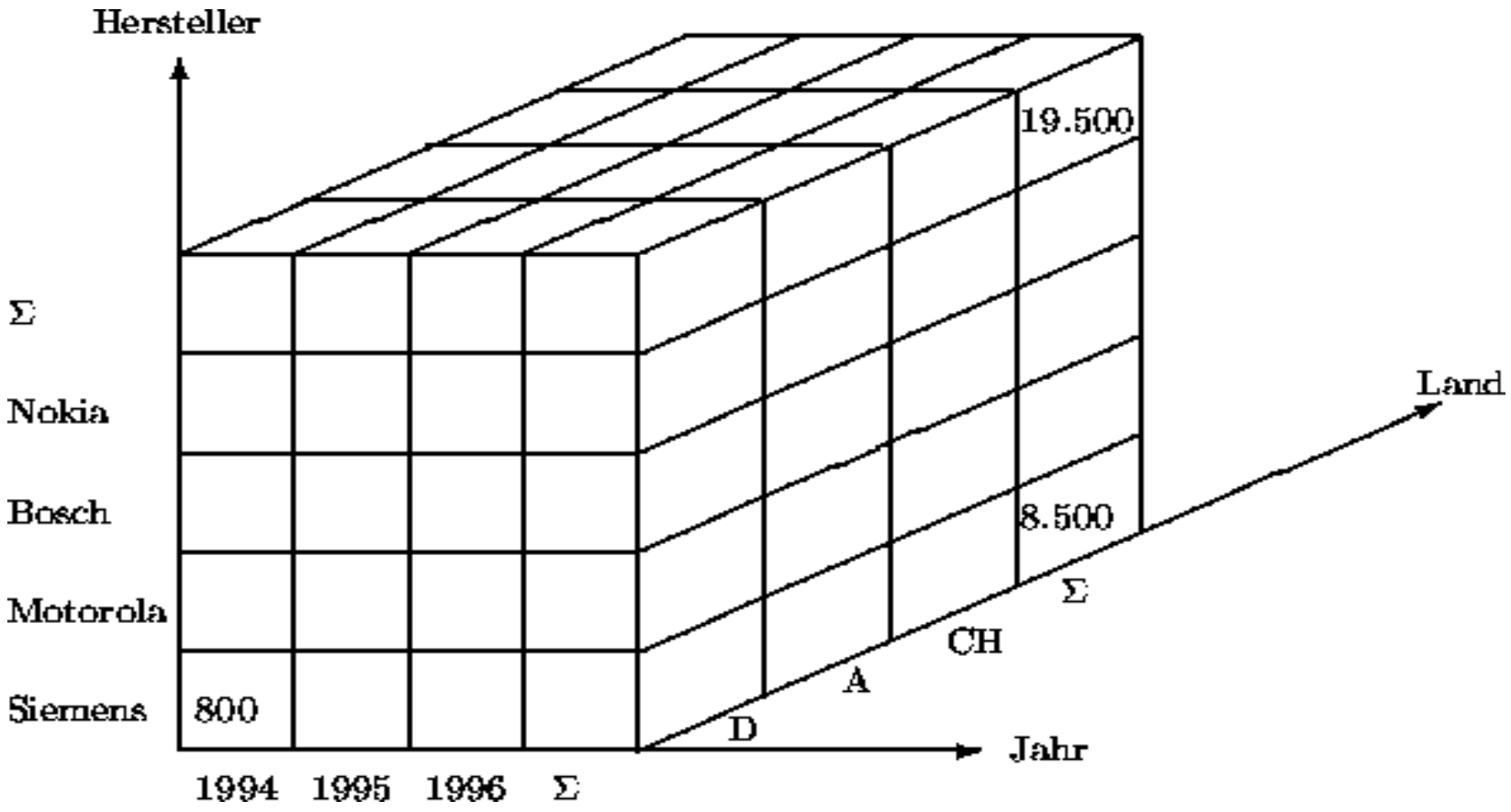
```
-- roll up
      (select null, z.Jahr, sum(v.Anzahl)
       from Verkäufe v, Produkte p, Zeit z
       where v.Produkt = p.ProduktNr
       and p.Produkttyp = 'Handy'
       and   v.VerkDatum = z.Datum
       group by z.Jahr
      )
     union
     (select null, to_number(null),sum(v.Anzahl)
   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

Materialisierung von Aggregaten

- Offenbar ist es recht mühsam, diese Art von Anfragen zu formulieren, da **bei *n* Dimensionen insgesamt 2^n Unteranfragen** formuliert und mit **union** verbunden werden müssen.

- Außerdem sind solche Anfragen extrem **zeitaufwendig** auszuwerten, da jede Aggregation individuell berechnet wird, **obwohl man viele Aggregate aus anderen** (noch nicht so stark verdichteten) Aggregaten **berechnen** könnte.

Der Cube-Operator



Der Cube-Operator

- Um der mühsamen Anfrageformulierung und der ineffizienten Auswertung zu begegnen, wurde der

- **SQL-Operator namens cube** eingeführt.

  Zur Erläuterung wollen wir ein 3-dimensionales Beispiel konstruieren, indem wir auch entlang der zusätzlichen Dimension *Filiale.Land* ein *drill down* vorsehen:

Der Cube-Operator

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

group  by z.Jahr, p.Hersteller, f.Land

with cube;
```

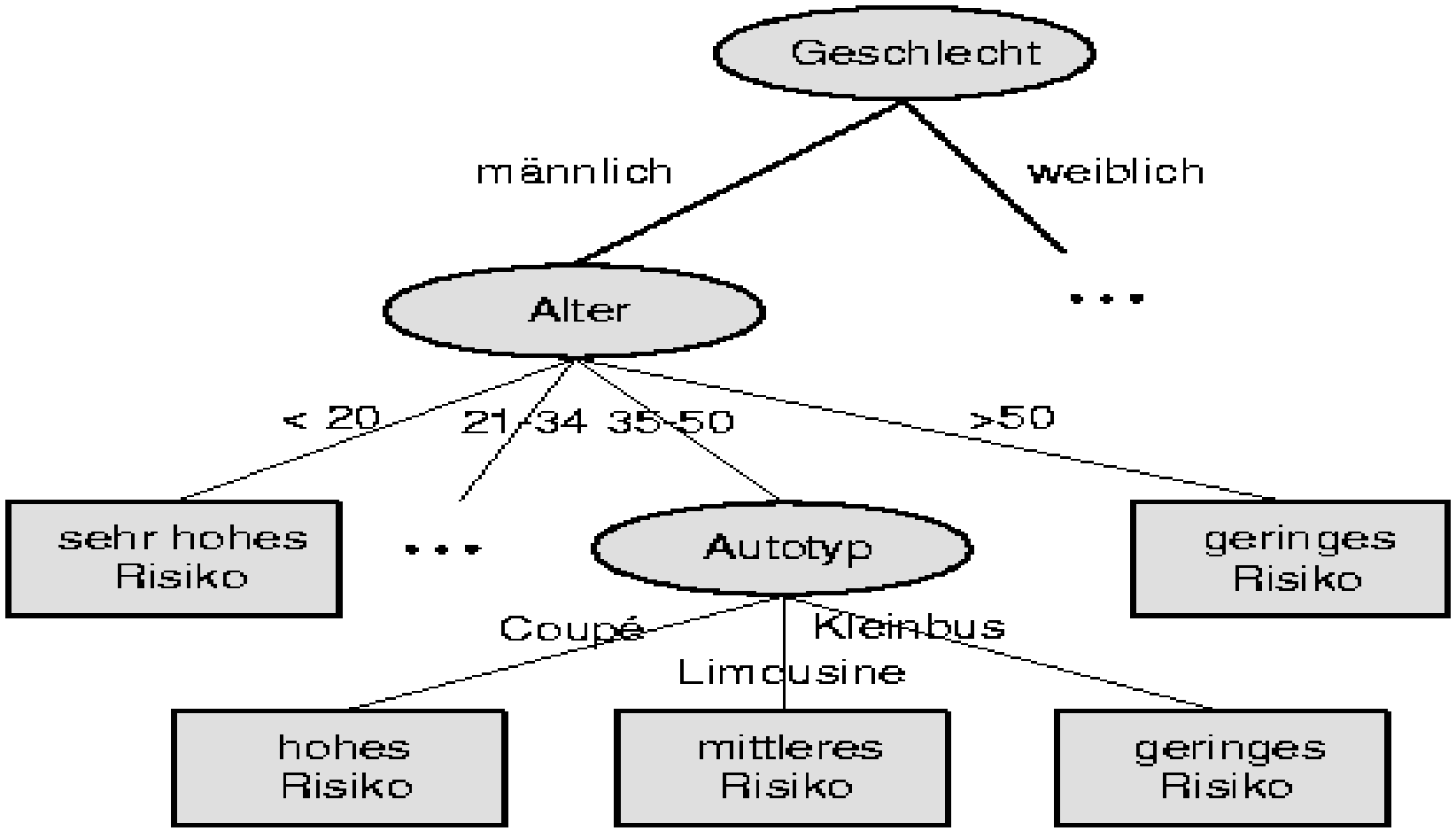
Data Mining

- **große Datenmengen** nach (bisher unbekannten) Zusammenhängen zu **durchsuchen**.

- Man unterscheidet zwei Zielsetzungen bei der Auswertung der Suche:

  - **Klassifikation** von Objekten,
  - Finden von **Assoziativregeln**

Klassifikation für Haftpflicht-Risikoabschätzung



Klassifikation von Objekten

- **Klassifikation** von Objekten(z.B: Menschen, Aktienkursen, ...) um, **Vorhersagen über das zukünftige Verhalten auf Basis bekannter Attributwerte** zu machen.

- Für die Risikoabschätzung könnte man vermuten, daß Männer zwischen 35 und 50 Jahren, die ein Coupé fahren, in eine hohe Risikogruppe gehören. Diese Klassifikation wird dann anhand einer repräsentativen Datenmenge verifiziert. Die Wahl der Attribute für die Klassifikation erfolgt (benutzergesteuert oder auch automatisch) durch "Ausprobieren".

Suche nach Assoziativregeln

- Um **Zusammenhänge bestimmter Objekte durch Implikationsregeln(Assoziativregeln) auszudrücken**, die vom Benutzer vorgeschlagen oder vom System generiert werden.

- Zum Beispiel könnte eine Regel beim Kaufverhalten von Kunden folgende (informelle) Struktur haben:

  - **Wenn** jemand einen PC kauft
    dann kauft er auch einen Drucker.

Suche nach Assoziativregeln

- Bei der Verifizierung solcher Regeln wird keine 100 %-ige Einhaltung erwartet.
  Stattdessen geht es um zwei Kenngrößen:
  - Confidence
  - Support

Suche nach Assoziativregeln

- **Confidence:**
  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 *Confidence* von 80% sagt aus, dass vier Fünftel der Leute, die einen PC gekauft haben, auch einen Drucker dazu genommen haben.

- **Support:**
  Dieser Wert legt fest, **wieviel Datensätze überhaupt gefunden wurden**, um die Gültigkeit der Regel zu verifizieren.
  Bei einem Support von 1% wäre also jeder Hunderste Verkauf ein PC zusammen mit einem Drucker.