

Data Warehouses und Data Mining

Themen der Data Warehousing (I)

- Einführung: Datenintegration, OLAP vs. OLTP
- Gründe für die Benutzung einer Data Warehouse (DWH):
 - - Kosten des Zugriffs zur Information
 - - Vereinheitlichung der Information
 - - Geschichtliche Information
- Drei komplementäre Trends:
 - - Data Warehousing
 - - Data Mining
 - - OLAP: On-line Analytical Processing
- Data Warehousing Merkmale:
 - - Multidimensionales Datenmodell
 - - Implementierungen: MOLAP und ROLAP
 - - Dimensionshierarchien

Themen der Data Warehousing (II)

- Merkmale zu OLAP Abfragen:
 - Star Schema
 - Aggregatfunktionen, wie z.B. Aufsummierung
 - Roll-up, drill-down, pivoting, slicing, dicing
 - OLAP Abfragen mit SQL: CUBE, Windowing
- Indizes und Views
 - Bitmap index
 - View materialization
- Interaktive Abfragen
 - Top N
 - Online Aggregation

Informationsgenerierung und -Speicherung

DM outline

- Häufig vorkommende Objektmengen
- Assoziationsregeln
- Klassifikations- und Regressionsregeln
- KI: Neuronale Netze, SOMs

Wieso eine DWH? (Bill Inmon)

Um wie viel besser zu sein, müssen die Anwendungen betrachtet werden
Betrachten wir eine Firma, die Information braucht

Ohne DWH
Zeit: 1-2 Jahr
Kosten: 25,000 - 1,000,000

Mit DWH
Zeit: 2 Minuten – 2 Wochen
Kosten: 250 – 10,000

Der einzige Unterschied zwischen beide Szenarien ist die Data Warehouse.
Die Kosten sind in die Vorbereitung der DWH „versteckt“

Der Gewinn wird durch die Wiederverwendung der Berichte realisiert

Wenn Sie das verstehen, verstehen Sie grundsätzlich die Nützlichkeits einer DWH

Anwendungen **Berichte**

Informationsgenerierung und -Speicherung

Aber es gibt noch weitere Gründe für eine DWH

- Betrachten wir die Anwendungen, die eine Firma über Zeit kreiert oder gekauft hat:

Für die gesamte Firma:
Was ist ein Kunde?
Jede Anwendung hat ihre eigene Interpretation von was ein Kunde ist. Es gibt kein einheitliches Konzept vom Kunde

- Daten werden aus der einzelnen Anwendungen geholt und in der Data Warehouse integriert

- Mit der Integration werden Daten vereinheitlicht und stellen dann eine einheitliche Sicht über dem Begriff „Kunden“

Für das erste Mal hat die Firma eine einheitliche Sicht über ihre eigene Information!

Aber es gibt noch ein wichtiger Grund, wieso eine Firma eine Data Warehouse braucht ...

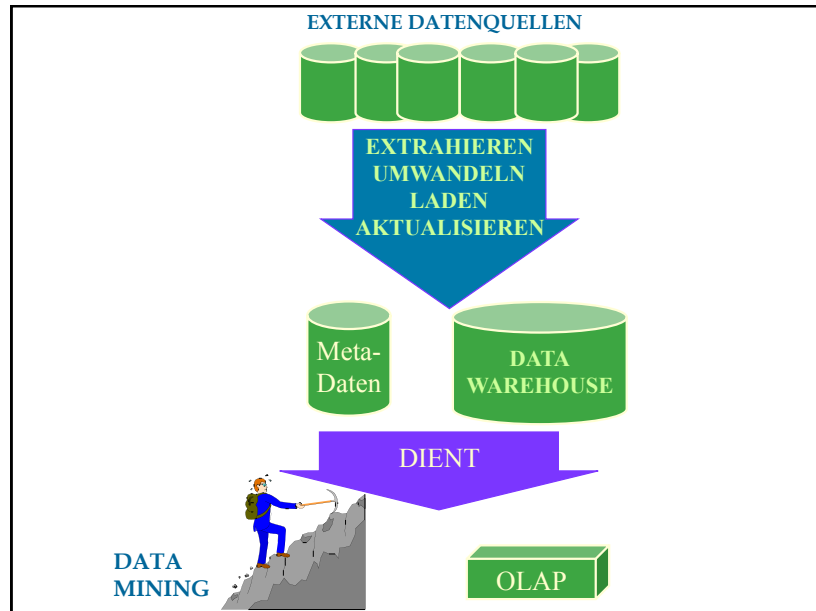
14 Tage
Nur aktuelle Daten
1 Woche
2 Tage

5 – 10 Jahre

Die Data Warehouse ist ein sehr guter Ort, um sie zu behalten

Anwendungen haben eine sehr begrenzte Menge von archivierten Daten

Informationsgenerierung und -Speicherung





OLTP	OLAP
Transaktionsorientiert	Analysisorientiert
Anwendungsorientiert	Themenorientiert
Abfrageresultate detailliert	Abfrageresultate zusammengefasst
Zeitlich genau	Zeitlich ungenau (snapshots)
Für die Administration	Für Managers
Wird ständig modifiziert (z.B. Jede Millisekunde)	Wird nicht oder selten (im Vergleich mit operativen Datenbanken) modifiziert (z.B. jeden Tag)
Läuft repetitiv	Läuft heuristisch
Untersuchte Prozesse sind a priori verstanden	Untersuchte Prozesse sind a priori nicht verstanden
Performanz-sensitiv	Nicht Performanz-sensitiv
Zugriff Datensatz-orientiert	Zugriff Mengenorientiert

Informationsgenerierung und -Speicherung

OLAP-Abfragen, Pivot Table

	AG	ZH	Total
1995	63	81	144
1996	38	107	145
1997	75	35	110
Total	176	223	399

Pivot Table auf Excel

The screenshot shows an Excel spreadsheet with a PivotTable and a bar chart. The PivotTable is located in the top right corner of the spreadsheet, showing the following data:

	AG	ZH	Total
1995	63	81	144
1996	38	107	145
1997	75	35	110
Total	176	223	399

The bar chart is located in the bottom right corner of the spreadsheet, showing the total values for each year: 1995 (144), 1996 (145), and 1997 (110).

Übung 1 Besprechung

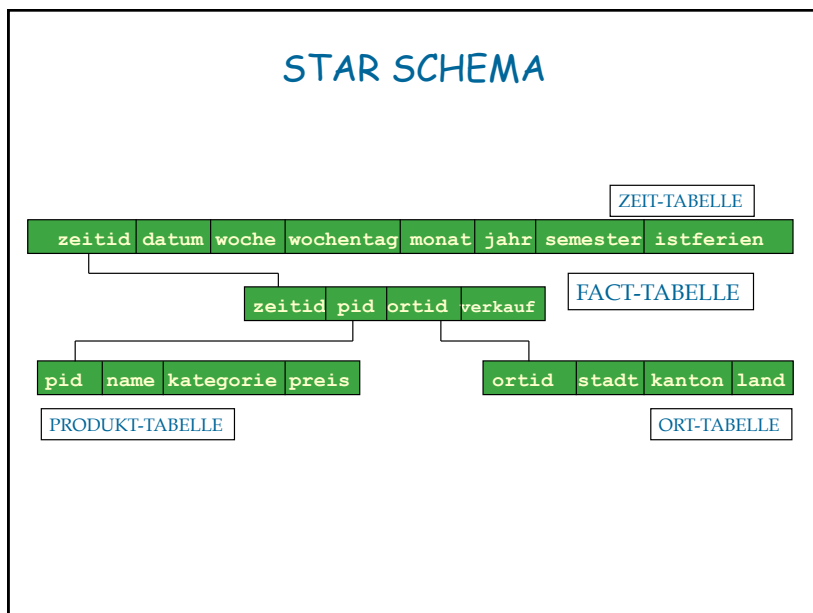
Informationsgenerierung und -Speicherung

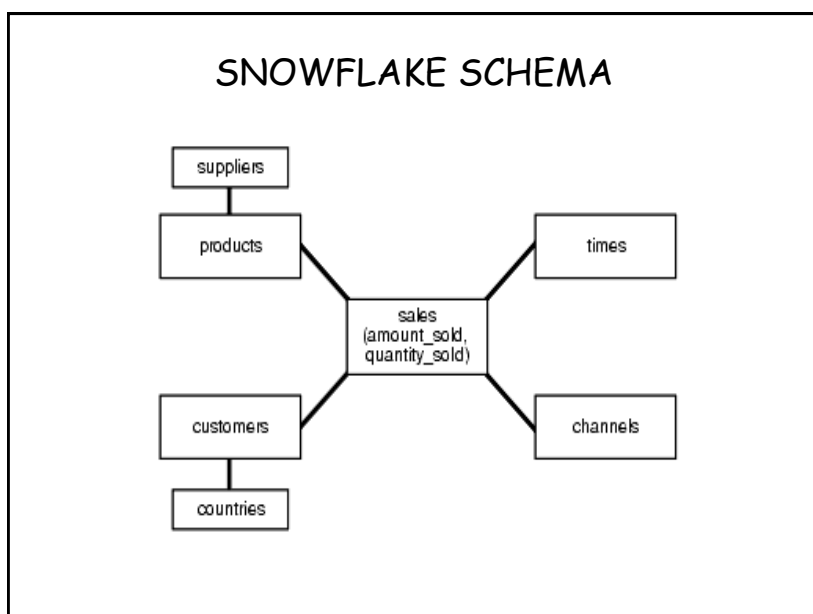
Multidimensionales Datenmodell

Sammlung von numerische **Messungen**, die von einer Menge **Dimensionen** abhängen.
 - Beispiel: Messung **Verkauf**, Dimensionen **Produkt** (key: **pid**), **Ort** (**ortid**), und **Zeit** (**zeitid**).

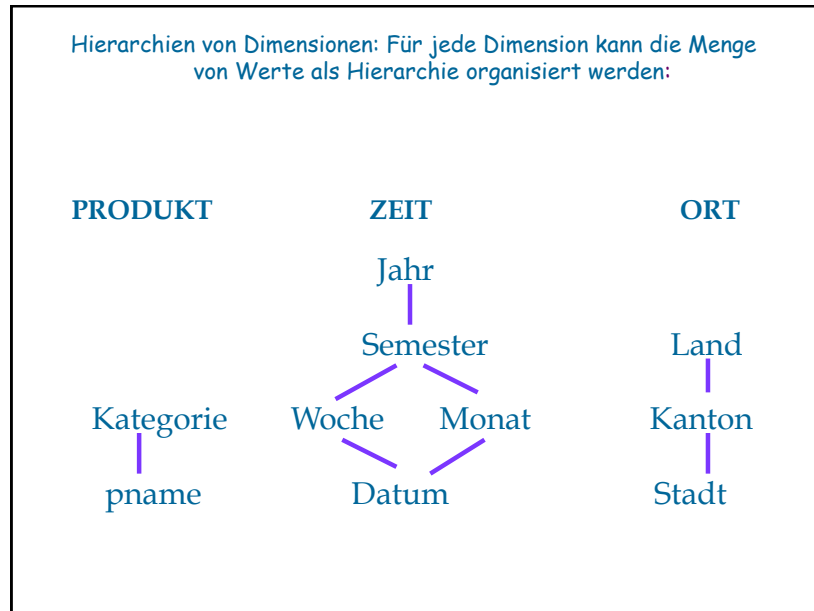
Hier sehen wir die Tranche [slice] **ortid=1**

pid	zeitid	ortid	verkauf
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30
12	2	1	20
12	3	1	50
13	1	1	8
13	2	1	10
13	3	1	10
11	1	2	35





Informationsgenerierung und -Speicherung



Übung 2

```
SELECT SUM(v.verkauf)
FROM Verkauf V, Zeit Z, Ort O
WHERE V.zeitid=Z.zeitid AND V.ortid=O.ortid
GROUP BY Z.jahr, O.kanton
---
SELECT SUM(v.verkauf)
FROM Verkauf V, Zeit Z
WHERE V.zeitid=Z.zeitid
GROUP BY Z.jahr
---
SELECT SUM(v.verkauf)
FROM Verkauf V, Ort O
WHERE V.ortid=O.ortid
GROUP BY O.kanton
---
SELECT SUM(v.verkauf)
FROM Verkauf V, Ort O
WHERE V.ortid=O.ortid
```

Informationsgenerierung und -Speicherung

```
SELECT channel_desc, TO_CHAR(SUM(amount_sold),
'9,999,999,999') SALES$,
RANK() OVER (ORDER BY SUM(amount_sold)) AS default_rank,
RANK() OVER (ORDER BY SUM(amount_sold) DESC NULLS LAST)
AS custom_rank
FROM sales, products, customers, times, channels, countries
WHERE sales.prod_id=products.prod_id AND
sales.cust_id=customers.cust_id AND
sales.time_id=times.time_id AND
sales.channel_id=channels.channel_id AND
times.calendar_month_desc IN ('2000-09', '2000-10') AND
country_iso_code='US'
GROUP BY channel_desc;
```

CHANNEL_DESC	SALES\$	DEFAULT_RANK	CUSTOM_RANK
Direct Sales	2,443,392	3	1
Partners	1,365,963	2	2
Internet	467,478	1	3

View

```
CREATE VIEW
LokalVerkauf(kategorie,verkauf, kanton)
AS SELECT P.kategorie, V.verkauf, O.kanton
FROM Produkte P, Verkauf V, Ort O
WHERE V.pid=P.pid AND V.ortid=O.ortid
```

Materialized View

```
CREATE MATERIALIZED VIEW
LokalVerkauf(kategorie,verkauf, kanton)
AS SELECT P.kategorie, V.verkauf, O.kanton
FROM Produkte P, Verkauf V, Ort O
WHERE V.pid=P.pid AND V.ortid=O.ortid
```

Informationsgenerierung und -Speicherung

Index auf materialisierte View

```
CREATE INDEX ON RegionalVerkauf  
(kategorie, verkauf, kanton)
```

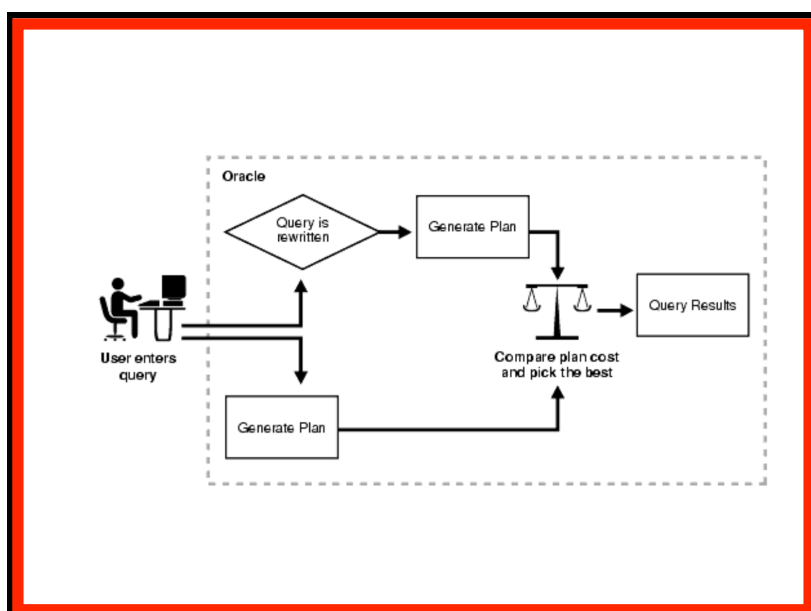
```
SELECT R.kanton,  
       SUM(R.verkauf)  
FROM Regionalverkauf R  
WHERE R.kategorie="Laptop"  
GROUP BY R.kanton
```

Index auf vorgerechnete View ist gut!

```
SELECT R.kanton,  
       SUM(R.verkauf)  
FROM Regionalverkauf R  
WHERE R.kanton="Aargau"  
GROUP BY R.kategorie
```

Index ist weniger nützlich (alle Blätter müssen durchgescannt werden)

```
CREATE MATERIALIZED VIEW cust_sales_mv  
PCTFREE 0 TABLESPACE demo  
STORAGE (INITIAL 16k NEXT 16k PCTINCREASE 0)  
PARALLEL  
BUILD IMMEDIATE  
REFRESH COMPLETE  
ENABLE QUERY REWRITE AS  
SELECT c.cust_last_name, SUM(amount_sold) AS sum_amount_sold  
FROM customers c, sales s WHERE s.cust_id = c.cust_id  
GROUP BY c.cust_last_name;
```



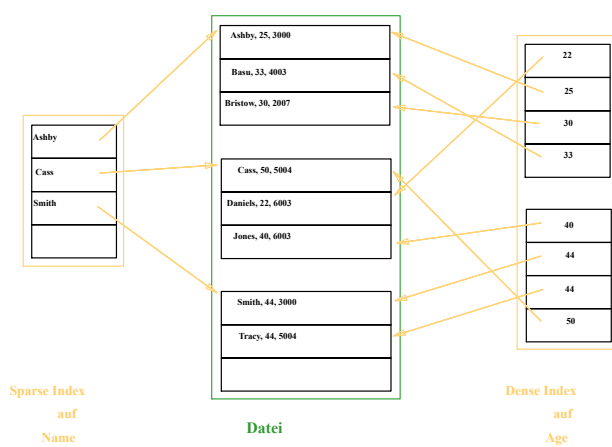
Top N Abfragen

```
SELECT P.p_id, P.pname, S.verkauf
FROM verkauf S, Products P
WHERE S.p_id=P.p_id AND S.ort_id=1 AND S.zeit_id=3
ORDER BY S.verkauf DESC
OPTIMIZE FOR 10 ROWS
```

```
SELECT P.p_id, P.pname, S.verkauf
FROM verkauf S, Products P
WHERE S.p_id=P.p_id AND S.ort_id=1 AND S.zeit_id=3
AND S.verkauf > c
ORDER BY S.verkauf DESC
```

- OPTIMIZE FOR Klausel ist nicht in SQL:1999!
- Cut-off Wert c ist vom Optimizer gewählt

Beispiele von Indizes



Bitmap Index

Bit-Vektoren

1 bit für jeden mögliche Wert
 Viele Abfragen können mit Operationen auf bit-Vektoren direkt beantwortet werden!

Geschlecht	Kunde	Name	gesch.	Bewertung	Bit-Vektor
10	112	Joe	M	3	00100
10	115	Ram	M	5	00001
01	119	Sue	F	5	00001
10	112	Woo	M	4	00010
