13 Data Warehouses in a nutshell

13.1 Introduction OLTP vs. OLAP
13.2 DWH methodology
13.3 Stars and Stripes
13.4 OLAP operators: Roll up and Drill down, SQL operators ROLLUP and CUBE
13.5 ROLAP and MOLAP ... and more

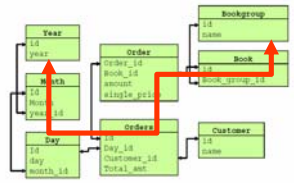
Kemper / Eickler: chap.4.17.2,
Melton: chap. 12
some slides inspired by U. Leser

Example

```
SELECT   Y.year, BG.name, count(O.id)
FROM     year Y, month M, day D, order O, orders OS,
         book B, bookgroup BG
WHERE    M.year = Y.id and
         M.id = D.month and
         OS.day_id = D.id and
         OS.id = O.order_id and
         B.id = O.book_id and
         B.book_group_id = BG.id and
         D.day < 24 and M.
...
```

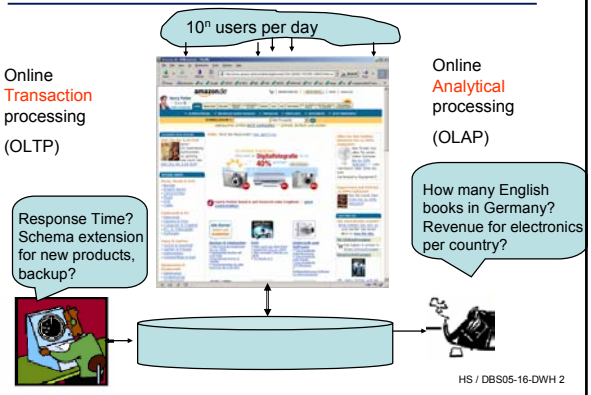
6 Joins

• Year:	10 Records
• Month:	120 Records
• Day:	3650 Records
• Orders:	36.000.000
• Order:	72.000.000
• Books:	200.000
• Bookgroups:	100



Huge temporary tables, difficult to optimize

13.1 Introduction OLTP versus OLAP

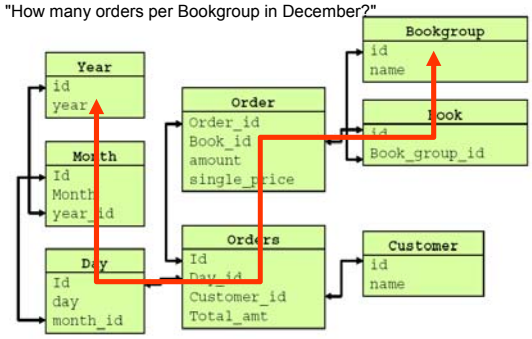


OLTP vs OLAP

- OLTP
  - huge number of records for transactions
  - typically more than one DB --> amazon.de, .fr, .com
  - slightly (?) different schemas
    Amazon US versus Amazon D ?
  - must be operational ~ 99.999 % of the year
  - no long running queries
- OLAP
  - very complicated ad hoc queries
  - data have to be homogenized
  - ad hoc queries, but structurally similar (aggregation)

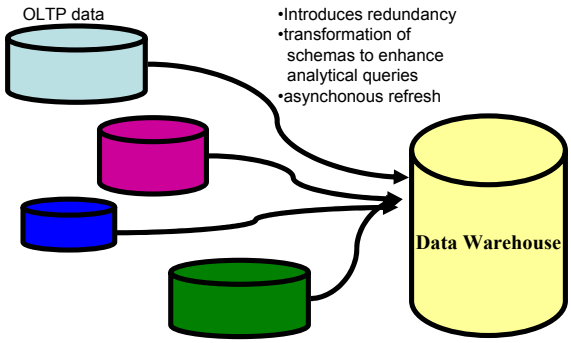
The DB and a query

"How many orders per Bookgroup in December?"



example from U.Leser

DWH as a centralized data resource



ETL : Extract – Transform - Load

DWH / OLAP : Tools for Decision support

- DWH exist since the 70ies
  - Decision support systems
  - Information management Systems
- No general methodology
- Not very successful, why?

- Infrastructure missing
  - networks
  - high performance DBS
  - high performance computer systems
    cycles, main memory, disks
- Data resources

Large databases

SBC Communications (US Telecommunication)
- 57.000.000 customers (phone, DSL)
- 360 Terabyte
- 12.000 tables
- 300.000 logins / day

Deutsche Telecom
- 100 Terabyte
- IBM, Oracle

An ERP application:

```
SQL> select count(*) from dba_tables
  2    where owner ='SAPR3';

  COUNT(*)
  ----------
  25885
```


DWH versus Data mining

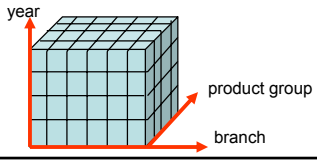
- DHW:
  - data have different "dimensions" : time, location, product group, ...
  - Aggregate dataset according to one or more dimensions
    - "total sales in january"
    - "total number or orders in region 'Berlin' for PG 'digital camera'
- Data Mining
  - analyze data statistically (more or less) and find out regular patterns
  - prediction of values by
    - association rules ("Customers who by beer by cigarettes in 70% of all transactions")
    - classification "Good customer / bad customer"

and more....

13.2 DWH methodology

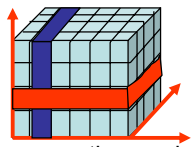
- Basic modeling
  - facts → Book ISBN 3456 sold 2005.6.10 by .de-branch to Customer 12789
  - Dimensions
    categories for describing facts: time, space, groups
    time, branch, product group
  - Classification hierarchies → day -> month -> quarter -> year

⇒ facts are points in a multidimensional space



DWH methodology

- OLAP data analysis uses operations on hypercubes
  - "slice and dice" : keep values in a dimension constant, group by values in a dimension
  - Change hierarchy level:
    e.g. day -> month

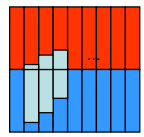


- Assumption: dimensions are orthogonal
  values in different dimensions are independent

DWH and data analysis

- n-dim Hypercube = n-dim contingency table

wealth values:		poor	rich
agegroup	10s	2507	3
	20s	11262	743
	30s	9468	3461
	40s	6738	3986
	50s	4110	2509
	60s	2245	809
	70s	668	147
	80s	115	16
	90s	42	13



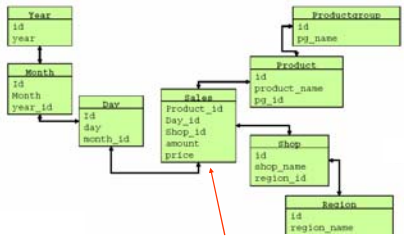
2-D-cube / histogram

table from Allan Moore, www.cs.cmu.edu/~awm

13.3 Stars and stripes

- n-dimensional cube in a database

(i)

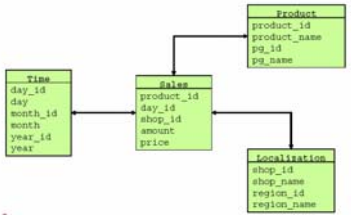


fact table

"Snowflake scheme"

Cubes in the Database: Stars

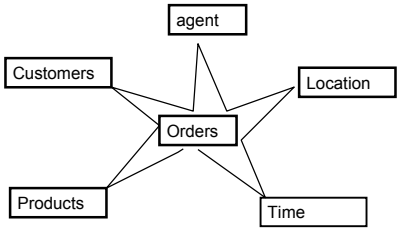
(ii) Star schema



One join of fact table with each dimension

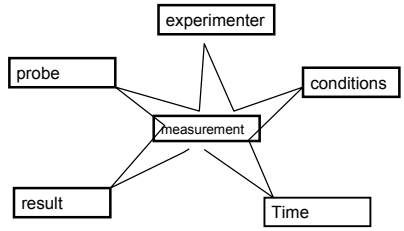
Methodology for DHW Building

- why "star"?



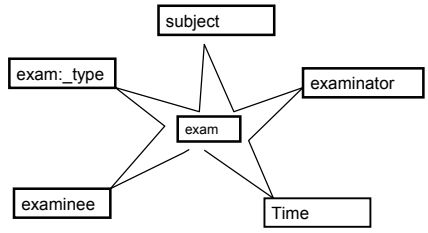
Star

Example: Data management for scientific experiments



Typical ER-pattern: n-ary relationship between dimension entities

Yet another star

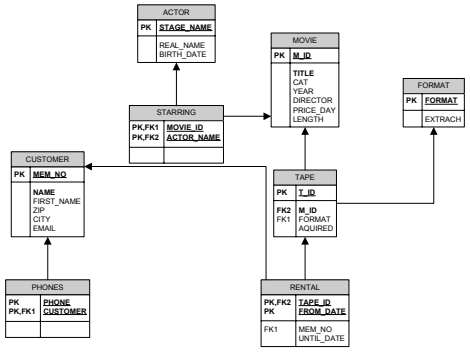


Typical situation: one "central" table with many "dimensions"
- 1:N relationship
- foreign key or attribute
- not normalized!

Normalization / denormalization

- OLTP
  - normalized tables make sense: no update inconsistencies
  - functional dependencies can be checked easily (check for duplicate key, very efficent operation)
- OLAP
  - normalized relations ⇒ many join when slicing
    ⇒ denormalized tables
- ETL tools transform operational OLTP DB into data warehouse
  - Must not necessarily be reversible:
    - drop unimportant attributes
    - aggregate values
    - introduce classification hierarchie ("time" -> month ->quarter -> year, see below)

Movie DB once again: OLTP schema



13.4 OLAP operators

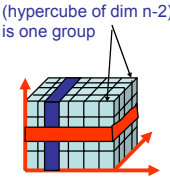
OLAP queries do not ask for individuals:

```
SELECT c.name, t.season, m.title
FROM customer c JOIN  time t on  (…)
     JOIN  movie_tape m on (…)
     JOIN rental on(...)

WHERE c.zipcode LIKE '14%'
```

Result set: customers who…

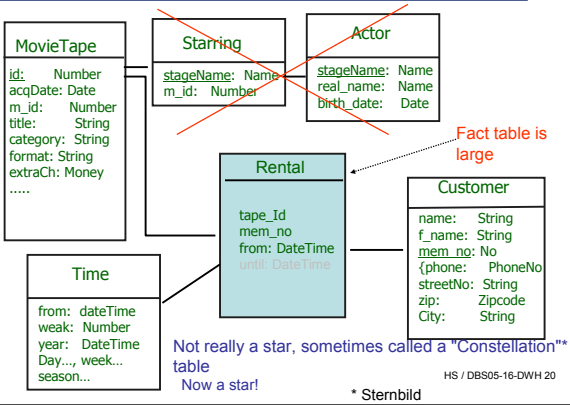
A row (hypercube of dim n-2) is one group



Typical: grouping and aggregation

```
SELECT t.season, m.category, count(*)
FROM customer c JOIN  time t on  (…)
     JOIN  movie_tape m on (…)
     JOIN rental on(...)
WHERE c.zipcode LIKE '14%'
GROUP by t.season, m.category
```


MovieDB: possible OLAP schema



Not really a star, sometimes called a "Constellation"* table

Now a star!

* Sternbild

Roll up and Drilldown

Drill down: "drill deeper into hypercube"

– more grouping attributes ⇨ less aggregation

⇨ lower dimensional HC

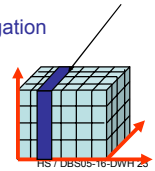
```
SELECT ... c.name,...
GROUP by t.season, m.category, c.name
```

Roll up

– less grouping attributes ⇨ more aggregation

```
SELECT t.season, count(*) FROM...
GROUP by t.season
```

… aggregates rental over each season



Star schema tables

Time table attributes depend on resolution of time dimension

Time

date	dd	mm	y	q	week	season
12.3.2005	12	3	2005	1	10	spring
17.1.2005	17	1	2005	1	3	winter
23.8.2004	.....					

Movie_tape

Rental

Mem_no	tid	from
531	1730	17.1.2005
278	311	23.8.2005
23.8.2004	.....	

customer

Star scheme ⇨ star join :

```
SELECT c.name, t.season, m.category
FROM customer c JOIN  time t on  (…)
          JOIN  movie_tape m on (…) JOIN rental on(...)
WHERE c.zipcode LIKE '14%'
 // zipcode supposed to be string, if not: translation
```


ROLLUP operator: motivation

```
SELECT m.category, t.season, count(*)
FROM customer c JOIN  time t on  (…)
          JOIN  movie_tape m on (…)
          JOIN rental r on (…)
WHERE c.zipcode LIKE '14%'
GROUP by m.category, t.season
```

Result table....

```
action spring  2388
action summer  2115
action fall    2917
action winter  3012
....
comedy summer  3527
....
thriller winter 5418
```

Missing: aggregate value for groups e.g.

```
action    –   10422
...
thriller  -   12317
```

ROLLUP operator

```
SELECT m.category, t.season, count(*)
FROM customer c JOIN  time t on  (…)
           JOIN  movie_tape m on (…)
           JOIN  rental r on (…)
WHERE c.zipcode LIKE '14%'
GROUP BY ROLLUP  (m.category, t.season)
```

Result:

```
action spring  2388
action summer  2115
action fall    2917
action winter  3012
action    _    10422
....
comedy summer  3527
....
thriller winter 5418
thriller  _    15317
_         _    93717
```

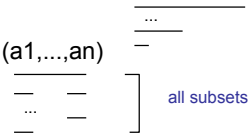
Super-aggregates

n Rollup attributes
⇨n+1 groupings

ROLLUP BY (a1,...,an)
⇨
GROUP BY a1,...an
GROUP By a1,..a(n-1)
...
GROUP BY a1
[GROUP BY _]
i.e. aggregate over all rows

The CUBE operator

- n attributes in GROUP BY (a1,...,an)
  ⇨ 2^n -1 nonempty subsets of {a1,..an}
- GROUP BY CUBE (a1,...,an) :
  UNION of 2^n -1 groupings together (UNION) with aggregation of all rows
- Obvious extension of ROLLUP a1,a2,....an
- GROUP BY CUBE (a1,...,an)



ROLLUP and GROUPING

Slight problem:

Suppose season has a NULL in some row

⇨ Result looks like

```
action spring  2388
action summer  2115
action fall    2917
action    _     0008
action    _    10422
....
comedy summer  3527
....
thriller winter 5418
thriller  _    15317
_         _    93717378
```

Confusing!
Want to distinguish NULL values in rows from superaggregates

CUBE example

```
SELECT m.category, t.season count(*)
FROM customer c JOIN  time t on  (…)
           JOIN  movie_tape m on (…)
           JOIN rental r on (...)
WHERE c.zipcode LIKE '14%'
GROUP BY CUBE  (m.category, t.season)
```

```
action spring  2388
action summer  2115
action fall    2917
action winter  3012
action    _    10422
....
comedy summer  3527
....
thriller winter 5418
thriller  _    15317
   _    spring 19317
...
   _    winter 27381
   _      _    93717
```

4 more rows for the of the second superaggregation over one attribute (season, 4 values)

GROUPING

```
SELECT m.category, t.season, count(*)AS ct,
       GROUPING (t.season)AS S, GROUPING (m.category) C
FROM customer c JOIN  time t on  (…)
           JOIN  movie_tape m on (…)
           JOIN rental r on (...)
WHERE c.zipcode LIKE '14%'
GROUP BY ROLLUP  (m.category, t.season)
```

```
category season  ct     S  C
action spring  2388     0  0
action summer  2115     0  0
action    _     3012     0  0
action    _    10422     1  0
....
comedy summer  3527     ...
....
thriller winter 5418
thriller  _    15317     1  0
_         _    93717378  1  1
```

not a superaggregate

13.5 ROLAP and...

- Reuse of temporary results -> materialization
  time transformed into attributes
  `d -> m -> season -> year *`
  Aggregation on `m`(onth) can be used for aggregating over `y`(ear) ⇨ reuse
- Materialized views
  - store aggregates which may be used frequently
  - combinatorial explosion prevents to store all of them
  - redundancy is NOT the problem in OLAP

* Classification is a partial order (lattice) in general, e.g. time: christmas_season = {Yes, NO} with semantics january..august = NO, ...

ROLAP

- Efficiency is a heavy problem in the DWH context
- CUBE over 3 or more attributes is heavy stuff

⇨ first solution: materialization
and specific index structures ("Bitmap index")

⇨ second solution: use completely different data structures instead of tables

Buzz words:

ROLAP : **R**elational **O**n**L**ine **A**nalytical **P**rocessing

MOLAP : **M**ultidimensional OLAP

Summary

- OLAP important for strategic planning
- Transforms operational data into "Data Warehouse"
- Analyzes data by aggregation and (simple) statistical operations
- OLAP = data analysis in a multidimensional space
- ROLLUP, CUBE etc part of SQL-3
- Implemented in most commercial system (Oracle, DB2)
- OLAP functions may are based upon RDBMS (ROLAP) or multidimensional data structures (MOLAP)