CSE 544
Principles of Database Management Systems

Magdalena Balazinska

Fall 2007

Lecture 16 - Data Warehousing

Class Projects

- Class projects are going very well!
- Project presentations: 15 minutes
  - On Wednesday in two weeks
  - There are 14 teams, so we will need to schedule extra time
  - We will give you the grading guidelines ahead of time
- Final reports
  - On Wednesday in two weeks (anytime)
  - See class website for details about content & presentation
  - Unfortunately, cannot accommodate extensions this year

Where We Are

- We covered all the fundamental topics

- We are now discussing various extra topics
  - Distributed databases (query optimization and transactions)
  - Replication
  - Physical database tuning (guest)
  - Data warehousing (today)
  - Model management (guest)
  - Stream processing
  - XML

References

- **Data Cube: A Relational Aggregation Operator Generalizing Group By, Cross-Tab, and Sub-Totals.**

  Jim Gray et. al. Data Mining and Knowledge Discovery 1, 29-53. 1997

- **Database management systems.**

  Ramakrishnan and Gehrke.

  Third Ed. **Chapter 25**

Why Data Warehouses?

- DBMSs designed to manage operational data
  - Goal: support every day activities
  - Online transaction processing (OLTP)
  - Ex: Tracking sales and inventory of each Wal-mart store
- Enterprises also need to analyze and explore their data
  - Goal: summarize and discover trends to support decision making
  - Online analytical processing (OLAP)
  - Ex: Analyzing sales of all Wal-mart stores by quarter and region
- To support OLAP consolidate all data into a warehouse
  - (note: this is an example of lazy master replication)

Outline

- Multidimensional data model and operations
- Data cube & rollup operators
- Data warehouse implementation issues
- Other extensions for data analysis

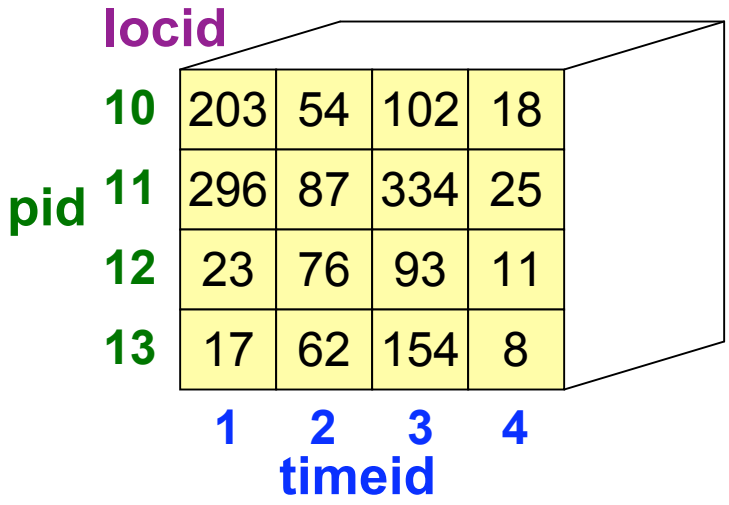
Data Analysis Cycle

- Formulate query that extracts data from the database
  - Typically ad-hoc complex query with group by and aggregate
- Visualize the data (e.g., spreadsheet)
  - Dataset is an N-dimensional space
- Analyze the data
  - Identify “interesting” subspace by aggregating other dimensions
  - Categorize the data an compare categories with each other
  - Roll-up and drill-down on the data

Multidimensional Data Model

- Focus of the analysis is a collection of measures
  - Example: Wal-mart sales

- Each measure depends on a set of dimensions
  - Example: product (pid), location (lid), and time of the sale (timeid)

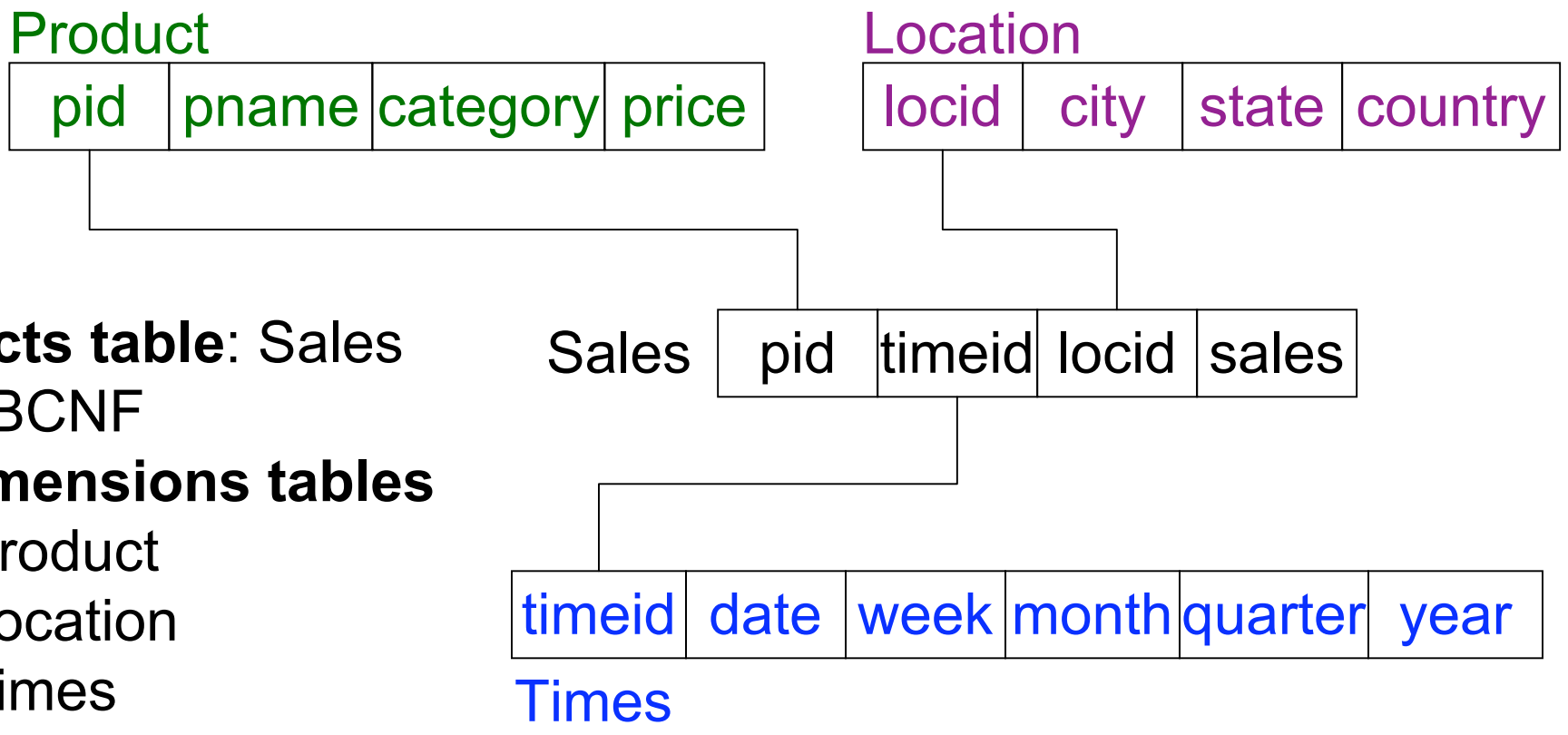


Slicing: equality selection on one or more dimensions

Dicing: range selection

Star Schema

Representing multidimensional data as relations (ROLAP)



Facts table: Sales
In BCNF

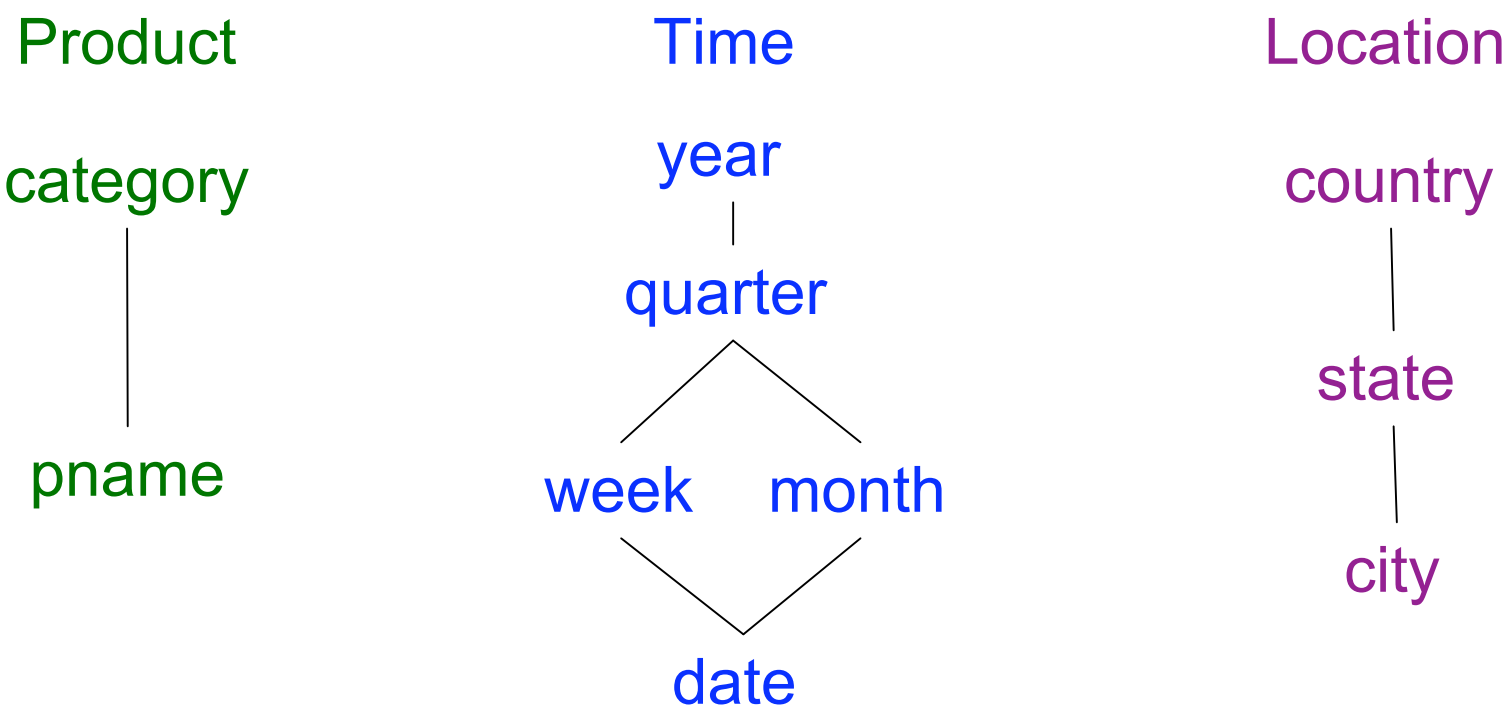
Dimensions tables

- Product
- Location
- Times

Not normalized

Dimension Hierarchies

Dimension values can form a hierarchy described by attributes



Desired Operations

- Histograms (agg. over computed categories)
  - Problem: awkward to express in SQL (paper p.34)
- Summarize at different levels: roll-up and drill-down
  - Ex: total sales by day, week, quarter, and year
- Pivoting
  - Ex: pivot on location and time
  - Result of pivot is a cross-tabulation
  - Column values become labels

	WI	CA	Total
2005	500	200	**700**
2006	150	850	**1000**
2007	250	400	**650**
Total	**900**	**1450**	**2350**

Challenge 1: Representation

- Problem: How to represent multi-level aggregation?
  - Ex: Table 3 in the paper need 2^N columns for N dimensions!
  - Ex: Table 4 has even more columns!
  - And that's without considering any hierarchy on the dimensions!
- Solution: special "all" value

T.year	L.state	SUM(S.sales)
2005	WI	**500**
2005	CA	**200**
2005	ALL	**700**
...	...	**...**
ALL	ALL	**2350**

Note: SQL-1999 standard uses NULL values instead of ALL

Challenge 2: Computing Agg.

- Need 2^N different SQL queries to compute all aggregates
  - Expressing roll-up and cross-tab queries is thus daunting
  - Cannot optimize all these independent queries

- Solution: CUBE and ROLLUP operators

Outline

- Multidimensional data model and operations
- Data cube & rollup operators
- Data warehouse implementation issues
- Other extensions for data analysis

Data Cube

- CUBE is the N-dimensional generalization of aggregate
- Cube in SQL-1999

```
SELECT T.year, L.state, SUM(S.sales)
FROM Sales S, Times T, Locations L
WHERE S.timeid=T.timeid and S.locid=L.locid
GROUP BY CUBE (T.year,L.state)
```

- Creating a data cube requires generating the power set of the aggregation columns

Rollup

- Rollup produces a subset of a cube

- Rollup in SQL-1999

```
SELECT T.year,T.quarter, SUM(S.sales)
FROM Sales S, Times T
WHERE S.timeid=T.timeid
GROUP BY ROLLUP (T.year,T.quarter)
```

- Will aggregate over each pair of (year,quarter), each year, and total, but will **not** aggregate over each quarter

Computing Cubes and Rollups

- Naive algorithm
  - For each new tuple, update each of 2^N matching aggregates
- More efficient algorithm
  - Use intermediate aggregates to compute others
  - Relatively easy for distributive and algebraic functions
- Updating a cube in response to updates is more challenging

Outline

- Multidimensional data model and operations
- Data cube & rollup operators
- Data warehouse implementation issues
- Other extensions for data analysis

Data Warehouse Properties

- Consolidated data from many sources
  - Must create a single unified schema
- Very large size: terabytes of data are common
- Complex read-only queries (no updates)
- Fast response time is important

Creating a Data Warehouse

- **Extract** data from distributed operational databases
- **Clean** to minimize errors and fill in missing information
- **Transform** to reconcile semantic mismatches
  - Performed by defining views over the data sources
- **Load** to materialize the above defined views
  - Build indexes and additional materialized views
- **Refresh** to propagate updates to warehouse periodically

Indexes

- Bitmap indexes: good for sparse attributes (few values)

M	F
0	1
1	0
1	0

custid	name	gender	rating
10	Alice	F	3
11	Bob	M	4
12	Chuck	M	1

1	2	3	4
0	0	1	0
0	0	0	1
1	0	0	0

- Join indexes: to speed-up specific join queries
  - Example: Join fact table F with dimension tables D1 and D2
  - Index contain triples of rids $<r_1,r_2,r>$ from D_1, D_2, and F that join
  - Alternatively, two indexes, each one with pairs $<v_1,r>$ or $<v_2,r>$ where v_1, v_2 are values of tuples from D_1, D_2 that join with r

Materialized Views

- How to choose views to materialize?
  - Physical database tuning: see last lecture
- How to keep view up-to-date?
  - Could recompute entire view for each update: expensive
  - Better approach: incremental view maintenance
  - Example: recompute only affected partition
  - How often to synchronize? Periodic updates (at night) are typical

Outline

- Multidimensional data model and operations
- Data cube & rollup operators
- Data warehouse implementation issues
- Other extensions for data analysis

Additional Extensions for Decision Support

- Window queries

```
SELECT L.state, T.month, AVG(S.sales) over W AS movavg
FROM Sales S, Times T, Locations L
WHERE S.timeid = T.timeid AND S.locid=L.locid
WINDOW W AS (PARTITION BY L.State
             ORDER BY T.month
             RANGE BETWEEN INTERVAL '1' MONTH PRECEDING
             AND INTERVAL '1' MONTH FOLLOWING)
```

- Top-k queries: optimize queries to return top k results
- Online aggregation: produce results incrementally