Data Mining MTAT.03.183
Online Analytical Processing and Data Warehouses
Jaak Vilo
2011 Fall

Acknowledgment

• This slide deck is a "mashup" of the following publicly available slide decks:
  - http://www.postech.ac.kr/~swhwang/grass/DataCube.ppt
  - http://www.cs.uiuc.edu/homes/hanj/bk2/03.ppt
  - Hector Garcia-Molina, Stanford University
  - Marlon Dumas, Univ. of Tartu,
  - Sulev Reisberg, Quretec & STACC
  - …

Outline

• The “data cube” abstraction
• Multidimensional data models
• Data warehouses

Sales data example

<table>
<thead>
<tr>
<th>ID</th>
<th>Region</th>
<th>Store</th>
<th>Category</th>
<th>Product</th>
<th>Date</th>
<th>Sale</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tallinn</td>
<td>Yemiesde TV</td>
<td>Samsung</td>
<td>13.10.2011</td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Tallinn</td>
<td>Mustika Radio</td>
<td>Sony</td>
<td>10.9.2011</td>
<td>1200</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Tallinn</td>
<td>Lounakekus TV</td>
<td>Samsung</td>
<td>11.11.2011</td>
<td>1150</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Tallinn</td>
<td>Lounakekus TV</td>
<td>Philips</td>
<td>11.11.2011</td>
<td>1500</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>tallinn</td>
<td>Mustika Radio</td>
<td>Samsung</td>
<td>12.9.2010</td>
<td>800</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Tallinn</td>
<td>Lounakekus TV</td>
<td>Sony</td>
<td>12.9.2011</td>
<td>1200</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Tallinn</td>
<td>Mustika Radio</td>
<td>Philips</td>
<td>11.11.2011</td>
<td>350</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Tallinn</td>
<td>Lounakekus TV</td>
<td>Sony</td>
<td>11.11.2011</td>
<td>1150</td>
<td></td>
</tr>
</tbody>
</table>

Excel pivot table

Example: Sales

<table>
<thead>
<tr>
<th>Market</th>
<th>Amount</th>
<th>Product</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atlanta</td>
<td>$8,000</td>
<td>Apples</td>
<td>$8,000</td>
</tr>
<tr>
<td>Chicago</td>
<td>$8,000</td>
<td>Cherries</td>
<td>$8,000</td>
</tr>
<tr>
<td>Denver</td>
<td>$8,000</td>
<td>Grapes</td>
<td>$8,000</td>
</tr>
<tr>
<td>Detroit</td>
<td>$8,000</td>
<td>Melons</td>
<td>$8,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Quarter</th>
<th>Total Qtr 1</th>
<th>Total Qtr 2</th>
<th>Total Qtr 3</th>
<th>Total Qtr 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Time</td>
<td>$32,000</td>
<td>$32,000</td>
<td>$32,000</td>
<td>$32,000</td>
</tr>
</tbody>
</table>

Table 2.1: Three different views of fruit sales: time, market, and product

Jaak Vilo and other authors
UT: Data Mining 2009
Multidimensional View of Sales

- Multidimensional analysis involves viewing data simultaneously categorized along potentially many dimensions

<table>
<thead>
<tr>
<th>Sales</th>
<th>Multidimensional view of sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qu 1</td>
<td>400.00</td>
</tr>
<tr>
<td>Qu 2</td>
<td>500.00</td>
</tr>
<tr>
<td>Total</td>
<td>900.00</td>
</tr>
</tbody>
</table>

Typical Data Analysis Process

- **Formulate** a query to extract relevant information
- **Extract** aggregated data from the database
- **Visualize** the result to look for patterns.
- **Analyze** the result and formulate new queries.
- **Online Analytical Processing (OLAP)** is about supporting such processes
- OLAP characteristics: No updates, lots of aggregation, need to visualize and to interact
- Let’s first talk about aggregation…

Relational Aggregation Operators

- SQL has several aggregate operators:
  - `SUM()`, `MIN()`, `MAX()`, `COUNT()`, `AVG()`
- The basic idea is:
  - Combine all values in a column into a single scalar value
- Syntax
  ```sql
  SELECT AVG(Temp) FROM Weather;
  ```

The Relational **GROUP BY** Operator

- **GROUP BY** allows aggregates over table sub-groups
  ```sql
  SELECT Time, Altitude, AVG(Temp) FROM Weather GROUP BY Time, Altitude;
  ```

Limitations of the GROUP BY

- Group-by is one-dimensional: one group per combination of the selected attribute values

<table>
<thead>
<tr>
<th>Model</th>
<th>Year</th>
<th>Color</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>Black</td>
<td>50</td>
</tr>
<tr>
<td>Chevy</td>
<td>1995</td>
<td>Black</td>
<td>85</td>
</tr>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>White</td>
<td>40</td>
</tr>
<tr>
<td>Chevy</td>
<td>1995</td>
<td>White</td>
<td>115</td>
</tr>
</tbody>
</table>

1. Calculate total sales per year
2. Compute total sales per year and per color
3. Calculate sales per year, per color and per model
Grouping with Sub-Totals (Pivot table)

- Sales by Model by Year by Color

<table>
<thead>
<tr>
<th>Model</th>
<th>Year</th>
<th>Color</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>Black</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>1995</td>
<td>Black</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>115</td>
</tr>
</tbody>
</table>

- Note that sub-totals by color are missing, if added it becomes a cross-tabulation

Grouping with sub-totals (cross-tab)

<table>
<thead>
<tr>
<th>Model</th>
<th>Year</th>
<th>Color</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>Black</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>1995</td>
<td>Black</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>115</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>Black</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>150</td>
</tr>
</tbody>
</table>

Grouping with Sub-Totals (Relational version)

<table>
<thead>
<tr>
<th>Sales by Model by Year by Color</th>
<th>Sales by Model by Year</th>
<th>Sales by Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy 1994 Black 50</td>
<td>Chevy 1994 Black 50</td>
<td></td>
</tr>
<tr>
<td>Chevy 1994 White 40</td>
<td>Chevy 1994 White 40</td>
<td></td>
</tr>
<tr>
<td>Chevy 1995 Black 85</td>
<td>Chevy 1995 Black 85</td>
<td></td>
</tr>
<tr>
<td>Chevy 1995 White 115</td>
<td>Chevy 1995 White 115</td>
<td></td>
</tr>
<tr>
<td>Chevy 1995 Black 115</td>
<td>Chevy 1995 Black 115</td>
<td></td>
</tr>
</tbody>
</table>

Sub-totals by color are still missing...

SQL Query

```
SELECT 'ALL', 'ALL', 'ALL', SUM(Sales) FROM Sales
WHERE Model = 'Chevy'
UNION
SELECT Model, 'ALL', 'ALL', SUM(Sales) FROM Sales
WHERE Model = 'Chevy'
GROUP BY Model
UNION
SELECT Model, Year, Color, SUM(Sales) FROM Sales
WHERE Model = 'Chevy'
GROUP BY Model, Year, Color
```

Adding the colors...

```
SELECT 'ALL', 'ALL', 'ALL', SUM(Sales) FROM Sales
WHERE Model = 'Chevy'
GROUP BY Model
```

CUBE and Roll Up Operators
The Cube

• An Example of 3D Data Cube

- By Make & Year
- By Make & Color
- By Color & Year
- By Year
- By Make
- By Color
- Sum

Cube: Each Attribute is a Dimension

• N-dimensional Aggregate (sum(), max(),...)
  – Fits relational model exactly:
    • \( a_1, a_2, ..., a_N, f() \)
  • Super-aggregate over N-1 Dimensional sub-cubes
    • \( \text{ALL}, a_2, ..., a_N, f() \)
    • \( a_1, \text{ALL}, a_3, ..., a_N, f() \)
    • \( \text{...} \)
    • \( a_1, a_2, ..., \text{ALL}, f() \)
  – This is the N-1 Dimensional cross-tab.
  • Super-aggregate over N-2 Dimensional sub-cubes

The Data Cube Concept

Sub-cube Derivation

• Dimension collapse, * denotes ALL

CUBE Operator
Possible syntax

• Proposed syntax example:
  – SELECT Model, Make, Year, SUM(Sales)
    FROM Sales
    WHERE Model IN {"Chevy", "Ford"}
    AND Year BETWEEN 1990 AND 1994
    GROUP BY CUBE Model, Make, Year
    HAVING SUM(Sales) > 0;
  – Note: GROUP BY operator repeats aggregate list
    • in select list
    • in group by list

ROLLUP Operator

• ROLLUP Operator: special case of CUBE Operator
  Return “Sales Roll Up by Store by Quarter” in 1994.: 
  SELECT Store, quarter, SUM(Sales)
  FROM Sales
  WHERE nation="Korea" AND Year=1994
  GROUP BY ROLLUP Store, Quarter(Date) AS quarter;
Summary

- Problems with GROUP BY
  - GROUP BY cannot directly construct
    - Pivot tables / roll-up reports
    - Cross-Tabs
- CUBE Operator
  - Generalizes GROUP BY and Roll-Up and Cross-Tabs!!

Cube Operator Example

Now let's have a look at one...

- NASA Workforce cubes
  http://nasapeople.nasa.gov/workforce

- Btell demo reports
  http://www.btell.de
  Follow the "demo" link and start a demo, then go to reports

OLAP Screen Example

Warehouse Architecture
Why a Warehouse?

- Two Approaches:
  - Query-Driven (Lazy)
  - Warehouse (Eager)

Multidimensional Data

- Sales volume as a function of product, month, and region

Dimensions: Product, Location, Time
Hierarchical summarization paths

Product
  - Category
  - Country
  - Quarter
  - City
  - Month
  - Week
  - Office
  - Day

Dimension Hierarchies

- Store
  - Region
  - City
  - Location

- Snowflake schema
  - Constellations

Terms

- Fact table
- Dimension tables
- Measures

Star

<table>
<thead>
<tr>
<th>Product</th>
<th>ProductId</th>
<th>Name</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>104</td>
<td>bolt</td>
<td>10</td>
</tr>
<tr>
<td>p2</td>
<td>105</td>
<td>nut</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Store</th>
<th>StoreId</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>nyc</td>
<td></td>
</tr>
<tr>
<td>c2</td>
<td>sfo</td>
<td></td>
</tr>
<tr>
<td>c3</td>
<td>la</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sale</th>
<th>OrderId</th>
<th>Date</th>
<th>CustId</th>
<th>ProdId</th>
<th>StoreId</th>
<th>Qty</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>o100</td>
<td>1/7/97</td>
<td>53</td>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>102</td>
<td>o102</td>
<td>2/7/97</td>
<td>53</td>
<td>p2</td>
<td>c1</td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>105</td>
<td>o105</td>
<td>3/6/97</td>
<td>111</td>
<td>p1</td>
<td>c3</td>
<td>5</td>
<td>50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Customer</th>
<th>CustId</th>
<th>Name</th>
<th>Address</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>81</td>
<td>jen</td>
<td>tom</td>
<td>10 man</td>
<td>sfo</td>
</tr>
<tr>
<td>111</td>
<td>sally</td>
<td>bob</td>
<td>50 willow</td>
<td>la</td>
</tr>
</tbody>
</table>

Star Schema
Hector Garcia Molina: Data Warehousing and OLAP

Cube

Fact table view:

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>p1</td>
<td>c3</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>c3</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Multi-dimensional cube:

dimensions = 2

3-D Cube

Fact table view:

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>c1</td>
<td>1</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>c3</td>
<td>1</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>c2</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>c1</td>
<td>2</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>c1</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>c2</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

Multi-dimensional cube:

dimensions = 3

Star Schema

Sales Fact Table

- time_key
- item_key
- branch_key
- location_key
- units_sold
- dollars_sold
- avg_sales

Measures

- time
- item
- branch
- location

Snowflake Schema

Sales Fact Table

- time_key
- item_key
- branch_key
- location_key
- units_sold
- dollars_sold
- avg_sales

Measures

- time
- item
- branch
- location

OLTP vs. OLAP

- OLTP – Online Transaction Processing
  - Traditional database technology
  - Many small transactions (point queries: UPDATE or INSERT)
  - Avoid redundancy, normalize schemas
  - Access to consistent, up-to-date database
- OLTP Examples:
  - Flight reservation
  - Banking and financial transactions
  - Order Management, Procurement, ...
- Extremely fast response times...

Carsten Binnig, ETH Zürich

OLTP vs. OLAP

- OLAP – Online Analytical Processing
  - Big aggregate queries, no Updates
  - Redundancy a necessity (Materialized Views, special-purpose indexes, de-normalized schemas)
  - Periodic refresh of data (daily or weekly)
- OLAP Examples
  - Decision support (sales per employee)
  - Marketing (purchases per customer)
  - Biomedical databases
- Goal: Response Time of seconds / few minutes

Carsten Binnig, ETH Zürich
### OLTP vs. OLAP (Water and Oil)
- **Lock Conflicts:** OLAP blocks OLTP
- **Database design:**
  - OLTP normalized, OLAP de-normalized
- **Tuning, Optimization**
  - OLTP: inter-query parallelism, heuristic optimization
  - OLAP: intra-query parallelism, full-fledged optimization
- **Freshness of Data:**
  - OLTP: serializability
  - OLAP: reproducibility
- **Integrity:**
  - OLTP: ACID
  - OLAP: Sampling, Confidence Intervals

### Solution: Data Warehouse
- **Special Sandbox for OLAP**
- **Data input using OLTP systems**
- **Data Warehouse aggregates and replicates data** (special schema)
- **New Data is periodically uploaded to Warehouse**

### DW Architecture

### What is data warehouse
- **Information system for reporting purposes**
- **The goal is to fulfill reporting needs which are unsatisfied in operational system**
  - It is easy to modify old and design new reports
  - No „write spec to software developer to get the report“ anymore
  - Reports can be filled with data quickly
  - No „start the report generation at night to prevent system load“ anymore
- **The data comes from operational system(s)**

### Goal of the work package
- **Work out the main concepts for building data warehouse for hospital IS**
  - What are the reporting needs?
  - What are the data cubes that cover most reporting needs for „universal“ hospital?
  - How to get the data into these cubes?

### Partners in this work package
- **Ida-Tallinn Keskkaitla (ITK)**
  - One of the biggest hospitals in Estonia
  - Huge amount of data in operational system (system called ESTER)
  - Has difficulties in generating reports on operational system
  - Interested in improving the report management
- **Quretec**
  - Provides data management software for different clients in Europe, especially in healthcare area
  - Interested in increasing the knowledge of data warehousing area
So far... (1)

- We have analyzed the data and data structures in operational system

So far... (2)

- We have designed the interface for getting the data from ESTER
- We have built 2 data cubes

So far... (3)

- We have designed 10 reports on the data

So far... (4)

- Showed that report generation time has reduced from tens of minutes to few seconds

<table>
<thead>
<tr>
<th>Selected period</th>
<th>Number of patients</th>
<th>Seconds for generating report in operational system</th>
<th>Seconds for generating the same report in data warehouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 day</td>
<td>138</td>
<td>149</td>
<td>1</td>
</tr>
<tr>
<td>1 month</td>
<td>2944</td>
<td>150</td>
<td>1</td>
</tr>
<tr>
<td>1 year</td>
<td>32386</td>
<td>164</td>
<td>1</td>
</tr>
</tbody>
</table>

So far... (5)

- We showed that data warehouse offers additional benefits:
  - Multiple output formats
  - Reports can be redesigned easily
  - New combined reports -> new value from the data

Implementing a Warehouse

- Monitoring: Sending data from sources
- Integrating: Loading, cleansing,...
- Processing: Query processing, indexing, ...
- Managing: Metadata, Design, ...
Monitoring

- Source Types: relational, flat file, IMS, VSAM, IDMS, WWW, news-wire, ...
- Incremental vs. Refresh

<table>
<thead>
<tr>
<th>customer id</th>
<th>name</th>
<th>address</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>53</td>
<td>joe</td>
<td>10 main</td>
<td>sfo</td>
</tr>
<tr>
<td>51</td>
<td>fred</td>
<td>12 main</td>
<td>sfo</td>
</tr>
<tr>
<td>111</td>
<td>sally</td>
<td>80 willow</td>
<td>la</td>
</tr>
</tbody>
</table>

Monitoring Techniques

- Periodic snapshots
- Database triggers
- Log shipping
- Data shipping (replication service)
- Transaction shipping
- Polling (queries to source)
- Screen scraping
- Application level monitoring

Advantages & Disadvantages!!

Monitoring Issues

- Frequency
  - periodic: daily, weekly, ...
  - triggered: on “big” change, lots of changes, ...
- Data transformation
  - convert data to uniform format
  - remove & add fields (e.g., add date to get history)
- Standards (e.g., ODBC)
- Gateways

Integration

- Data Cleaning
- Data Loading
- Derived Data

Data Cleaning

- Migration (e.g., yen ⇒ dollars)
- Scrubbing: use domain-specific knowledge (e.g., social security numbers)
- Fusion (e.g., mail list, customer merging)
- Auditing: discover rules & relationships (like data mining)

Loading Data

- Incremental vs. refresh
- Off-line vs. on-line
- Frequency of loading
  - At night, 1x a week/month, continuously
  - Parallel/Partitioned load
### Derived Data

- Derived Warehouse Data
  - indexes
  - aggregates
  - materialized views (next slide)
- When to update derived data?
- Incremental vs. refresh

### Materialized Views

Define new warehouse relations using SQL expressions

#### Table: Materialized Views

<table>
<thead>
<tr>
<th>sale</th>
<th>prodId</th>
<th>storeId</th>
<th>date</th>
<th>amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>e1</td>
<td>1</td>
<td>12</td>
<td>62</td>
</tr>
<tr>
<td>p2</td>
<td>e1</td>
<td>1</td>
<td>11</td>
<td>19</td>
</tr>
<tr>
<td>p1</td>
<td>e3</td>
<td>1</td>
<td>50</td>
<td>54</td>
</tr>
<tr>
<td>p2</td>
<td>e2</td>
<td>1</td>
<td>8</td>
<td>63</td>
</tr>
<tr>
<td>p1</td>
<td>e1</td>
<td>2</td>
<td>44</td>
<td>120</td>
</tr>
<tr>
<td>p1</td>
<td>e3</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>product</th>
<th>id</th>
<th>name</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>bolt</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>nut</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

### Processing

- ROLAP servers vs. MOLAP servers
- Index Structures
- What to Materialize?
- Algorithms

### ROLAP Server

- Relational OLAP Server

```sql
<table>
<thead>
<tr>
<th>product</th>
<th>id</th>
<th>name</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>bolt</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>nut</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>
```

### MOLAP Server

- Multi-Dimensional OLAP Server

- Index Structures
  - Traditional Access Methods
    - B-trees, hash tables, R-trees, grids, ...
  - Popular in Warehouses
    - inverted lists
    - bit map indexes
    - join indexes
    - text indexes
Inverted Lists

- Query:
  - Get people with age = 20 and name = “fred”
  - List for age = 20: r4, r18, r34, r35
  - List for name = “fred”: r18, r52
  - Answer is intersection: r18

Using Inverted Lists

Bit Maps

- Query:
  - Get people with age = 20 and name = “fred”
  - List for age = 20: r4, r18, r34, r35
  - List for name = “fred”: r18, r52
  - Answer is intersection: r18

Using Bit Maps

Join

- “Combine” SALE, PRODUCT relations
- In SQL: SELECT * FROM SALE, PRODUCT

Join Indexes

- Product
- Join index
What to Materialize?

- Store in warehouse results useful for common queries
- Example:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>c2</td>
<td>c3</td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>44</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td>p2</td>
<td>12</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>c2</td>
<td>c3</td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>56</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td>p2</td>
<td>11</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p1</td>
<td>110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td>19</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Materialization Factors

- Type/frequency of queries
- Query response time
- Storage cost
- Update cost

Cube Aggregates Lattice

Dimension Hierarchies

- Use greedy algorithm to decide what to materialize

Interesting Hierarchy

- Conceptual dimension table
Design
- What data is needed?
- Where does it come from?
- How to clean data?
- How to represent in warehouse (schema)?
- What to summarize?
- What to materialize?
- What to index?

Tools
- Development
  - design & edit: schemas, views, scripts, rules, queries, reports
- Planning & Analysis
  - what-if scenarios (schema changes, refresh rates), capacity planning
- Warehouse Management
  - performance monitoring, usage patterns, exception reporting
- System & Network Management
  - measure traffic (sources, warehouse, clients)
- Workflow Management
  - “reliable scripts” for cleaning & analyzing data

DW Products and Tools
- Oracle 11g, IBM DB2, Microsoft SQL Server, ...
  - All provide OLAP extensions
- SAP Business Information Warehouse
  - ERP vendors
- MicroStrategy, Cognos (now IBM)
  - Specialized vendors
  - Kind of Web-based EXCEL
- Niche Players (e.g., Btell)
  - Vertical application domain

MDX (Multi-Dimensional eXpressions)
- MDX is a Microsoft implementation of query language for OLAP

Example
SELECT
  {[Dim Date].[Time Year].[Time Year]} ON COLUMNS,
  {[Dim Location].[Region].[Region]} ON ROWS
FROM [Mini DW]
WHERE ([Measures].[Sales Amount])

<p>|</p>
<table>
<thead>
<tr>
<th>2007</th>
<th>2008</th>
</tr>
</thead>
<tbody>
<tr>
<td>Southeast</td>
<td>324975.18</td>
</tr>
<tr>
<td>West</td>
<td>351101.35</td>
</tr>
</tbody>
</table>

Chapter 2: Data Preprocessing
- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Discretization
- Three types of attributes:
  - Nominal — values from an unordered set, e.g., color, profession
  - Ordinal — values from an ordered set, e.g., military or academic rank
  - Continuous — real numbers, e.g., integer or real numbers
- Discretization:
  - Divide the range of a continuous attribute into intervals
  - Some classification algorithms only accept categorical attributes.
  - Reduce data size by discretization
  - Prepare for further analysis
Discretization and Concept Hierarchy

- **Discretization**
  - Reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals
  - Interval labels can then be used to replace actual data values
  - Supervised vs. unsupervised
  - Split (top-down) vs. merge (bottom-up)
  - Discretization can be performed recursively on an attribute

- **Concept hierarchy formation**
  - Recursively reduce the data by collecting and replacing low level concepts (such as numeric values for age) by higher level concepts (such as young, middle-aged, or senior)

Segmentation by Natural Partitioning

- A simply 3-4-5 rule can be used to segment numeric data into relatively uniform, “natural” intervals.
  - If an interval covers 3, 6, 7 or 9 distinct values at the most significant digit, partition the range into 3 equi-width intervals
  - If it covers 2, 4, or 8 distinct values at the most significant digit, partition the range into 4 intervals
  - If it covers 1, 5, or 10 distinct values at the most significant digit, partition the range into 5 intervals

Example of 3-4-5 Rule

<table>
<thead>
<tr>
<th>Range</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-$400,000 .. 0)</td>
<td>(-$400,000 .. -300,000)</td>
</tr>
<tr>
<td>(0 .. 1,000,000)</td>
<td>(0 .. 200,000)</td>
</tr>
<tr>
<td>(1,000,000 .. 2,000,000)</td>
<td>(1,000,000 .. 1,200,000)</td>
</tr>
<tr>
<td>(2,000,000 .. 5,000,000)</td>
<td>(2,000,000 .. 3,000,000)</td>
</tr>
</tbody>
</table>

Example

- MIN=-351,976.00
- MAX=4,700,896.50
- LOW = 5th percentile -159,876
- HIGH = 95th percentile 1,838,761
- msd = 1,000,000 (most significant digit)
- LOW = -1,000,000 (round down) HIGH = 2,000,000 (round up)

3 value ranges
1. (-1,000,000 .. 0]
2. (0 .. 1,000,000]
3. (1,000,000 .. 2,000,000]

Adjust with real MIN and MAX
1. (-400,000 .. 0]
2. (0 .. 1,000,000]
3. (1,000,000 .. 2,000,000]
4. (2,000,000 .. 5,000,000]

Concept Hierarchy Generation for Categorical Data

- Specification of a partial/total ordering of attributes explicitly at the schema level by users or experts
  - street < city < state < country
- Specification of a hierarchy for a set of values by explicit data grouping
  - {Urbana, Champaign, Chicago} < Illinois
- Specification of only a partial set of attributes
  - E.g., only street < city, not others
- Automatic generation of hierarchies (or attribute levels) by the analysis of the number of distinct values
  - E.g., for a set of attributes: {street, city, state, country}
**Automatic Concept Hierarchy Generation**

- Some hierarchies can be automatically generated based on the analysis of the number of distinct values per attribute in the data set.
- The attribute with the most distinct values is placed at the lowest level of the hierarchy.
- Exceptions, e.g., weekday, month, quarter, year.

(country: 15 distinct values)
(province_or_state: 365 distinct values)
(city: 3567 distinct values)
(street: 674,339 distinct values)

---

**Summary**

- OLAP and DW – a way to summarise data
- Prepare data for further data mining and visualisation
- Fact table, aggregation, queries\&indexes, ...

---

**Reference (highly recommended)**

- [http://citeseer.ist.psu.edu/old/392672.html](http://citeseer.ist.psu.edu/old/392672.html)
- Data Warehousing chapter of Jianwei Han’s textbook (chapter 3)

---

**Homework**

- Exercises 1 and 4 at: [http://www.systems.ethz.ch/education/courses/fs09/data-warehousing/ex2.pdf](http://www.systems.ethz.ch/education/courses/fs09/data-warehousing/ex2.pdf)
- Multidimensional data modeling exercise in course Wiki pages