Data Warehouses

Anne Denton

Department of Computer Science
North Dakota State University
Outline

1. Concepts of Data Warehouses
   - General properties of data warehouses

2. Multi-dimensional data model
   - Data cubes
   - Rollup and drilldown
   - Star schema
1. Concepts of Data Warehouses
   - General properties of data warehouses

2. Multi-dimensional data model
   - Data cubes
   - Rollup and drilldown
   - Star schema
Motivations for data warehouses

- Read-only transactions faster
  - Read-only transactions cannot block each other
- Integrated, subject-oriented collection of data
  - Items related to the same subject may be spread over multiple transactional databases
  - Data cleaning may be necessary to perform analytical processing
- Time variant, but non-volatile
  - Snapshots that don’t change but keep track of historical development
OLAP vs. OLTP

- **On-Line Transaction Processing (OLTP)**
  - Objective of conventional transactional relational databases
  - Managing of day-to-day operations, like purchasing, inventory, accounting, etc.

- **On-Line Analytical Processing (OLAP)**
  - Objective of data warehouse systems
  - Data analysis and decision-making

- **Distinct features**

<table>
<thead>
<tr>
<th></th>
<th>OLTP</th>
<th>OLAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>current and detailed relational design</td>
<td>historical, consolidated star-schema + subject knowledge evolutionary and integrated</td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>View</td>
<td>current and detailed updates</td>
<td>read-only but complex queries</td>
</tr>
<tr>
<td>Access</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Reasons for a separate data warehouse

- Improves performance for both systems
  - Read-only queries with table scope would slow down transactional system
  - Small updates would slow down read-only queries with table scope
  - Similar to creating snapshots and using those for read-only queries

- Data representation
  - Multi-dimensional views
  - Data consolidation from multiple transactional databases
  - Filling in missing data and reconciling formats
Question 1

Could there be a reason to have a data warehouse that uses the same type of RDBMS as the transactional databases of a business

1. Yes
2. No
1. Concepts of Data Warehouses
   - General properties of data warehouses

2. Multi-dimensional data model
   - Data cubes
   - Rollup and drilldown
   - Star schema
Data cubes

- Generalization of spreadsheets to arbitrarily many dimensions
  - Based on representation of one “fact”, for example, “sales”, ie. the dollar-value of combined sales
  - Each explanatory attribute considered one dimension
  - Example: “sales” depending on location, time, and model

- Dimensions greater than three?
  - Fundamentally, one “fact” can depend on arbitrarily many dimensions
  - For the above example, “sales” could also depend on customer type, like educational vs. commercial
  - Also called hyper-cube, considering that cubes conventionally have 3 dimensions
  - Dimensions don’t have to have same number of entries, and they do not have to have numerical data
  - In the following “cube” will be used regardless of dimension
Data cubes

- 2 or 3 dimensional cubes can still be graphically represented
- For higher dimensions, slices can be shown

![Cube Diagram](image)

- Model A
- Model B
- Model C
- Model D

<table>
<thead>
<tr>
<th></th>
<th>2018</th>
<th>2019</th>
<th>2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model A</td>
<td>2,300</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model B</td>
<td></td>
<td>1,850</td>
<td></td>
</tr>
<tr>
<td>Model C</td>
<td></td>
<td></td>
<td>2,655</td>
</tr>
<tr>
<td>Model D</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ND  MN
Question 2

How many dimensions would a data “cube” have that represents the following fact table of rainfall in various regions have?

<table>
<thead>
<tr>
<th>Month</th>
<th>Region</th>
<th>Rainfall in inches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>NW</td>
<td>5.4</td>
</tr>
<tr>
<td>Feb</td>
<td>NW</td>
<td>6.7</td>
</tr>
<tr>
<td>Jan</td>
<td>NE</td>
<td>12.1</td>
</tr>
</tbody>
</table>

1. 1
2. 2
3. 3
4. 4
Question 3

If the following data "cube" was represented as a relational table, how many columns would it have?

<table>
<thead>
<tr>
<th></th>
<th>NW</th>
<th>NE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>5.4</td>
<td>12.1</td>
</tr>
<tr>
<td>Feb</td>
<td>6.7</td>
<td>8.9</td>
</tr>
<tr>
<td>Mar</td>
<td>7.1</td>
<td>13.3</td>
</tr>
</tbody>
</table>

1. 2
2. 3
3. 4
4. 6
### Question 4

If the following data “cube” was represented as a relational table, how many rows would it have?

<table>
<thead>
<tr>
<th></th>
<th>NW</th>
<th>NE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>5.4</td>
<td>12.1</td>
</tr>
<tr>
<td>Feb</td>
<td>6.7</td>
<td>8.9</td>
</tr>
<tr>
<td>Mar</td>
<td>7.1</td>
<td>13.3</td>
</tr>
</tbody>
</table>

- 1️⃣ 2
- 2️⃣ 3
- 3️⃣ 4
- 4️⃣ 6
Table of Contents

1 Concepts of Data Warehouses
   - General properties of data warehouses

2 Multi-dimensional data model
   - Data cubes
   - Rollup and drilldown
   - Star schema
Rollup and drilldown

- Rollup along one dimension results in a cube of one fewer dimension
  - For example, the 3-dimensional “sales” cube can be rolled up along the model dimension to result in a 2-dimensional “cube” of sales values depending on location and time
  - Alternatively, it can be rolled up along the time dimension to result in a cube of sales depending on model and location
  - It can also be rolled-up along the model and time dimension to result in a 1-dimensional “cube” depending on location

- Rollup normally implemented using aggregate functions and group by statements
- Materialized views used for storing aggregates
- Which cubes should be materialized is question of physical design of the data warehouse
### Question 5

If the following data "cube" was rolled up along the region dimension, what would be the dimensionality of the result?

<table>
<thead>
<tr>
<th></th>
<th>NW</th>
<th>NE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>5.4</td>
<td>12.1</td>
</tr>
<tr>
<td>Feb</td>
<td>6.7</td>
<td>8.9</td>
</tr>
<tr>
<td>Mar</td>
<td>7.1</td>
<td>13.3</td>
</tr>
</tbody>
</table>

1. 1
2. 2
3. 3
4. 4
Question 6

If the following data “cube” was rolled up along the region dimension, how many elements would the result have?

<table>
<thead>
<tr>
<th></th>
<th>NW</th>
<th>NE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>5.4</td>
<td>12.1</td>
</tr>
<tr>
<td>Feb</td>
<td>6.7</td>
<td>8.9</td>
</tr>
<tr>
<td>Mar</td>
<td>7.1</td>
<td>13.3</td>
</tr>
</tbody>
</table>

1, 2, 3, 4
The alternatives of roll-up can be represented as a lattice.
Typical OLAP operations

**Roll up:** summarize data through dimension reduction

**Drill down:** reverse roll-up by going back to less aggregated data

**Slice:** select elements matching a slice of the cube (i.e. reduced by one dimension)

**Dice:** select elements matching cube of lower dimensionality

**Pivot:** reorient the cube, visualization, 3D to series of 2D planes (i.e. rotate)

**Drill across:** involving (across) more than one fact table

**Drill through:** through the bottom level of the cube to its back-end relational tables (using SQL)
Question 7 (Multiple answers can be correct)

Which of the following reduces the volume of results

1. Roll up
2. Drill down
3. Slice
4. Dice
5. Pivot
Question 8 (Multiple answers can be correct)

Which of the following involves multiple levels of aggregates

1. Roll up
2. Drill down
3. Slice
4. Dice
5. Pivot
Support for data cube functions in PostgreSQL

- Support for data cube functions used to be limited to dedicated data warehouse software.
- Initially even a physical cube organization was considered as storage model.
- Now, standard RDBMS offer adequate functionality.
- Even open source RDBMSs like PostgreSQL:

  https://www.postgresqltutorial.com/postgresql-cube/
Table of Contents

1. Concepts of Data Warehouses
   - General properties of data warehouses

2. Multi-dimensional data model
   - Data cubes
   - Rollup and drilldown
   - Star schema
Motivation for star schema

- Typically there is additional information to be represented for each dimension
  - Time may be broken down in years, months, and days
  - Location information may be complex
- This information may be needed for queries
Example of a star schema

- The dimensions may contain extensive information

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_key</td>
<td>day, day_of_week, month, quarter, year</td>
</tr>
<tr>
<td>location_key</td>
<td>street_address, city, state, country</td>
</tr>
<tr>
<td>model_key</td>
<td>color, finish, model_type</td>
</tr>
<tr>
<td>customer_key</td>
<td>business_type, classification</td>
</tr>
<tr>
<td>units_sold, dollars_sold</td>
<td>Sales</td>
</tr>
</tbody>
</table>

- Example of a star schema with dimensions:
  - Time (day, day_of_week, month, quarter, year)
  - Location (street_address, city, state, country)
  - Model (color, finish, model_type)
  - Customer (business_type, classification)
  - Units sold, dollars sold (Sales)
Concept hierarchies

- Dimensions can have different granularities
- Examples
  - A year has multiple months, which has multiple days
  - A region has multiple districts with multiple stores
- Implementing functions that recognize concept hierarchies adds implementation challenges
Further extensions

- **Snowflake schema**
  - Branches may have further branches

- **Constellation schema**
  - Branches may interconnect

- **Benefits of original star schema**
  - Corresponds to normalizing dimension tables
  - Not always desired or necessary in data warehouses because there are no updates
  - Data warehouses typically show some level of “denormalization"
  - Little space wasted because of small size of dimension tables
Question 9 (Multiple answers can be correct)

A star schema is often preferred over a snowflake schema because

1. It is properly normalized
2. Storage takes less space
3. It involves fewer join operations
Some functionality of data warehouses is now available as standard even in some open source databases like PostgreSQL.

Data warehouses are still useful for analytical processing:
- For performance reasons
- For logistic reasons (allow data aggregation, cleaning, storing of snapshots of data)