OLAP and Data Warehousing

Advanced Topics in Database Management (INFSCI 2711)
Distributed Databases (TELCOM 2326)

Some materials are from INFSCI2710’ a Database Management Systems, R. Ramakrishnan and J. Gehrke

Vladimir Zadorozhny, GIST, University of Pittsburgh

Introduction

- Increasingly, organizations are analyzing current and historical data to identify useful patterns and support business strategies.
- Emphasis is on complex, interactive, exploratory analysis of very large datasets created by integrating data from across all parts of an enterprise; data is fairly static.
  - Contrast such On-Line Analytic Processing (OLAP) with traditional On-line Transaction Processing (OLTP): mostly long queries, instead of short update Xacts.
Three Complementary Trends

- **Data Warehousing:** Consolidate data from many sources in one large repository.
  - Loading, periodic synchronization of replicas.
  - Semantic integration.
- **OLAP:**
  - Complex SQL queries and views.
  - Queries based on spreadsheet-style operations and “multidimensional” view of data.
  - Interactive and “online” queries.
- **Data Mining:** Exploratory search for interesting trends and anomalies (not considered in this class)

Data Warehousing

- Integrated data spanning long time periods, often augmented with summary information.
- Several gigabytes to terabytes common.
- Interactive response times expected for complex queries; ad-hoc updates uncommon.
Warehousing Issues

- **Semantic Integration**: When getting data from multiple sources, must eliminate mismatches, e.g., different currencies, schemas.
- **Heterogeneous Sources**: Must access data from a variety of source formats and repositories.
  - Replication capabilities can be exploited here.
- **Load, Refresh, Purge**: Must load data, periodically refresh it, and purge too-old data.
- **Metadata Management**: Must keep track of source, loading time, and other information for all data in the warehouse.

Multidimensional Data Model

- Collection of numeric measures, which depend on a set of dimensions.
  - E.g., measure Sales, dimensions Product (key: pid), Location (locid), and Time (timeid).

Slice locid=1 is shown:

<table>
<thead>
<tr>
<th>pid</th>
<th>timeid</th>
<th>locid</th>
<th>sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>3</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>2</td>
<td>35</td>
</tr>
</tbody>
</table>
**MOLAP vs ROLAP**

- Multidimensional data can be stored physically in a (disk-resident, persistent) array; called **MOLAP** systems. Alternatively, can store as a relation; called **ROLAP** systems.
- The main relation, which relates dimensions to a measure, is called the **fact table**. Each dimension can have additional attributes and an associated **dimension table**.
  - E.g., `Products(pid, pname, category, price)`
  - Fact tables are *much* larger than dimensional tables.

**Dimension Hierarchies**

- For each dimension, the set of values can be organized in a hierarchy:

```
  PRODUCT       TIME       LOCATION
  category      week       month
    name       date       state
      date     city
```
Star Schema Design

- Fact table is large, updates are frequent; dimension tables are small, updates are rare.
- This kind of schema is very common in OLAP applications, and is called a **star schema**; computing the join of all these relations is called a **star join**.

<table>
<thead>
<tr>
<th>TIMES</th>
<th>SALES (Fact table)</th>
<th>PRODUCTS</th>
<th>LOCATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeid</td>
<td>date</td>
<td>week</td>
<td>month</td>
</tr>
<tr>
<td>pid</td>
<td>timeid</td>
<td>locid</td>
<td>sales</td>
</tr>
<tr>
<td>pid</td>
<td>name</td>
<td>category</td>
<td>price</td>
</tr>
<tr>
<td>locid</td>
<td>city</td>
<td>state</td>
<td>country</td>
</tr>
</tbody>
</table>

OLAP Queries

- Influenced by SQL and by spreadsheets.
- A common operation is to **aggregate** a measure over one or more dimensions.
  - Find total sales.
  - Find total sales for each city, or for each state.
  - Find top five products ranked by total sales.
- **Roll-up**: Aggregating at different levels of a dimension hierarchy.
  - E.g., Given total sales by city, we can roll-up to get sales by state.
OLAP Queries

- **Drill-down**: The inverse of roll-up.
  - E.g., Given total sales by state, can drill-down to get total sales by city.
  - E.g., Can also drill-down on different dimension to get total sales by product for each state.
- **Pivoting**: Aggregation on selected dimensions.
  - E.g., Pivoting on Location and Time yields this **cross-tabulation**:

<table>
<thead>
<tr>
<th></th>
<th>WI</th>
<th>CA</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1995</td>
<td>63</td>
<td>81</td>
<td>144</td>
</tr>
<tr>
<td>1996</td>
<td>38</td>
<td>107</td>
<td>145</td>
</tr>
<tr>
<td>1997</td>
<td>75</td>
<td>35</td>
<td>110</td>
</tr>
<tr>
<td>Total</td>
<td>176</td>
<td>223</td>
<td>339</td>
</tr>
</tbody>
</table>

- **Slicing and Dicing**: Equality and range selections on one or more dimensions.

Using SQL for Pivoting

- The cross-tabulation obtained by pivoting can also be computed using a collection of SQL queries:

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

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

```
SELECT SUM(S.sales) 
FROM Sales S, Location L 
WHERE S.timeid=L.timeid 
GROUP BY L.state
```
**The CUBE Operator**

- Generalizing the previous example, if there are \( k \) dimensions, we have \( 2^k \) possible SQL GROUP BY queries that can be generated through pivoting on a subset of dimensions.
  - `CUBE pid, locid, timeid BY SUM Sales`
    - Equivalent to rolling up Sales on all eight subsets of the set \{pid, locid, timeid\}; each roll-up corresponds to an SQL query of the form:

```
SELECT SUM(S.sales)
FROM Sales S
GROUP BY grouping-list
```

Lots of work on optimizing the CUBE operator!

**Views and Decision Support**

- OLAP queries are typically aggregate queries.
  - Precomputation is essential for interactive response times.
  - The CUBE is in fact a collection of aggregate queries, and precomputation is especially important: lots of work on what is best to precompute given a limited amount of space to store precomputed results.

- Warehouses can be thought of as a collection of asynchronously replicated tables and periodically maintained views.
  - Has renewed interest in view maintenance!
**View Modification (Evaluate On Demand)**

CREATE VIEW `RegionalSales` (category, sales, state)
AS
SELECT P.category, S.sales, L.state
FROM Products P, Sales S, Locations L
WHERE P.pid=S.pid AND S.locid=L.locid

SELECT R.category, R.state, SUM(R.sales)
FROM `RegionalSales` AS R
GROUP BY R.category, R.state

SELECT R.category, R.state, SUM(R.sales)
FROM (SELECT P.category, S.sales, L.state
      FROM Products P, Sales S, Locations L
      WHERE P.pid=S.pid AND S.locid=L.locid) AS R
GROUP BY R.category, R.state

**View Materialization (Precomputation)**

- Suppose we precompute `RegionalSales` and store it.
- Then, previous query can be answered more efficiently (modified query will not be generated).
Issues in View Materialization

- What views should we materialize, and what indexes should we build on the precomputed results?
- Given a query and a set of materialized views, can we use the materialized views to answer the query?
- How frequently should we refresh materialized views to make them consistent with the underlying tables? (And how can we do this incrementally?)

Summary

- Decision support is an emerging, rapidly growing subarea of databases.
- Involves the creation of large, consolidated data repositories called data warehouses.
- Warehouses exploited using sophisticated analysis techniques: complex SQL queries and OLAP "multidimensional" queries (influenced by both SQL and spreadsheets).
- New techniques for database design, indexing, view maintenance, and interactive querying need to be supported.
- Commonly requires integrating DISTRIBUTED HETEROGENEOUS DATA