Introduction to Databases, Fall 2005

IT University of Copenhagen

Lecture 6, part 2: OLAP and data cubes

October 10, 2005

Lecturer: Rasmus Pagh

Today's lecture, part II

- Information integration.
- On-Line Analytical Processing (OLAP) vs On-Line Transaction Processing (OLTP).
- Data cubes and Relational OLAP.

Information integration

Information integration is the process of combining and using together information from several sources (databases).

Three main approaches to information integration:

- *Federated database systems* have 1-to-1 connections between all pairs of databases.
- *Data warehouses* merge data from all sources into a single, coherent database.
- *Mediators* look like data warehouses, but are just views (store no data).

All approaches have to deal with difficulties such as different representations of the same thing, or similar representations of different things, and differences in data types.

Advantages and disadvantages

- Federated database systems are appropriate for limited communication and interaction among databases. However, they require special software for each pair of communicating databases.
- Data warehouses are useful for analyzing data gathered from many sources. However, it might only be feasible to update the data periodically (e.g., once a day or once a week).
- Mediators have the same uses as data warehouses. The data is always up to date, but usually takes longer to access and query than in a data warehouse.

On-Line Analytical Processing

Fueled by advances in information integration, there is an increasing demand for *decision support* systems supporting complex queries on large data sets.

The desired mode of operation is that answers to queries come "on-line", i.e. almost immediately, hence the term:

- On-Line Analytical Processing (OLAP)

In contrast, in the classical use of databases for processing transactions, most updates and queries concern a small part of the database:

- On-Line Transaction Processing (OLTP)

Aggregates

OLAP queries are typically about *aggregates* such as sums and averages. SQL can be used to specify aggregate queries.

Some examples:

```
SELECT SUM(price) FROM Sales;

SELECT dealer, AVG(price) FROM Sales
GROUP BY dealer;

SELECT state, AVG(price)
FROM Sales, Dealers
WHERE dealer=name AND date>'2001-09-11'
GROUP BY state;
```

OLAP technology

OLAP systems come in two flavors:

- MOLAP - specialized software tailored especially for OLAP.
- ROLAP - a relational database with features to make OLAP queries efficient. (To be discussed next.) Usually done on a data warehouse.

Next: Data cubes and Relational OLAP

Facts and measures

Data for analysis can usually be though of as a collection of *facts* about events or objects of interest.

A Relational OLAP system has a *fact table* with a tuple for each fact.

Examples: Sales, customers, web site clicks.

A fact will typically have associated with it one or more *measures* (or *dependent attributes*) that can be aggregated.

Examples: Sales price, customer debit, time to next click.

Dimensions

Facts will typically also contain other information than measures, which may be used to select certain facts of interest.

Examples: ID of sales person, name of shop, state of shop,...

To limit redundancy, the fact table should not have any avoidable FDs, e.g.

$$\texttt{salespersonID} \quad \rightarrow \quad \texttt{shop} \quad \texttt{state}$$

When decomposing according to an FD, one gets a relation with the attributes mentioned in the FD. This is called a *dimension table* and referring attributes are called *dimension attributes*. `[Figure 20.13]`

Example: The "date" dimension might contain information about which week, month, quarter, and year a date is in.

Why "dimension"?

The term is related to a view of the fact table as points in a (multidimensional) cube.

Each axis of the cube corresponds to a dimension.

[Figure 20.12 shown on slide]

Typical OLAP queries will perform aggregations on *slices* of this cube, with certain (ranges of) values for each dimension attribute. [Figure 20.15]

Example:

```
SELECT SUM(price) FROM Sales
WHERE dealer='John Doe' AND date>='2004-01-01';
```

Dicing

Often one will be interested not only in a single aggregate, but a number of aggregates grouped according to some dimension.

Examples:

```
SELECT dealer, AVG(price) FROM Sales
GROUP BY dealer;
```

```
SELECT state, AVG(price)
FROM Sales, Dealers, Dates
WHERE dealer=name AND date=Dates.key AND year>2003
GROUP BY state;
```

The cube can be thought of as *diced* according to the granularity with which we look at the dimensions. [Figure 20.14 shown on slide]

Problem session (5 minutes)

Explain to each other the following terms (in the context of this lecture):

- Fact
- Measure
- Dimension
- Slicing
- Dicing

Identify any unclarities about the terms to be discussed in class.

Normalizing the dimension tables

For efficiency reasons, one sometimes chooses not to normalize the dimension tables (they typically use much less space than the fact table). This is known as a *star schema*.

If dimension tables are normalized, one obtains a *snowflake schema*.

Modern RDBMSs recognize star schemas and snowflake schemas, and use algorithms tailored to be efficient on such schemas when evaluating queries.

Using materialized views

To compute aggregates over large data sets efficiently, OLAP systems precompute certain aggregates which can be used to answer queries quickly.

Example: Consider the aggregate queries from before:

```
SELECT SUM(price) FROM Sales;

SELECT dealer, AVG(price) FROM Sales
GROUP BY dealer;
```

We don't have to go through all sales if we precomputed the number of sales and total sales price for each dealer.

Using materialized views 2

We specify what is to be precomputed through materialized views.

Example: If we create the following materialized view:

```
CREATE MATERIALIZED VIEW monthsales AS
SELECT month, year, SUM(price) FROM Sales, Dates
WHERE date=Dates.key
GROUP BY month, year;
```

. . . then subsequent queries for sales in quarters and years can be computed by just adding a few numbers in the materialized view.

Some DBMSs assist in choosing and using materialized views.

Most important points in this lecture

As a minimum, you should after this week:

- Know the meaning of some buzzwords: OLAP, information integration, data warehousing, multidimensional databases.
- Know how multidimensional databases can be organized and queried using relations (star schema, snowflake schema).

Next time

After the fall break we will study **relational algebra**, the theoretical basis of SQL queries.

- Relational algebra on sets.
- Relational algebra on bags.
- . . . and the relation to SQL queries.