Databasesystemer, forår 2006

IT Universitetet i København

Forelæsning 11: Data warehousing og OLAP

20. april 2006

Forelæser: Rasmus Pagh

Today's lecture

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

Then guest lecture by Tøger Nørgaard, SAS Institute.

Data warehouse

A *data warehouse* is an integrated collection of data used in support of decision-making processes. Typical characteristics:

- **Time-variant.** Records data over time (not just "current" data).
- **Integrated.** May contain data from many sources. (Ideally all data of an organisation.)
- **Non-updatable.** Just a snapshot of the operational data. Periodically updated.

Selecting a good architecture of a data warehouse is nontrivial, due to performance issues, and because of the effort that goes into *data reconciliation*.

Some experts propose an architecture in which a data warehouse consists of a number of *data marts*, which are data warehouses with a limited scope.

Data reconciliation

Typical operational data is:

- **Transient** – not historical
- **Restricted in scope** – not comprehensive
- **Sometimes poor quality** – inconsistencies and errors

After ETL (Extract-Transform-Load), data should be:

- **Detailed** – not summarized yet
- **Historical** – periodic
- **Comprehensive** – enterprise-wide perspective
- **Timely** – data should be current enough to assist decision-making
- **Quality controlled** – accurate with full integrity

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.

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

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

Example: In the previous queries, we don't have to go through all sales if we precomputed the number of sales and average sales price for each dealer.

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.

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*.

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

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 (to 3NF or 4NF), 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

Precomputation is essential to "on-line" answering of queries.

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).