Data Warehousing: Buzzwords, Definitions and Research Issues

Janet L. Wiener and Venky Harinarayan
Stanford University

A data warehouse is a repository of information integrated from multiple database or other information sources and available for complex querying and analysis. The topic of data warehousing encompasses architectures, algorithms, and tools for (1) designing the warehouse schema; (2) creating and maintaining the warehouse; and (3) querying the warehouse data. These 3 aspects are interrelated: the warehouse design may be optimized for efficient maintenance, for efficient querying and analysis (in both cases by adding extra data), or for the lowest storage cost. Supporting certain query operations may require maintaining extra information, e.g., historical data may be added to the source information so that warehouse queries can identify trends.

The first half of this tutorial will focus on warehouse creation and maintenance, while the second half will focus on querying and analysis. Design issues will be covered in both halves. In addition to identifying the open research problems in data warehousing, we will cover the current state-of-the-art in both research systems and in industry products (where information is available). Currently, IBM, Sybase, Prism, Redbrick, Bull, Arbor, Kenan, DBStar, and Pyramid all offer products that can be used to either maintain or organize and query a data warehouse.

Warehouse creation and maintenance: A data warehouse in effect maintains materialized views over the source data, and warehouse creation and maintenance shares many similaries with materialized view maintenance. However, key differences are that (1) the changes to the source data must be extracted; (2) data from multiple sources must be integrated, which may involve translation to a common (warehouse) data model, joins across data from multiple sources, and data scrubbing or cleaning; and (3) additional information may need to be added, such as historical information, annotations, and aggregates. Other issues includes expiration of data at the warehouse, crash recovery during maintenance, and whether the views can be self-maintained at the warehouse (possibly by altering the original warehouse design to store extra data) or require queries to other sources. Optimizations may be applied at all stages, e.g., extracted changes may be filtered by the view definitions before translation.

Warehouse querying and analysis: The primary application for most data warehouses is decision support. However, database systems and research have mainly focussed on OLTP (On-Line Transaction Processing) applications. Decision-Support queries, also referred to as OLAP (On-Line Analytical Processing) queries, have very different characteristics from OLTP queries: (1) OLAP queries are very complex, almost always involving aggregation; and (2) OLAP queries are mostly read-only. Even though the database is very large and the queries very complex, the interactive nature of OLAP means queries must complete very quickly. Thus query processing and optimization techniques are critical to the success of OLAP. We will cover some important new techniques to process complex queries. Most OLAP systems are based on the datacube data model. Datacubes, being derived data, can be virtual or materialized. When materialized they can also be stored in specialized "multi-dimensional" databases. We will discuss the issues involving the datacube data model in detail. Other issues we will cover are (1) common OLAP architectures and approaches; (2) OLAP enhancements to SQL like the datacube operator and new aggregate functions; and (3) using statistical techniques like sampling to answer OLAP queries approximately but quickly.

Warehouse design: Warehouse design issues interact with both data maintenance and querying goals. For example, self-maintainable views are significantly easier to maintain at the warehouse; choosing a set of auxiliary views to add to make the desired views be self-maintainable is a design issue. Since OLAP queries are mostly read-only, redundancy can be used in the warehouse design to improve query performance. This redundancy usually takes the form of indices and materializing datacubes. Choosing which points in a datacube to materialize and which of indexes to build, is a design issue. Other facets of warehouse design to be covered include star and snowflake schemas and specialized indices that are optimized for them.