INTRODUCTION

Information systems provide organizations with the necessary information to achieve their goals. Relevant information is gathered and stored to allow decision makers to obtain quick and elaborated reports from the data.

A data warehouse is an especially designed database that allows large amounts of historical and contextual information to be stored and accessed through complex, analytical, highly aggregated, but efficient queries. These queries capture strategic information, possibly presented in the form of reports and support management decision making. As we will see, data warehouses differ from general, transactional databases in many ways.

BACKGROUND

The pristine motivation for building an information system was to integrate the information that would be necessary to support decision making. With the advent of databases and transactional applications, the main goal of information systems drifted towards the organization of data in such a way that software applications work. This drift triggered a specialization of database technology to serve transactional information systems, evolving database design and implementation, as well as database management systems (DBMSs), towards this end. The kind of work performed on these databases was called on-line transactional processing (OLTP).

However, the need for decision-making support did not fade away. On the contrary, more and more organizations were requiring further analytical tools to support decision making: reporting tools, EIS (executive information systems), and other DSS (decision support systems) tools, which many DBMS vendors began to call “business intelligence.” These tools were able to exploit the database in a more powerful way than traditional query tools. These new tools made it easier to aggregate information from many different tables, to summarize data in a much more powerful way, and to construct textual and graphical reports over them, full of condensed, graphical, and statistical information. This other kind of work that was performed on the database was called on-line analytical processing (OLAP).

Although the previous scenarios, working on the same physical database, can still be found in many organizations all over the world, there are three important problems associated with this approach.

- First, and most important, is the use of the same database for both OLTP and OLAP. OLAP queries are usually complex, considering much historical data and interweaving many tables with several levels of aggregations. Many OLAP queries are “killer” queries, requiring many resources, and may highly disturb or even collapse the transactional work. Because of this, many reports and complex queries were run at night or during the weekends.

- Second, the data stored in the transactional database are the data required only by the applications. Data that are not to be used by any application are not stored. Additionally, many transactional databases remove or simply do not include historical data. On the other hand, adding all the historical data and other sources of information for OLAP work in the same transactional database can be an important overhead storage for the transactional work.

- Third, the design and organization of the database is specialized for OLTP: Normalization is common, indexes are created for improving transactions, and so forth. However, these choices might not be good for OLAP operations. This means that even with a separate, devoted database (i.e., a replica), OLAP will not be efficient for large databases.

The previous problems stimulate the construction of separate data repositories, specialized for analytical purposes. In the early nineties, these repositories were called data warehouses and its associated technology data warehousing (Inmon, 1992). The attention widened from enterprises and vendors, but it was not until the late 1990s that the academic world paid attention. All this and the appearance of more mature tools turned data warehousing into a new database discipline on its own.
DATA WAREHOUSES

The establishment of data warehousing as a new discipline has much to do with the fact that, once one database (the transactional) has been clearly separated from the other (the historical/analytical), they are quite different kind of “beasts.” Table 1 shows the strong differences between them. The special characteristics of the data and operations in the data warehouse case has led to a specialization in the data warehousing technology, establishing new designing paradigms, adopting other data models, operators, and implementation tools.

Having seen the advantages of constructing a separate data repository, the first question is to determine the data that will be included in the repository. This will surely depend on the analysis of the data requirements for the business intelligence applications that will use the data warehouse, such as the OLAP tools, the DSS systems, and, possibly, data mining. Frequently, this means that part of the information existing in the transactional databases (but not all and not at the same level of detail) has to be loaded into the data warehouse, as well as any external data that could be useful for the analytical processing. Figure 1 illustrates this integration.

Integrating data from several sources, with different formats, data models, and metadata, is not an easy task. Integration from different sources has always a negative effect on data quality; missing data are created because some features exist in some sources but not in others, as well as incompatible formats, units, and so forth. It is extremely important that data are transformed into unified formats and schema, and cleansed from missing and anomalous data. We will review these issues, but next we must address another crucial issue, the way in which information is organized in a data warehouse.

Data Warehouse Organization: The Multidimensional Model

One idea for constructing a data warehouse can be to apply a traditional design, using an entity-relationship or UML-like conceptual model, and transform it into a relational or an object-relational database, integrating all the sources of information into one highly normalized, off-the-shelf database. However, this has been shown to be a bad way for organizing data in a data warehouse. The reason is that classical data models, and especially the relational model, are well suited for performing transactional work but do not deal well with complex queries with a high degree of aggregation.

As a consequence, the most widespread model for data warehouses is a different data model: the multidimensional model (Golfarelli, Maio, & Rizzi, 1998). Data under the multidimensional model are organized around facts, which have related measures and can be seen in more or less detail according to certain dimensions. As an example, consider a multinational European supermarket chain. Its basic facts are the sales, which can have several associated measures: total income amount, quantity, number of customers, and so forth, and can be detailed in many dimensions: the time of purchase, the product sold, the place of the sale, and so forth. It is enlightening to see that measures usually correspond to the question, how much and how many? whereas the dimensions correspond to the questions when, what, and where.

The interesting thing about the dimensional model is that it eases the construction of queries about facts at several levels of aggregation. For instance, the fact “the sales summed up 22,000 products in the subcategory pet accessories in all the supermarkets in Spain in the first quarter of 2005” represents a measure (quantity = 22,000) of a sale with granularity month for the time dimension (first quarter of 2005), with granularity country for the location dimension (Spain) and with granularity subcategory for the product dimension (pet accessories).

The structure of possible aggregations on each dimension constructs a hierarchy from which we can select any granularity. For instance, as shown in Figure 2, the dimension time is hierarchical, existing several
Related Content

Relaxing Queries with Hierarchical Quantified Data Abstraction
[www.igi-global.com/article/relaxing-queries-hierarchical-quantified-data/3394?camid=4v1a](www.igi-global.com/article/relaxing-queries-hierarchical-quantified-data/3394?camid=4v1a)

Sequential Pattern Mining from Sequential Data
[www.igi-global.com/chapter/sequential-pattern-mining-sequential-data/20748?camid=4v1a](www.igi-global.com/chapter/sequential-pattern-mining-sequential-data/20748?camid=4v1a)

Toward a Formal Semantics for Control-Flow Process Models
[www.igi-global.com/article/toward-formal-semantics-control-flow/65542?camid=4v1a](www.igi-global.com/article/toward-formal-semantics-control-flow/65542?camid=4v1a)

Data Dependencies in Codd's Relational Model with Similarities
[www.igi-global.com/chapter/data-dependencies-codd-relational-model/20371?camid=4v1a](www.igi-global.com/chapter/data-dependencies-codd-relational-model/20371?camid=4v1a)