# DATA WAREHOUSES – MODELS, TECHNIQUES AND APPLICATIONS

KRZYSZTOF GOCZYŁA

*Faculty of Electronics, Telecommunications and Informatics,*
*Gdansk University of Technology,*
*Narutowicza 11/12, 80-233 Gdansk, Poland*
*kris@eti.pg.gda.pl*

(Received 5 January 2011; revised manuscript received 20 February 2011)

**Abstract:** This paper discusses the basic concepts of modern data warehouses. It presents the multidimensional data model (logical model) and the physical model of a data warehouse, as well as selected design and implementation issues. The focus is on the practical aspects of the application of data warehousing in business enterprises and organizations.

**Keywords:** data warehousing, business intelligence, star schema, ETL process

## 1. Introduction

Data warehouse technology has been around for a dozen or so years. The development of this technology (and of its theoretical rudiments) was initiated by R. Kimball [1], however, the need for data warehouse systems stems from practical applications. Due to the popularization of relational databases for the purposes of business transactions in the 1990s, American companies found themselves gathering large amounts of historical business data. For a company, these data could naturally constitute the basis for various statistical analyses, carried out for the purposes of strategic planning. With the competitiveness on the market on the rise, such analyses could prove invaluable for further development, even survival, of a company. Therefore, the concept of data warehousing, and of the elegant data model behind it, have met with a favourable response and started to develop rapidly. Data warehouses became so popular that the corresponding concepts were reflected in the most recent standards of SQL, even though specific systems usually use their own specialized access languages.

This paper, perforce briefly, reviews the models, techniques and applications of data warehouses. Section 2 gives and discusses the definition of a data warehouse. Section 3 defines the multidimensional data model, which constitutes the logical core of the concept of a data warehouse. Section 4 discusses the selected aspects of the logical modelling of data warehouses, while Section 5 focuses on the aspects of physical modelling, including several implementation aspects. Finally, Section 6 presents possible applications.

## 2. Defining a data warehouse

From among numerous informal definitions of a data warehouse, the following is the most adequate for the purposes of this paper:

> A data warehouse is a *centralized*, *non-transactional database* for information storage over a long time span, *globally* at the level of institutions, in *multidimensional analytical structures*, aimed at searching for and analysing information directly by end users.

Let us focus on the significant elements of this definition, italicized above.

- *Database* – a data warehouse is a database with all corresponding characteristics; it is, however, a database optimized for *analytical processing*, as opposed to transactional processing (more on that later). Data warehouses are usually very large databases, comprising many terabytes, which is a consequence of the potentially long time span of the stored information (the longer the better).
- *Non-transactional* (*analytical*) processing – operations on a data warehouse do not change its contents and mainly consist in extracting information through various intersections and aggregates (typically we are not interested in individual data). Such processing is termed online analytical processing (OLAP), as opposed to online transactional processing (OLTP) typical of "traditional" databases, used in everyday business activities. OLAP is characteristic of *business intelligence* applications, *i.e.* applications supporting strategic decision-making of a company.
- *Centralized* – a data warehouse is a centralized, as opposed to distributed, database. The data from individual external database systems are gathered during the process known as extract, transform and load (ETL) in one computer system, in which the data warehouse resides, where they are processed by OLAP. Centralization of the data is crucial, since, as follows from the above definition, OLAP analyses must be carried out systematically by the users. Any delays related to the transmission of large data sets over the network are unacceptable. Moreover, from the perspective of data warehouse management, the homogeneity (unification) of data is of paramount importance.
- *Globally* – a data warehouse should encompass the entire activity of a company in a given domain. For instance, if a company is engaged in trade, a data warehouse should encompass data from all retail outlets, for all the products in which it trades, and over the longest time span possible. Only this can guarantee that the results of OLAP are reliable. Occasionally, however, the scope of a data warehouse is purposefully restricted to a certain

geographical area or to one aspect of a given activity. In such a case we deal with a data mart. Usually the only difference between a data warehouse and a data mart is their size, and the scope of functionality is similar.

- *Multidimensional analytical structures* – the so-called multidimensional data model constitutes a logical basis for a data warehouse. This model, although directly related to the classical relational model, is distinctive enough to be discussed in a separate section.

## 3. Multidimensional data model

When creating a logical model of a given data warehouse, we use an approach based on a multidimensional model, according to which the basic data of a warehouse are stored as *facts*, which can be the subject of quantitative analyses. Facts are associated with *measures*, which are numerical. The role of *dimensions* is to aggregate facts according to different criteria imposed on dimensions. *Dimension members* are text labels describing facts. In this way, a data warehouse can be perceived as a multidimensional OLAP cube, whose constituents are facts selected by coordinates corresponding to elements of individual dimensions.

The concept of a multidimensional OLAP cube can be illustrated with a simple example. Let us consider a trade network consisting of a number of retail outlets. The simplest data warehouse for such a network could look like the one shown in Figure 1. This warehouse has 3 dimensions: Product, Outlet and Time, representing, respectively: products sold in this network, its individual outlets and the calendar time, accurate to a day. In reality, the number of elements along each of these dimensions would be much larger. A triple ($p_i$, $o_j$, $d_k$), corresponding to an elementary cube in Figure 1, represents the fact of selling product $p_i$ in outlet $o_j$ on day $d_k$. Each fact must be associated with certain measures, such as, in this case, the amount and income. The amount is a number which tells us how much of a given product was bought in a given outlet on a given day, while the income is the total amount of money paid by the customers. Note that the assumed time granulation is essential here. Since we assumed a day to be the elementary unit of time, a single fact is in reality an aggregate of individual operations of buying a given product in a given outlet, which all took place on the same day.

Not all elements of the OLAP cube must be filled. In fact, an OLAP cube can be thought of as a "multidimensional sieve", whose holes signify a lack of facts for the corresponding coordinates on the dimension axes.

Analytical queries issued to such a cube could include, for instance:

1. give the average daily sale of milk in all outlets of the network;
2. give the total sale of bread in the entire network in 2010;
3. verify whether there is correlation between the sales of milk and bread.

Each of these queries deals with different problems related to OLAP analyses. Query no. 1 is the simplest: it suffices to calculate the average value for the measure "Amount" over all the existing facts for the dimension member "Product" that corresponds to milk. Query no. 2 is related to the dimension member of

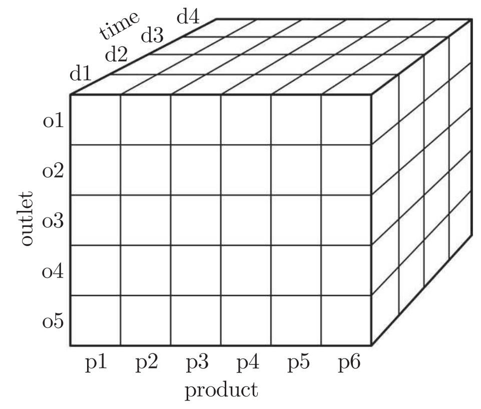

**Figure 1.** Example of a 3-dimensional OLAP cube

"Time" (not given explicitly), in this case, a year. Therefore, it entails calculating the given value only for the dimension members of "Time" in the given year. In fact, this means that the dimension member of "Time" is naturally hierarchical, which is worth noticing at the stage of warehouse design (this will be discussed later). Query no. 3 is of an entirely different kind. Answering this query requires using appropriate *business intelligence* tools, often employed in "wrappers" for OLAP cubes. Such tools enable performing various advanced statistical analyses on the data stored in the cube.

## 4. Logical modelling of data warehouses

Logical modelling of data warehouses is carried out according to the rules of relational modelling, however, certain additional rules must be observed. These rules, in their simplest form, can be formulated as follows:

1. define the fact table;
2. define the dimension tables related to the fact table by 1-to-$n$ mappings, with "1" corresponding to the dimension tables.

In so doing, we obtain a star schema, which for the cube presented in Figure 1, could look like the one shown below in Figure 2.

At the centre of the star schema there is a fact table, which contains all foreign keys from the dimension tables (product_no., outlet_no., day_no.). The dimension tables have surrogate keys, although it is good practice to leave natural (business) keys in them (in this case: product name, outlet name, day). Clearly, each single sale can be mapped to exactly one product, one outlet and one day, whereas the converse mapping is a mapping to "many".

Let us make several important observations about a star schema. Firstly, a star schema is characterized by data redundancy. For instance, if there are 30 outlets in Gdansk, the information that Gdansk is located in Pomorskie Province must be repeated 30 times. Similar holds for the Time table (*e.g.* each of the 12 months will feature the same year) and for the Product table (many

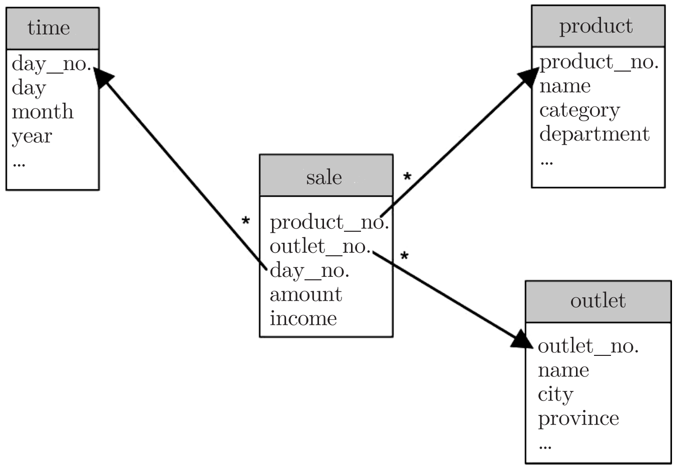

**Figure 2.** Star schema for the cube in Figure 1

products belong to the same category, many product categories belong to the same department, *etc.*). From relational database theory, we know that this is a manifestation of the violation of the third normal form (3NF). In the case of data warehouses this does not constitute a disadvantage, since it is the size of the fact table that accounts for the majority (more than 90%) of the size of a data warehouse, and, moreover, a data warehouse is not subject to updating (except for periodical data loading, see below) and therefore there is no risk of update anomalies known from theory. However, if the size of the dimension tables becomes non-negligible, we can expand the star schema into a snowflake schema, the discussion of which is beyond the scope of this paper.

Another characteristic feature of the star schema is the fact that certain dimensions can have a hierarchical nature. Let us consider the Time table: one month consists of many days, one year consists of many months, *etc.* The situation can be similar for other tables. We can thus define a dimension in such a way that it contains different elements of the dimension at different levels of the hierarchy. This allows us to easily formulate such queries as query no. 2 in Section 3. This does not mean that every dimension is hierarchical. Let us imagine that we expand our cube with a dimension called Promotion, whose elements could be: a newsletter, radio, television, corresponding to the medium by means of which the given product was advertised. In this case we do not deal with any reasonable hierarchy; dimensions like that are termed categorical.

There are many further aspects of designing a logical model of a data warehouse. Those most important from the practical viewpoint are creating the so-called constellations of stars (or of snowflakes), defining computed measures and degenerate dimensions, the issues of modelling $n$-to-$n$ mappings, recursive structures in the dimension tables, *etc.* We refer interested readers to the extensive literature on the topic, for instance to [2–4].

## 5. Physical model of a data warehouse

The process of building a data warehouse consists of several stages and largely depends on the form and quality of the source data, *i.e.* of the transaction data gathered at business venues. In general, this process consists of three stages: extracting data, transforming data, and loading data, for this reason it is termed ETL (extract, transform and load), *cf.* Figure 3.

Designing the Extract stage consist in determining the data necessary for strategic analyses, identifying the sources of these data, coming up with mappings between the source and target data (*e.g.* changing the format of a spreadsheet or a text file into a unified format of a relational database, unifying date and currency formats, *etc.*), defining program procedures for extracting the data, *etc.* This is an extremely important stage of the entire process, since the quality of business analyses strongly depends on the quality of the source data. The absence of source data or the impossibility of gathering them at business venues makes the implementation of *business intelligence* solutions in a company a futile effort.

Designing the Transform stage consists in working out the rules of transforming and cleansing the data. Transformation rules are certain algorithmic rules of transforming the source data into target data. For instance, let us assume that the source data contains the customer's date of birth and the date the transaction took place. Meanwhile, in the data warehouse, in the Customer dimension table, there is a customer_age attribute. In such a case, the transformation would consist in calculating the customer's age on the basis of their data of birth and the date of transaction. The transformation can also consist in changing numerical values into category labels. For instance, let us assume that we are interested in analysing sales depending on the size of the city. Such an analysis is only reasonable if we divide the cities into categories, *e.g.* small, medium, large and metropolises. However, if the source data specifies the number of inhabitants of a given city (accurate to, say, a thousand people), we must transform the numerical value into the corresponding category value. Data cleansing consists in finding and correcting errors and logical inconsistencies in the data. To this end, we define how exceptional situations, such as missing or outlying data, should be handled. What should we do if, for instance, the value of income is missing in the description of a certain transaction. Should we ignore such a situation? Assume the income is zero? Or perhaps arbitrarily assume a certain value calculated on the basis of previous transactions? What if the description of a transaction contains an exceptional value, say, an order of magnitude larger than for other similar transactions? We could accept this value unquestioningly and place it in the warehouse, or we could assume that this value is incorrect and employ some method to correct it. In planning such activities, we have at our disposal tools that aid in designing the ETL process in a given data warehouse system.

Designing the Load stage consists in defining memory structures for the transformed and cleansed source data, as well as in creating program procedures of loading the data into these structures. These structures, termed a staging area,

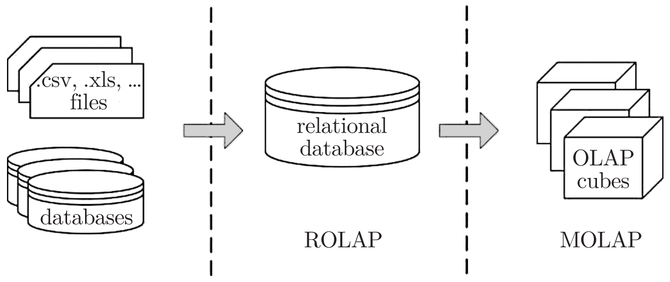

**Figure 3.** Data transfer in ETL

typically constitute a certain relational database with a schema determined by the logical model of the warehouse (a star schema or similar). The staging area can become the target area of the warehouse; in other words, such a warehouse is implemented as a relational database and is not subject to further transformations. In this case, we say that the physical model of a warehouse is a ROLAP (*Relational* OLAP) model. In this model, queries to a data warehouse are formulated in standard SQL with extensions oriented at the optimization of grouping (CUBE and ROLLUP commands and associated functions [5]). However, higher efficiency and wider capabilities of formulating strong analytical queries can be achieved only when the data from the staging area are loaded into specialized memory structures of a MOLAP (*Multidimensional* OLAP) model. The MOLAP structures are optimised for OLAP processing in various ways. Firstly, we know that the queries will not modify data, and therefore the mechanisms of handling transactional processing and conflict resolution for concurrent access by many users will not be needed. Secondly, we know that the most common operations will be groupings and the calculation of aggregates. Therefore, we can design data structures in such a way that these two types of operations are performed very efficiently. This can be achieved in many ways, *e.g.* we can tentatively calculate the aggregates when loading the data from the ROLAP model into the MOLAP model. The system operating a data warehouse will be able to "predict" the queries and have the answers ready. Naturally, in real-life data warehouses, where the number of dimensions is in the order of several dozen, with most of them hierarchical, the number of possible aggregations is vast, therefore, initially only the aggregates at higher levels of aggregation are calculated. They will subsequently serve as ready-made answers to the queries or as data that can be used to accelerate the realization of queries at lower levels of aggregation. The MOLAP model requires additional memory, which in the case of large warehouses can significantly affect the costs of constructing and maintaining the system. To circumvent this problem, hybrid models, termed HOLAP (*Hybrid* OLAP), can be utilized, where the fundamental data of the warehouse remain in a ROLAP area, whereas the MOLAP structures only store the tentatively calculated aggregates.

Physical modelling of a data warehouse entails many other important aspects, the discussion of which is beyond the scope of this paper. For instance, the problem of dividing a large warehouse into independently stored partitions, as well as the need to define procedures for periodically and incrementally adding new data to the warehouse. We refer interested readers to [6], online documentation systems of companies specializing in data warehouse systems and *business intelligence* tools, such as Oracle (www.oracle.com) or SAS (www.sas.com), as well as to other repositories, including those related to non-commercial products.

## 6. Applications

Data warehouse systems and *business intelligence* solutions in general are becoming more widely applied in various fields, above all, those characterised by handling mass transactions performed by multiple customers. Telecommunications (fact: a single connection), banking (fact: a bank transaction), insurance (facts: entering an agreement, damage repair), air transport (fact: flight of a single passenger), and finally the abovementioned retail network are examples of such business fields. Coming up with possible measures and dimensions for such data warehouses is left as an exercise to the Reader. Since the enterprise of implementing a data warehouse in a company, "wrapped" with suitable *business intelligence* tools is expensive and carries a business risk, there are remote services, which offer transferring the data to a remote data warehouse, where they are analysed and the results are transferred back to the client (this service could be termed *cloud warehousing*). The disadvantage of this approach is that the data must be physically transferred to a remote warehouse and made externally accessible, which can be unacceptable for many companies. Therefore, the enterprise of creating an in-house data warehouse is, and undoubtedly will be, one of the most serious IT ventures carried out in many companies nowadays.

It is worth noticing that data warehouses, together with the tools of statistical analysis can serve not only to support management, but also as powerful tools for research dealing with a large amount of experimental or measurement data. In such applications, where there is no need to implement them into existing business activities, we can afford using non commercial solutions, which makes it possible to significantly reduce the costs of the tools used, along with their maintenance and administration.

### *References*

[1] Kimball R 1996 *Data Warehouse Toolkit*, J. Wiley&Sons

[2] Inmon W H 2002 *Building the Data Warehouse*, J. Wiley&Sons

[3] Poe V, Klauer P and Brebst S 2000 *Development of a Data Warehouse*, WNT (in Polish)

[4] Ponniah P 2001 *Data Warehousing*, J. Wiley&Sons

[5] Connolly T M and Begg C E 2005 *Database Systems: a Practical Approach to Design, Implementation and Management*, Pearson Education

[6] Mendrala D nad Szeliga M 2009 *SQL 2008. Business Services. Analysis and Mining of Data*, Helion (in Polish)