Deductive Data Warehouses

Kornelije Rabuzin, University of Zagreb, Varazdin, Croatia

ABSTRACT

This paper presents the idea of deductive data warehouses. Deductive data warehouses rely on deductive databases but instead of a database in the background a data warehouse is used. The authors show how Datalog (as a logic programming language) can be used to perform OLAP analysis on data. Since data warehouses don’t use all the technologies that databases do (locking, transactions, integrity constraints, etc., which are not relevant in this context), some things are different and simpler when working with deductive databases. The authors demonstrate the idea on an example and the authors show that Datalog rules can be used to perform OLAP analysis on data.

Keywords: Business Intelligence, Data Warehouse, Database, Datalog, Online Analytical Processing (OLAP)

Databases (DB) have been with us for a long time; relational databases for more than four decades and hierarchical and network databases for even longer. Codd’s papers on the relational data model that appeared in the 1960s and 1970s have greatly influenced events in the past decades. The ability to store and efficiently manage large amounts of data is very important in the era that we live in, making relational databases quite popular. Although the relational model dominates, NoSQL databases are becoming more and more popular. However, this paper does not cover this topic as it doesn’t influence the ideas presented in this paper. Some good database reference works include (Date, 2004; Garcia-Molina, Ullman & Widom, 2009; Paton, 1998).

However, during the past two decades one problem has appeared that has become quite obvious and still causes difficulties when making decisions. The problem is that (over the years) many applications have been implemented in variety of companies, and it turns out that all of them are quite incompatible and cannot work with each other (the word work here is used in the context of exchanging data). In other words, many applications exist but they are implemented in different programming languages and use different ways to store data. As a result it is hard to extract data to build reports that could be used to make decisions. In order to resolve this problem data warehousing has been introduced.

A data warehouse is a special kind of a database that is organized according to different design principles then databases are. Its purpose is to integrate (physically) data from different sources and provide a trusted place of important pieces of information. Although it may seem simple (at first), this is not easy to accomplish, and many data warehousing projects are not successful because many issues have to be resolved during the way. By integrating data from different sources one gets a database (called a data warehouse) that contains integrated, de-duplicated, cleaned and

DOI: 10.4018/ijdwm.2014010102
transformed data from different applications and such data can be used to make important decisions. The problem with heterogeneous applications (i.e., applications that use different programming languages and different ways to store data) is that each application supports some business process (or more processes), but to know what is really going on one (usually) has to compare data from different applications. Data warehouses are here to do that; once successfully integrated, data can be found in data warehouse and decision making becomes much simpler and much faster as well. All data from different systems has to be integrated in order to make decisions. This is not an easy task, even if all data were in databases. When data is scattered in different systems and stored in different formats, the integration problem is even more obvious. Integration and cleaning of data subsumes complex queries used to extract, clean and conform data in order to make them appropriate to make decisions (this process is known as ETL process i.e. Extract Transform Load).

In order to build a database that is usually (for end users) perceived as a set of tables, some techniques are used to build a data model for an application domain. One of the best known techniques is the ERA diagram (ERA stands for Entity – Relationship – Attribute). Although in the IT field many things are not old, this technique could be called old as it has been used for almost four decades. The reason why it is still so popular lies in the fact that it is quite simple to understand, even for non-technical users. The main goal is to define entities (i.e. types of entities), relationships among them (i.e. types of relationships) as well as attributes (that belong to types of entities); once the diagram is drawn, it can be easily transformed into a set of tables. With a little bit of experience some common pitfalls can be avoided and we can end up with good database design.

Normalization is also relevant for database design. During normalization the set of normal forms (1NF, 2NF, 3NF, BCNF, 4NF, 5NF and some other normal forms) is used to reduce anomaly and to avoid potential inconsistencies which also advances good database design. Although people that work with databases are usually not aware of the normalization theory, it is however very important and can be used to achieve good database design. Since this paper is not on normalization, other details are skipped; for more information you should look at Date (2012).

In order to work with databases one of the database languages must be used; one of the most popular languages is SQL. It is standardized and supported by many database management system vendors. SQL is a language that consists of a set of statements that can be divided into several categories:

- **DDL** stands for Data Definition Language; DDL is used to define (create) objects in database (i.e. tables, functions, sequences, etc.),
- **DML** stands for Data Manipulation Language; DML is used to manipulate data i.e. to insert, update or delete data, and
- **QL** is another category with a SELECT statement used to query (retrieve) data from database.

In order to use certain statements one has to be familiar with statement syntax. The most popular way to represent (define) the syntax of some SQL statement is to use BNF notation. The BNF notation uses several symbols (like [], {}, …) that have their meanings; based on the meaning one can construct a valid SQL statement. For example, this is INSERT statement syntax in PostgreSQL 9.2 database management system, presented in Box 1.

Square brackets represent that some things are optional, | represents or and … means that something can be repeated more than once. However, during the years SQL has become quite complex and some statements (especially queries) have started to cause problems even for professionals (and for end users as well).

In this paper we present the idea of deductive data warehouses; Datalog is used as a language to analyze data in the data warehouse by simulating (advanced) OLAP capabilities.
Formalizing the Mapping of UML Conceptual Schemas to Column-Oriented Databases
www.igi-global.com/article/formalizing-the-mapping-of-uml-conceptual-schemas-to-column-oriented-databases/208692?camid=4v1a

HASTA: A Hierarchical-Grid Clustering Algorithm with Data Field
www.igi-global.com/article/hasta/110385?camid=4v1a