Chapter 3.24
A Survey of Managing the Evolution of Data Warehouses

Robert Wrembel
Poznań University of Technology, Institute of Computing Science, Poland

ABSTRACT

Methods of designing a data warehouse (DW) usually assume that its structure is static. In practice, however, a DW structure changes among others as the result of the evolution of external data sources and changes of the real world represented in a DW. The most advanced research approaches to this problem are based on temporal extensions and versioning techniques. This article surveys challenges in designing, building, and managing data warehouses whose structure and content evolve in time. The survey is based on the so-called Multiversion Data Warehouse (MVDW). In details, this article presents the following issues: the concept of the MVDW, a language for querying the MVDW, a framework for detecting changes in data sources, a structure for sharing data in the MVDW, index structures for indexing data in the MVDW.

INTRODUCTION

Contemporary model of managing enterprises, institutions, and organizations is based on decision support systems. In these systems, knowledge is gained from data analysis. Nowadays, core components of the majority of decision support systems are data warehouses (DWs) (Kimball & Ross, 2002). The purpose of building a data warehouse is twofold. Firstly, to integrate multiple heterogeneous, autonomous, and distributed external data sources (EDSs) within an enterprise. Secondly, to provide a platform for advanced, complex, and efficient analysis of integrated data.

From a technological point of view, a DW is a large database that stores current and past elementary data as well as data aggregated at different levels of granularity. These data are analyzed by the so-called *On-Line Analytical Processing* (OLAP) applications, based on complex queries.

OLAP applications are used for the purpose of discovering trends, patterns of behavior, and anomalies as well as for finding dependencies between data. The process of good decision making often requires forecasting and simulating future business behavior, based on present and past data as well as based on assumptions made by decision makers. This kind of data processing is called a 'what-if' analysis.

For a long period of time the existing DW technologies and research contributions have tacitly assumed that the structure of a DW is time invariant. As a consequence, many of the research developments and most of the commercially available DW technologies offer functionalities for managing data warehouses of static (time invariant) structures. In practice, however, a DW structure changes as the result of the evolution of external data sources (Rundensteiner, Koeller, & Zhang, 2000), changes of the real world represented by a DW, new user requirements, as well as the creation of simulation environments, to list the most common cases.

Managing the evolution of a DW is challenging from a research and technological point of view. The basic issues that should be solved include: (1) a DW model and metadata capable of representing and storing the history of evolution that concerns not only data but also data structures, (2) a query language capable of querying possibly heterogeneous DW states and capable of analyzing metadata on an evolving DW, (3) techniques for detecting changes in EDSs that have an impact on the structure of a DW, (4) physical data structures supporting storage and efficient access to evolving data.

This article discusses challenges in designing, building, and managing data warehouses whose structure and content evolve in time. The issues and solutions presented in this article are based on an experience of the author and his team in building a prototype Multiversion Data Warehouse (MVDW). This article is organized as follows. First, we present basic definitions used in this article. Second, we discuss real world examples illustrating DW evolution. Third, we present the concept of the MVDW. Fourth, we overview an approach to querying the MVDW. Next, we overview a framework for the detection of changes in EDSs and propagating them into the MVDW. Then, we describe a technique for sharing data between multiple DW versions. Next, we present solutions for indexing data in the MVDW. Finally, we summarize the article and discuss possible areas for future work.

BASIC DEFINITIONS

Data Modeling

Data in a DW are organized according to a dedicated model (Gyssens & Lakshmanan, 1997; Letz, Henn, & Vossen, 2002). In this model, an elementary information being the subject of analysis is called a *fact*. It contains numerical features, called *measures* (e.g., quantity, income, duration time) that quantify the fact and allow to compare different facts. Values of measures depend on a context set up by *dimensions*. A dimension is composed of *levels* that form a hierarchy. A lower level is connected to its direct parent level by a relation, further denoted as ↗. A dimension contains a distinguished top level, denoted as l_{All}, and a terminal/bottom level. Every level l_i has associated a domain of values. The finite subset of domain values constitutes the set of *level instances*. The instances of levels in a given dimension are related to each other, so that they form a hierarchy, called a *dimension instance*.

A typical example of a dimension, called *Location*, is shown in Figure 1a. It is composed of four hierarchically connected levels, i.e., *Shops* ↗*Cities*↗*Regions*↗l_{All}. An example instance of dimension *Location* is shown in Figure 1b. It is composed of 10 related level instances.

The aforementioned model of a DW can be implemented either in multidimensional OLAP