A Functional Model for Data Analysis

Nicolas Spyratos*

Laboratoire de Recherche en Informatique,
Université de Paris-Sud,
91405 Orsay Cedex, France
spyratos@lri.fr

Abstract. We present a functional model for the analysis of large volumes of detailed transactional data, accumulated over time. In our model, the data schema is an acyclic graph with a single root, and data analysis queries are formulated using paths starting at the root. The root models the objects of an application and the remaining nodes model attributes of the objects. Our objective is to use this model as a simple interface for the analyst to formulate queries, and then map the queries to a commercially available system for the actual evaluation.

1 Introduction

In decision-support systems, in order to extract useful information from the data of an application, it is necessary to analyse large amounts of detailed transactional data, accumulated over time - typically over a period of several months. The data is usually stored in a so-called "data warehouse", and it is analysed along various dimensions and at various levels in each dimension [5, 10, 12].

A data warehouse functions just like a usual database, with the following important differences: (a) the data of a data warehouse is *not* production data but the result of integration of production data coming from various sources, (b) the data of a data warehouse is *historic* data, that is data accumulated over time, (c) access to the data warehouse by analysts is almost exclusively for reading and not for writing and (d) changes of data happen only at the sources, and such changes are propagated periodically to the data warehouse.

The end users of a data warehouse are mainly analysts and decision makers, who almost invariably ask for data aggregations such as "total sales by store", or "average sales by city and product category", and so on. In this context, the basic requirements by data analysts are (a) a data schema that is easy to understand and (b) a flexible and powerful query language in which to express complex data analysis tasks. The so called "dimensional schemas" and their associated "OLAP query languages" were introduced precisely to satisfy these requirements.

* Work conducted in part while the author was a visitor at the Meme Media Laboratory, University of Hokkaido, Sapporo, Japan.

H. Legind Larsen et al. (Eds.): FQAS 2006, LNAI 4027, pp. 51–64, 2006.

This paper is focused on dimensional schemas and their OLAP query languages, as opposed to normalized relational schemas and their transaction processing languages.

Schema normalization was introduced in relational databases with the goal of increasing transaction throughput. Normalized schemas, however, rarely reflect the "business model" of the enterprise, that is the way the enterprise actually functions. Their main concern is to make database updating as efficient as possible, usually at the cost of rendering the schema virtually incomprehensible by the non specialist. Therefore normalized schemas are not suitable for data warehouses, as the analysts and decision makers of the enterprise are unable to "read" the schema and to formulate the queries necessary for their data analyses.

On-Line Analytic Processing, or OLAP for short, is the main activity carried out by analysts and decision makers [3, 4]. However, although several SQL extensions are available today for OLAP, there seems to be no agreement as to a simple conceptual model able to guide data analysis. The objective of this paper is to propose such a model.

The products offered today by data warehouse vendors are not satisfactory because (a) none offers a clear separation between the physical and the conceptual level, and (b) schema design is based either on methods deriving from relational schema normalization or on *ad hoc* methods intended to capture the concept of dimension in data. Consequently, several proposals have been made recently to remedy these deficiencies.

The proposal of the cube operator [7] is one of the early, significant contributions, followed by much work on finding efficient data cube algorithms [2, 9]. Relatively little work has gone into modelling, with early proposals based on multidimensional tables, called cubes, having parameters and measures [1, 11]. However, these works do not seem to provide a clear separation between schema and data. More recent works (e.g. in [8]) offer a clearer separation between structural aspects and content (see [17] for a survey).

However, a common characteristic of most of these models is that they somehow keep with the spirit of the relational model, as to the way they view a tuple in a table. Indeed, in all these models, implicitly or explicitly, a tuple (or a row in a table) is seen as a function associating each table attribute with a value from that attribute's domain; by contrast, in our model, it is each attribute that is seen as a function. Our approach is similar in spirit to the one of [6] although that work does not address OLAP issues.

Roughly speaking, in our model, the data schema is an acyclic graph with a single root, and a database is an assignment of finite functions, one to each arrow of the graph. The root of the graph is meant to model the objects of an application, while the remaining nodes model attributes of the objects. Data analysis queries (that we call OLAP queries) are formulated using paths starting at the root, and each query specifies three tasks to be performed on the objects: a classification of the objects into groups, following some criterion; a measurement of some property of objects in each group; a summarization of the measured properties in each group, with respect to some operation.