Integrating Star and Snowflake Schemas in Data Warehouses

Georgia Garani, Department of Computer Science and Telecommunications, Technological Educational Institute of Larisa, Greece

Sven Helmer, Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy

ABSTRACT

A fundamental issue encountered by the research community of data warehouses (DWs) is the modeling of data. In this paper, a new design is proposed, named the starnest schema, for the logical modeling of DWs. Using nested methodology, data semantics can be explicitly represented. Part of the design involves providing a translation mechanism from the star/snowflake schemas to a nested representation. The novel schema proposed in this paper is accomplished by converting the fact-dimension schema to a fact-nested dimension schema. The transformation of the denormalized dimension tables to nested dimension tables increases the efficiency of query execution by reducing the number of tuples accessed for query retrieval since dimensional attributes can be used directly in the Group-by clause. In order to facilitate the implementation of the proposed approach, specific algorithms are built based on the starnest schema.

Keywords: *Data Warehouse, Database, Logical Modeling, Nested Relation, Snowflake Schema, Star Schema*

INTRODUCTION

Data Warehouses (DWs) are repositories used for analyzing large volumes of archived data, in contrast to transactional systems, which focus on keeping operational data current. Typical applications of DWs can be found in areas such as decision support, data mining, market research and fraud detection.

The way DWs are employed has an impact on the data access patterns. Data analysts sift through huge data sets by sending many ad-hoc queries in an iterative fashion to the system, while updates are usually limited to appending new data extracted from operational systems. This is also reflected in how the data is organized in a DW: the preferred modeling approach is based on multiple dimensions. In the multidimensional data model there are measures representing numerical properties and dimensions categorizing the measures. For example, a measure could be a revenue in U.S. dollars, while the context of this number is set by dimensions such as region, time, and product: in January 2010 company A made $1,420,650 selling product B in the Midwest region. Usually, dimensions are organized hierarchically, for example, for the dimension time could have the categories year, quarter, month, and day. Analysts are very often interested in

DOI: 10.4018/jdwm.2012100102

aggregated and summarized data rather than individual facts and use the dimensions to select the appropriate level of aggregation. During data analysis they may even switch from one level of granularity to another. Switching to a finer level of granularity is called drill-down, switching to a coarser one roll-up.

Given that the relational data model does not directly support the multidimensional data model, there is no standard way of mapping measures and dimensions to relational tables. Two of the most popular and well-known schemas for implementing the multidimensional model in a relational database system are the star schema and the snowflake schema. Common to both schemas is the way they handle measures, which are mapped into (large) fact tables. Staying with the example, this means that every single sale done by company A is recorded in a fact table. Clustered around the fact tables are dimension tables, which partition the fact tables along the different categories. Snowflake and star schemas differ in the way they manage the dimensions. In a snowflake schema the dimension tables are normalized, i.e., the hierarchy of a dimension is broken down into different tables. For our instance, there is one table each for year, quarter, month and day. In a star schema, on the other hand, all the information for one dimension is stored in one denormalized table. For our example, there is one table with the attributes year, quarter, month and day.

Each of the two approaches has advantages and disadvantages. Snowflake schemas adhere to the normalization principles of relational design theory for the dimension tables. As a consequence, there is no redundancy in the tables, which decreases the storage overhead. Additionally, there are no anomalies and the schema is easier to extend. However, there is also a downside: during query processing the different hierarchy levels of a dimension have to be joined together with surrogate keys, which introduces a computational overhead. In a star schema, on the other hand, the dimensional hierarchy does not need to be re-assembled via costly join operations. Nevertheless, a denormalized schema is more difficult to maintain, due to the redundancy and may lack some clarity, since all the different hierarchy levels of a dimension and their dependencies are mapped into a single table.

We propose a new kind of schema, called starnest schema, which combines the advantages of both, the star and the snowflake schema. As in the star schema, we do not need expensive join operations to combine hierarchy levels, but we can avoid redundancy like in the snowflake schema. We achieve this goal by storing each dimension in a single nested table, preserving a dimension's hierarchy in a natural way. Our concrete contributions are described in the following. First, we give a formal definition of our novel starnest schema. Second, we present an algorithm for transforming a star schema into a starnest schema. Thus, a designer can use existing tools to develop a star schema and then, in a final step, convert it into a starnest schema. Third, we show how queries in a starnest schema can be processed employing a powerful nested relational algebra. Finally, we illustrate the application of our concepts in a case study, demonstrating that the starnest schema is a serious contender in the area of DW modeling.

The remainder of the paper is organized as follows. After the preliminaries, the main logical modeling approaches are presented. The starnest schema is introduced afterwards, the transformation of a star schema to the starnest schema is discussed and an algorithm is presented for this purpose. Query processing is examined. A case study is presented and related work is discussed. Finally, the paper is concluded and further research issues are outlined.

PRELIMINARIES

Definition 1 (Fact): A *fact* is a set of events used for decision-making process in a business.

Example 1: In the healthcare system, an example of a fact is Operations.

Discovering Frequent Embedded Subtree Patterns from Large Databases of Unordered Labeled Trees