A Hybrid Approach for Data Warehouse View Selection

Biren Shah, University of Louisiana at Lafayette, USA
Karthik Ramachandran, University of Louisiana at Lafayette, USA
Vijay Raghavan, University of Louisiana at Lafayette, USA

ABSTRACT

Materialized view selection is one of the crucial decisions in designing a data warehouse for optimal efficiency. Static selection of views may materialize certain views that are not beneficial as the data and usage trends change over time. On the contrary, dynamic selection of views works better only for queries demanding a high degree of aggregation. These facts point to the need for a technique that combines the improved response time of the static approach and the automated tuning capability of the dynamic approach. In this article, we propose a hybrid approach for the selection of materialized views. The idea is to partition the collection of all views into a static and dynamic set such that views selected for materialization from the static set are persistent over multiple query (and maintenance) windows, whereas views selected from the dynamic set can be queried and/or replaced on the fly. Highly aggregated views are selected on the fly based on the query access patterns of users, whereas the more detailed static set of views plays a significant role in the efficient maintenance of the dynamic set of views and in answering certain detailed view queries. We prove that our proposed strategy satisfies the monotonicity requirements, which is essential in order for the greedy heuristic to deliver competitive solutions. Experimental results show that our approach outperforms Dynamat, a well-known dynamic view management system that is known to outperform optimal static view selection.

Keywords: data warehouse; dynamic selection; hybrid selection; OLAP; static selection; view materialization; view selection

INTRODUCTION

Decision Support Systems (DSS) involve complex queries on very large databases. While operational databases maintain state information, data warehouses typically maintain historical information. A data warehouse stores a large collection of subject-oriented, integrated, time-variant, and nonvolatile data in support of management’s decision-making process. As a result, data warehouses tend to be very large and grow over time. To facilitate answering such complex queries that span over large amounts of data, the data
are Extracted, Transformed, and Loaded (ETL) (Kimball, 1996; Kimball & Caserta, 2004; Schlesinger et al., 2005) into the warehouse and stored in a way that supports common analytical operations. The ETL phase of the data warehouse development life cycle is one of the most difficult, time-consuming, and labor-intensive tasks of building a data warehouse. In most cases, the data that are extracted from two or more sources is dirty in nature due to its heterogeneous representation. The data that are heading to the data warehouse thus first need to be cleaned (Ezeife & Ohanekwu, 2005; Lee et al., 1999) in order for the warehouse to be reliable.

Online Analytical Processing (OLAP) and OLAP mining (Tjioe & Taniar, 2005) refer to a set of data analysis and mining techniques developed for analyzing data in data warehouses. To facilitate such analysis, the data warehouse presents a multidimensional, logical view of the data and, hence, is called a multidimensional database or a multidimensional data cube (Gray et al., 1996, Tan et al., 2004). The data in such a multidimensional data warehouse generally are organized as a set of fact tables and indexed by attributes (primary keys) of the dimension tables that store dimension information. Fact tables are thin and long, whereas dimension tables are thick and small. A star schema model (Kimball, 1996), as shown in Figure 1, is used to represent a data warehouse.

In Figure 1, there are two dimension tables; namely, Product and Location and a central fact table Sales. The fact value that is measured is sales, which indicates the total sales of a particular product sold in a particular location. In most real-life applications, the dimensions are organized as hierarchies of attributes that define relationships among a set of attributes in the dimension. A simple example is organizing the Product dimension into the hierarchy: productId, and type. (none) indicates that there are no attributes in the group-by clause. A sample hierarchy for each of the dimensions used in schema of Figure 1 is shown in Figure 2. The length of a dimension hierarchy is defined as the number of levels excluding (none). For example, length of the Product hierarchy is equal to 2.

The multidimensional data cube (or simply cube), as defined by Gray et al. (1996), is referred to as a cube without hierarchies. In our example, this is: (productId, locationId), (productId), (locationId) and (none). The cube with hierarchies computes additional aggregates (views) for: (type, locationId), (productId, state), (type,
Related Content

Change Detection in Large Evolving Networks
[www.igi-global.com/article/change-detection-in-large-evolving-networks/225807?camid=4v1a](www.igi-global.com/article/change-detection-in-large-evolving-networks/225807?camid=4v1a)

A Survey on Temporal Data Warehousing
[www.igi-global.com/article/survey-temporal-data-warehousing/1820?camid=4v1a](www.igi-global.com/article/survey-temporal-data-warehousing/1820?camid=4v1a)

Intelligent Data Processing Based on Multi-Dimensional Numbered Memory Structures
[www.igi-global.com/chapter/intelligent-data-processing-based-multi/69408?camid=4v1a](www.igi-global.com/chapter/intelligent-data-processing-based-multi/69408?camid=4v1a)

GeoCache: A Cache for GML Geographical Data
[www.igi-global.com/article/geocache-cache-gml-geographical-data/1779?camid=4v1a](www.igi-global.com/article/geocache-cache-gml-geographical-data/1779?camid=4v1a)