Information Systems 37 (2012) 412–429

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

Time–HOBI: Index for optimizing star queries ☆

Tadeusz Morzy [a], Robert Wrembel [a,*], Jan Chmiel [a,b], Artur Wojciechowski [a]

[a] Poznań University of Technology, Institute of Computing Science, Poznań, Poland
[b] Allegro.pl, Poznań, Poland

ARTICLE INFO

Available online 15 June 2011

Keywords:
Data warehouse
Query optimization
Star query
Roll-up query
Indexing dimensions
Hierarchical index
Bitmap index

ABSTRACT

One of the important research and technological problems in data warehouse query optimization concerns star queries. So far, most of the research focused on optimizing such queries by means of join indexes, bitmap join indexes, or various multidimensional indexes. These structures neither support navigation well along dimension hierarchies nor optimize joins with the *Time* dimension, which in practice is used in most of the star queries. In this paper we propose an index, called *Time–HOBI*, for optimizing the star queries that compute aggregates along dimension hierarchies. *Time–HOBI*, created on a dimension hierarchy, is composed of (1) a *Hierarchically Organized Bitmap Index* (*HOBI*), where one bitmap index is maintained for one dimension level, and (2) a *Time Index* (*TI*) that implicitly encodes time in every dimension. *HOBI* allows to quickly search for fact rows satisfying predicates defined on different levels of dimension hierarchies. With the support of *TI* joining a fact table with the *Time* dimension is avoided. Thus, *Time–HOBI* supports a broad class of star queries. In this paper we explain how query execution plans for star queries can profit from *Time–HOBI*. We show, based on experiments, the efficiency of *Time–HOBI* for different classes of queries, as compared to *HOBI* and a traditional bitmap index. Based on the experiments, we also demonstrate how sensitive *Time–HOBI* is to variable selectivity of queries. We also analyze the maintenance time of *Time–HOBI* as compared to *HOBI* and a traditional bitmap index. The experiments used in the paper have been conducted on a real dataset, coming from the biggest East-European Internet auction platform *Allegro.pl*. The experiments show that *Time–HOBI* can be successfully applied to the optimization of star queries as it offers promising performance improvement.

1. Introduction

A data warehouse architecture has been developed in order to integrate and analyze data coming from multiple distributed, heterogeneous, and autonomous data sources (DSs) deployed throughout an organization. A core component of this architecture is a database, called a *data warehouse* (DW), that stores current and historical data, integrated from multiple DSs. The content of a DW is analyzed by various analytical queries for the purpose of discovering trends (e.g., demand and sales of products), discovering patterns of behavior (e.g., customer habits, credit repayment history), and anomalies (e.g., credit card usage) as well as for finding dependencies between data (e.g., market basket analysis, suggested buying, and insurance fee assessment).

Analytical queries are very complex (expressed with multiple join, filtering, aggregate, and sort operations) and they process large volumes of data. A special class of analytical queries are the so-called *star queries* that join a central table with multiple referenced tables (forming the so-called dimensions). The execution of analytical queries

☆ This work was supported from the Polish Ministry of Science and Higher Education Grant no. N N516 365834.

* Corresponding author.
E-mail addresses: Tadeusz.Morzy@put.poznan.pl (T. Morzy), Robert.Wrembel@put.poznan.pl (R. Wrembel), Jan.Chmiel@allegro.pl (J. Chmiel), Artur.Wojciechowski@put.poznan.pl (A. Wojciechowski).

0306-4379/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.is.2011.06.002

may take hours or even days. For this reason, providing means for increasing the performance of analytical queries is one of the important research areas. In this area, different mechanisms have been proposed in the research literature and applied in practice, i.e., materialized views and query rewriting, e.g., [18], parallel processing and data partitioning, e.g., [13,34,42] as well as advanced indexing. The research on indexing resulted in multiple index structures. From these structures, the ones, which are successfully applied in commercial DBMSs, include: join indexes, e.g., [43], bitmap indexes, e.g., [32,40], bitmap join indexes, e.g., [3,33], and various multidimensional indexes, like for example R-tree [19], Quad-tree [11], and K-d-b-tree [35] (cf. Sections 2 and 7).

None of the aforementioned indexes provide an ultimate mean for analytical query optimization. The indexes do not reflect hierarchical structures of dimensions and, as a consequence, do not support well queries that compute aggregates at different levels of granularity. Moreover, they do not exploit the fact that most of the analytical queries analyze data in time. Despite over 20 years of efforts and substantial achievements in indexing DW data, it is still very important and active research and technological development area.

Paper focus, *contribution*, *and content*: In this paper we focus on index data structures providing means for optimizing star queries. This work is based on and substantially extends our previous works on this topic. In [6] we proposed a hierarchical index, called *HOBI*, for indexing hierarchical dimensions. With the support of experimental evaluations we showed that for star queries *HOBI* offers better performance than the Oracle native bitmap join index. For some patterns of star queries, the performance improvement that we measured in the worst and best cases equaled to 7% and 41%, respectively. *HOBI* was implemented as an application on top of the Oracle DBMS and as such, its performance was burdened with some additional time overhead. In [7] we proposed the extension of *HOBI*, called *Time–HOBI. Time–HOBI* implicitly includes the *Time* dimension in all the other dimensions. This way, it eliminates joins with the *Time* dimension. Thus, *Time–HOBI* is a more universal data structure and answers a broader class of analytical queries than *HOBI* and commercially applied indexes. In [7] we showed how *Time–HOBI* supports just one class of queries and we evaluated the efficiency of the index only for this class of queries.

In this paper we extend the work on *Time–HOBI* as follows:

- first, we analyze how commercial DBMSs (IBM DB2 and Oracle 11g) process star queries and next, we demonstrate how query execution plans for star queries can profit from *Time–HOBI* and how they can be simplified;
- we propose different implementations of *Time–HOBI*;
- we show in details the efficiency analysis of *Time–HOBI* for different classes of star queries, as compared to *HOBI* and a traditional bitmap index;
- we show, based on experiments, how sensitive *Time–HOBI* is to the variable selectivity of analytical queries and to query predicates defined in a variable number of dimensions;
- we evaluate experimentally the creation and maintenance times of *Time–HOBI*.

This paper is organized as follows. Section 2 presents basic definitions used in this paper. Section 3 overviews an example DW schema and star queries that are used throughout the paper. Section 4 presents star query execution plans obtained from commercial DBMSs. Section 5 presents the concept of *Time–HOBI*, its application to star query optimization, and implementation issues. Section 6 discusses performance characteristics of *Time–HOBI* obtained from multiple experimental evaluations. Section 7 presents related work in the area of indexing. Finally, Section 8 summarizes and concludes the paper as well as presents future research directions.

2. Basic definitions

2.1. DW model and schema

In order to support various analyses, data stored in a DW are represented in the so-called *multidimensional data model* [20,22]. In this model an elementary information being the subject of analysis is called a *fact*. It contains numerical features, called *measures*, that quantify the fact. Values of measures are analyzed in the context of *dimensions*. Dimensions often have a hierarchical structure composed of levels, such that $L_i \rightarrow L_j$, where $\rightarrow$ denotes hierarchical assignment between a lower level L_i and upper level L_j, also known as a roll-up or an aggregation path [28]. Following the aggregation path, data can be aggregated along a dimension hierarchy. Level data are called *level instances*. Hierarchically connected level instances form a *dimension instance*.

The multidimensional model is often implemented in relational OLAP servers (ROLAP) [5], where fact data are stored in a *fact table*, and level instances are stored in *dimension level tables*. In a ROLAP implementation two basic types of conceptual schemas are used, i.e., a star schema and a snowflake schema [5]. In the *star schema*, each dimension is composed of only one (typically denormalized) level table. In the *snowflake schema*, a dimension is composed of multiple normalized level tables connected by foreign key–primary key relationships. The two basic DW conceptual schemas can be used for creating other schemas, e.g., a starflake schema. In this schema some dimensions are composed of normalized and some of denormalized level tables.

2.2. Star queries

Based on these various schemas, analytical queries are executed in a DW. As mentioned earlier, such queries typically join multiple tables, filter and sort data, as well as aggregate data at different levels of dimension hierarchies. Typically, the queries join a fact table with multiple level tables and are called *star queries*. In our work we distinguish two classes of star queries, namely *filtering star queries* and *roll-up star queries*. Queries from the first class select data based on predicates