Reducing Query Processing Time for Non-Synonymous Materialized Queries With Differed Criteria

Sonali Ashish Chakraborty, GLS University, Ahmedabad, India
Jyotika Doshi, GLS University, Ahmedabad, India

ABSTRACT

Results of OLAP queries for strategic decision making are generated using warehouse data. For frequent queries, processing overhead increases as same results are generated by traversing through huge volume of warehouse data. Authors suggest saving time for frequent queries by storing them in a relational database referred as MQDB, along with its result and metadata information. Incremental updates for synonymous materialized queries are done using data marts. This article focusses on saving processing time for non-synonymous queries with differed criteria. Criteria is the query condition specified with ‘where’ or a ‘having’ clause apart from equijoin condition. Defined rules will determine if new results can be derived from existing stored results. If criteria of fired query are a subset of criteria in stored query, results are extracted from existing results using MINUS operation. When criteria are a superset of stored query criteria, new results are appended to existing results using the UNION operation.

KEYWORDS
Criteria Queries, Data Mart, Data Warehouse, Differed Criteria, Materialized Queries, MINUS Operation, OLAP Queries, Processing Time, UNION Operation

1. INTRODUCTION

Huge amount of enterprise data is stored in data warehouse. For strategic decision making, OLAP (Online Analytical Processing) queries are executed on warehouse data. Results are generated after traversing through warehouse records. For frequent ad hoc OLAP queries, same results are generated by accessing warehouse data every time they are invoked. This is quiet time consuming and results in increase of processing overhead.

Authors suggest saving query processing time and reducing processing overhead by storing executed OLAP queries with its details in a separate relational database; MQDB (Materialized Query Database). The details include query results and metadata information such as timestamp, frequency, threshold, number of records in result file, path of result file and related data mart. This eliminates the need to access data warehouse for frequent OLAP queries.

For a fired query, its synonymous query is searched in MQDB. Two queries can be called synonymous to each other if they are fired on same set of tables, fields, functions and criteria values. For a synonymous query requiring no incremental updates, stored results are fetched from MQDB. This is followed by updating query metadata information. This significantly reduces the query result retrieval time (S. Chakraborty, Performance Evaluation of Materialized Query, 2018).

DOI: 10.4018/IJNCR.2019040104
For synonymous queries requiring incremental updates, incremental data is processed either from updated warehouse data (S. Chakraborty, Materialized Queries with Incremental Updates, 2018) or using data marts (S. Chakraborty, Incremental Updates using Data Warehouse versus Data Marts, 2018) to achieve significant reduction in query processing time (S. Chakraborty, Incremental Updates using Data Warehouse versus Data Marts, 2018). Incremental and existing stored results are appended and result file is updated.

In this paper, authors attempt to save query processing time and reduce processing overhead for non-synonymous queries differing in their criteria values. Criteria refer to the conditions used in query other than equijoin conditions specified with either a ‘where’ or a ‘having’ clause.

For example:

If stored query criteria is “town_name = {Srinagar}”

Fired query criteria is “town_name = {Srinagar, Jammu}” Then the query differs with stored query in criteria. They are referred as non-synonymous queries with differed criteria.

It can be advantageous if new results for fired query can be derived from existing stored results. This reduces the need to access data warehouse repeatedly for non-synonymous queries which differ only in their criteria values.

If the intersection of fired query criteria and stored query criteria is NULL, then considering as a new query, results are generated from data warehouse. For non-NULL intersection of criteria values, fired query and stored query criteria will either be a subset or a superset of each other. If the criterion of fired query is a subset of criteria of stored query, then results can be extracted using MINUS operation. If given criteria is superset of criteria of stored query then new results are appended to existing results using UNION operation.

2. LITERATURE REVIEW

For faster query result retrieval, multidimensional cubes and their issues are discussed in (Vassiliadis, 1998), (P. Vassiliadis, 1999), (Harinarayan, 1996; Colossi, 2002; Datta, 1999; Agrawal; Deshpande, n.d.; Shanmugasundaram, n.d.; Li, n.d.) are used to store aggregate results from data warehouse whereas materialized views and their maintenance is discussed in (Gupta, 1993; Gupta, n.d.; Zhuge, 1995; Gupta, 1996; Ross, 1996) are used to store results of OLAP queries executed on data warehouse.

Authors (Bara, 2008) suggest various mechanisms for optimizing cost of data extraction from a data warehouse. Metrics based on organization’s needs for designing and maintaining high quality data warehouse are defined in (Serranoa, 2007). Storing queries with results in cache memory (F. Sultan, 2010) and use of right index structures (Vanichayobon, n.d.) also plays a major role in enhancing the performance of queries.

Authors (Theodoratos, 1997) suggested that by storing the results of all the queries of interest in data warehouse, high query performance can be obtained. (Karthik, 2012) discuss the ways of tuning an SQL query such that it decreases the time consumed by the query during runtime. Authors (O’Niel, 1997) present a review of indexing technology used to speed up queries in data warehouse environments. (Goldstein, 2001) presented a fast and scalable algorithm to determine if a part or all of a query can be computed from materialized view.

(Han, n.d.) Data marts; subset of data warehouse are created after gathering user requirements and formulating them into star join schema structure. Their requirements are based on enterprise size and user requirements. Author (Bonifati, 2001) proposed a three-step method for designing data marts while literature (Abdelhedi, 2011) discusses a hybrid driven approach for designing data mart. (Nabli, 2005) proposed an algorithm for automatically transforming OLAP requirements into data marts while (Alijanabi, 2013) uses Ralph Kimball’s methodology for developing a query dispatching tool facilitating access to information within data marts, eventually data warehouse in a fast and an organized way.
17 more pages are available in the full version of this
document, which may be purchased using the "Add to Cart"
button on the product's webpage:

www.igi-global.com/article/reducing-query-processing-time-
for-non-synonymous-materialized-queries-with-differed-
criteria/225824?camid=4v1

This title is available in InfoSci-Journals, InfoSci-Journal
Disciplines Computer Science, Security, and Information
Technology, InfoSci-Journal Disciplines Medicine,
Healthcare, and Life Science, InfoSci-Physical Sciences,
Biological Sciences, and Engineering eJournal Collection,
InfoSci-Healthcare Administration, Clinical Practice, and
Bioinformatics eJournal Collection. Recommend this product
to your librarian:

www.igi-global.com/e-resources/library-
recommendation/?id=2

Related Content

Applications of Neural Networks in Supply Chain Management
www.igi-global.com/chapter/applications-neural-networkssupply-chain/21154?camid=4v1a

Multi-Objective Evolutionary Algorithm NSGA-II for Variables Selection in Multivariate Calibration Problems
www.igi-global.com/article/multi-objective-evolutionary-algorithm-nsga-ii-for-variables-selection-in-multivariate-calibration-problems/93623?camid=4v1a
Particle Swarms: Optimization Based on Sociocognition
www.igi-global.com/chapter/particle-swarms-optimization-based-sociocognition/28330?camid=4v1a

Object-Oriented Evolutionary Testing: A Review of Evolutionary Approaches to the Generation of Test Data for Object-Oriented Software
www.igi-global.com/article/object-oriented-evolutionary-testing/119691?camid=4v1a