Journal of Engineering and Applied Sciences 12 (Special Issue 12): 9436-9439, 2017
ISSN: 1816-949X

Materialized View Based Data Warehouse Optimization

Ashwini A. Rangari and A.R. Raipurkar
Department of Computer Science and Engineering,
Shri Ramdeobaba College of Engineering and Management, 440013 Nagpur, India

Abstract: Traditional data warehouse systems are only periodically updated. It cannot satisfy the need of real-time analysis. To diminish this problem, the Real-Time Data Warehouse (RTDW) technology has emerged. The real-time data warehouses are very suitable for the real-time enterprises. In spite of the high complexity of the queries used, quick response time is the basic requirement of successful business applications. Materialized views are found to be very useful for fast query processing. In this study, the purpose is to better manage materialized views. In our approach, we are performing data view maintenance not only on the basis of data staleness but also on query time and query size. This will ensure better view maintenance as well as a better way to update the data in the views. We use an update policy that dynamically decides the method to maintain materialized views based on their access frequency.

Key words: RTDW, materialized views, materialize views, view maintenance, staleness, dynamic management

INTRODUCTION

Traditionally data warehouses do not contain very recent data. The demand for fresh data in data warehouses has always been an essential requirement. For example, airlines and government agencies require the most recent information when trying to detect suspicious groups of passengers or potentially illegal activity. Financial institutions need precise and up-to-date analytical reports regarding stocks. Real time data warehousing refers to a new trend where DWs are updated as frequently as possible due to the high demand of fresh data. It continually loads incoming data to support time-critical analysis. One of the most effective optimization techniques used in RTDWs is materialized views. Materialized views can be used to pre-compute and store aggregated data such as sum of sales. They can also be used to pre-compute joins with or without aggregations. So, a materialized view is used to eliminate overhead associated with expensive joins or aggregations for a large or important class of queries. A view is defined as a function from a set of base tables to a derived table and the function is recomputed every time the view is referenced. On the other hand, a materialized view is like a cache, i.e., a copy of data that can be accessed quickly. Materialized views are of three types.

Materialized views with aggregates, materialized views containing only joins and nested materialized views: When the distant fundamental data source changes, the materialized views in data warehouse are also updated in order to maintain the consistency and convergence, this causes the need for handling the problem of materialized view maintenance. The required freshness of data determines the frequency of the re-reconstruction of the materialized view. Although, the reason for using materialized views is to enhance performance, the overhead related with materialized views management can become a consequential system management problem when the data source changes are recurrent. So, maintaining a view is of the most important task in warehousing environment.

EXISTING APPROACHES FOR MAINTAINING VIEWS

Various approaches have been introduced for maintaining the view in a warehouse environment. Golab *et al*. (2009), the researchers define the concept of materialized view staleness in the context of RTDWs but they don't deal with how to select and to maintain materialized views. Colby *et al*. (1997) and Bellatreche *et al*. (2004), various update policies to refresh materialized views are proposed. The update policies are described:

Immediate view maintenance: The view is refreshed within the same transaction that updates the basic source tables. Although, this method inflicts a consequential overhead in RTDW because of the update frequency, the view is always up-to-date with fresh data.

Periodic view maintenance: The view is refreshed periodically, e.g., once a day. Nonetheless in RTDW, periodic maintenance is inefficient.

Corresponding Author: Ashwini A. Rangari, Department of Computer Science and Engineering,
Shri Ramdeobaba College of Engineering and Management, 440013 Nagpur, India

On-demand updates: Changes made in the fundamental data-source are propagated in a delayed style. Thus, the view is refreshed when it is queried.

Kotidis and Roussopoulus (1999), a dynamic view management system for data warehouses known as DynaMat has been proposed. It includes two different phases: throughout the 'On-line' phase, the purpose is to respond to as many queries as possible. The second phase is called the 'Update' in which updates received from the data sources are stored in the data warehouse and materialized results get refreshed, queries are not allowed throughout this phase. So, it does not satisfy the requirements of the users in RTDW that should be refreshed periodically.

Hamdi *et al.* (2014), a Dynamic Selection of materialized Views algorithm (DynaSeV) is proposed which selects views from results of incoming queries under the execution time and the storage space constraints of the system. But this does not consider the frequency factor of the queries.

Recent research in RTDW has mostly dealt with the problem of scheduling of updates (Leng *et al.*, 2011). Zhou *et al.* (2010), the incremental view maintenance techniques variation is given in terms of minimum incremental view maintenance. The principle of incremental view maintenance is that data source reports its changes to integrator who then calculates corresponding changes and inform the database with corresponding changes.

Yeung and Gruver (2005), researchers develop a multiagent system that achieves Immediate Incremental View Maintenance (IIVM) for continuous updating of data warehouse views. An IIVM system is described that processes updates as transactions are executed at the underlying data sources to eliminate view maintenance downtime for the data warehouse a crucial requirement for internet applications. The use of a multiagent framework provides considerable process speed improvement when compared with other IIVM systems.

Harinarayan *et al.* (1996) presented a Greedy algorithm for the selection of materialized views so that query evaluation costs can be optimized in the special case of "data cubes". However, the costs for view maintenance and storage were not addressed in this piece of research.

Ross *et al.* (1996) investigate the problem of incremental maintenance of an SQL view, since, the incremental checking of integrity constraints is known to be essentially equivalent to the view maintenance problem. Since, it costs a great deal to rebuild the materialized views, incremental view maintenance policy as an efficient alternative has been proposed and extensively studied by Hung *et al.* (2007) and Hanson (1987).

PROPOSED RESEARCH

In the existing research, the researchers have proposed a technique to create and update materialized views based on the staleness of the data update. But this approach is limited and has no consideration about the query size and query time. In our approach we are performing data view maintenance not only on the basis of data staleness but also on query time and query size. This will ensure better view maintenance as well as a better way to update the data in the views. We will be evaluating the performance analysis of both the techniques in order to get the comparative analysis.

Working of our system: We have classified our research in the following steps and presented a flow diagram of the system in Fig. 1:

Result evaluation steps: Comparison of approaches from 4) and 5) with result evaluation are:

1. Collection of database queries and databases for testing
2. Query creation for materialized view generation
3. Evaluation of query frequency, size and delay for the view
4. Application of threshold based algorithm for view maintenance with only query frequency
5. Application of threshold based algorithm for view maintenance with query frequency, delay and size

Constraints involved in the selection of materialized views

Criticality level: When the data criticality level of the data i (denoted C_i) is equal to 0, the data are considered as non critical data.

Storage space constraints: It is necessary to minimize the storage space used to implement materialized views. Similar to researchers by Leng *et al.* (2011) who allow users to express their real needs for their queries by specifying the acceptable result staleness when queries are submitted; we define an acceptable staleness for materialized views. The staleness f the data is calculated as follows:

$$\text{Staleness} = \alpha f+\beta(1-t)+\Delta(1-s)$$

Where:

f = Frequency
t = Time
s = Storage (bytes)

And α, β and Δ are pre-defined constants, so as to minimize the time and storage.

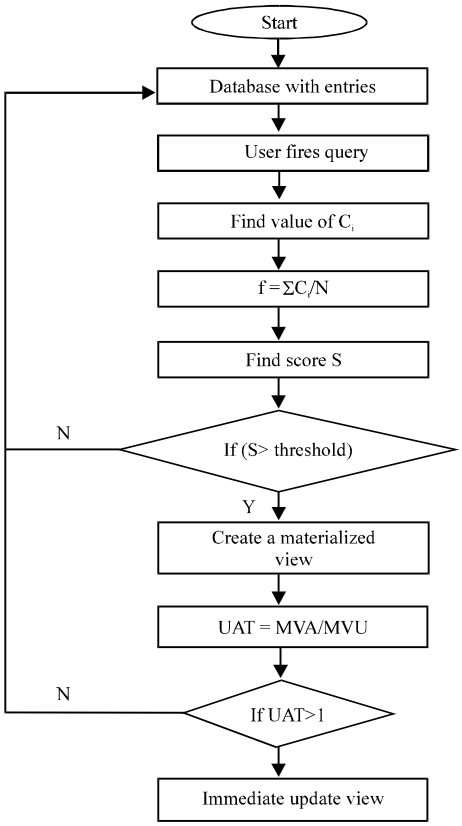

Fig. 1: Data flow

Execution time constraints: It is important to minimize the execution time. A value Update Adaptation Threshold (UAT) for selecting a right update policy for a materialized view. The UAT is assigned to a number between zero and one. Where, MVA refers to the number of times the materialized view accessed and MVU refers to the number of times the materialized view updated. When the value of UAT equals 1, this means that the materialized view is accessed at least as frequently as it is updated. A materialized view will be immediately updated if it has an UAT≥1:

$$UAT = \frac{MVA}{MVU}$$

Selection of candidate queries: In order to identify the candidate queries for the materialized view, the score of each query is calculated as follows:

$$S = \frac{C}{max.cost} * \alpha + \frac{f}{max.freq.} * \beta + \frac{delay}{max.delay} * \gamma$$

Where:

α = Weight of cost
β = Weight of query
γ = Weight of delay

The cost, frequency and delay factor (response time) for each query is divided by the maximum cost, maximum frequency and maximum delay respectively. This gives us the unit-less quantities to be added to calculate the score of the query. The maximum score is multiplied by the weight of threshold (Δ) and the result of the multiplication is the query threshold. If the score of a query is greater than the query threshold, then create the materialized view, otherwise do not create the view.

CONCLUSION

The aim of our research is to enhance the system performance in RTDW to ensure the quality of data by making sure that all materialized views are refreshed properly each time and to abstain from the overloading of the system. System will give better view maintenance than the existing research which is based only on query frequency. We will measure the quality of view maintenance based on the time needed by the system to give response to a particular query repeatedly and compare it with the existing system.

We have proposed a hybrid parameter based technique for evaluation of materialized views creation and it gives better efficiency when compared to the previous one parameter based techniques.

REFERENCES

Bellatreche, L., M. Schneider, H. Lorinquer and M. Mohania, 2004. Bringing together partitioning, materialized views and indexes to optimize performance of relational data warehouses. Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'04), September 1-3, 2004, Springer, Zaragoza, Spain, pp: 15-25.

Colby, L.S., A. Kawaguchi, D.F. Lieuwen, I.S. Mumick and K.A. Ross, 1997. Supporting multiple view maintenance policies. Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, May 11-15, 1997, ACM, Tucson, Arizona, ISBN:0-89791-911-4, pp: 405-416.

Golab, L., T. Johnson and V. Shkapenyuk, 2009. Scheduling updates in a real-time stream warehouse. Proceedings of the 2009 IEEE 25th International Conference on Data Engineering (ICDE'09), March 29-April 2, 2009, IEEE, Shanghai, China, ISBN:978-1-4244-3422-0, pp: 1207-1210.

Hamdi, I., E. Bouazizi and J. Feki, 2014. Dynamic management of materialized views in real-time data warehouses. Proceedings of the 6th International Conference on Soft Computing and Pattern Recognition (SoCPaR'14), August 11-14, 2014, IEEE, Tunis, Tunisia, ISBN:978-1-4799-5935-8, pp: 168-173.

Hanson, E.N., 1987. A performance analysis of view materialization strategies. Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data Vol. 16, May 27-29, 1987, ACM, San Francisco, California, ISBN:0-89791-236-5, pp: 440-453.

Harinarayan, V., A. Rajaraman and J. Ullman, 1996. Implementing data cubes efficiently. ACM SIGMOD Record, 25: 205-216.

Hung, M.C., M.L. Huang, D.L. Yang and N.L. Hsueh, 2007. Efficient approaches for materialized views selection in a data warehouse. Inf. Sci., 177: 1333-1348.

Kotidis, Y. and N. Roussopoulos, 1999. DynaMat: A dynamic view management system for data warehouses. Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, May 31-June 03, 1999, ACM, Philadelphia, Pennsylvania, ISBN:1-58113-084-8, pp: 371-382.

Leng, F., Y. Bao, G. Yu, J. Shi and X. Cai, 2011. Requirement-based query and update scheduling in real-time data warehouses. Proceedings of the 12th International Conference on Web-Age Information Management (WAIM'11), September 14-16, 2011, Springer, Wuhan, China, pp: 379-389.

Ross, K.A., D. Srivastava and S. Sudarshan, 1996. Materialized view maintenance and integrity constraint checking: Trading space for time. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, June 04-06, 1996, ACM, Montreal, Quebec, Canada, ISBN:0-89791-794-4, pp: 447-458.

Yeung, G.C. and W.A. Gruver, 2005. Multiagent immediate incremental view maintenance for data warehouses. IEEE. Trans. Syst. Man Cybern. Part A Syst. Hum., 35: 305-310.

Zhou, L., Q. Shi and H. Geng, 2010. The minimum incremental maintenance of materialized views in data warehouse. Proceedings of the 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR'10) Vol. 3, March 6-7, 2010, IEEE, Wuhan, China, ISBN:978-1-4244-5192-0, pp: 220-223.