An Approach for Query Optimization by using Schema Object Base View

Dhaval Patel
Research Scholar, CSE Department
Parul Institute of Technolody,
Limda, Vadodara, India.

Pratik Patel
Asst. Prof., CSE Department
Parul Institute of Technolody,
Limda, Vadodara, India.

ABSTRACT
Mining of Data is the extraction of hidden prognosticative information from large databases or set of data, is a strong new technology with great prospective to help companies focus on the most important information in their data base. Query optimization is a purpose of many relational database management systems. The query optimizer experiments to dictate the most efficient way to implement a given query by examining the possible query plans. There are different techniques is given for optimizing query using schema based and materialized views in data base namely- Query Graph, Tableaus, Optimization of Queries having Aggregates. In this paper we are using Different query optimization parameter and create an effective approach by using this approach we are reduce query execution cost, query space and more effectible for the query.

The complexity of Queries severely increase the execution cost of the queries and have a critical effect on performance and productivity of decision support systems. It is required to perform expensive join and aggregation operations frequently on the databases. Now if they are not pre calculated in advanced then it leads to reduce query performance. Schema object improve query performance by pre calculating expensive join and aggregation operations on the database prior to execution and storing the results in the database. Schema object define not only relationships, but also allow you to recompute expensive joins and aggregations which lead to optimized query performance in possible ways. Schema object leads to the decrease Query processing cost and Query Maintenance cost in terms of Time factor. Schema object improve query performance by pre calculating expensive join and aggregation operations on the database prior to execution and storing the results in the database. The big advantage of a Schema object based views is extremely fast retrieval of aggregate data, since it is precomputed and stored, at the expense of insert/update/delete so that it increase query performance than the ordinary view and table. Schema object based view is also called Materialized view.

Keywords

Query optimization, materialized view, Schema object base view

1. INTRODUCTION
Predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions is tooled by Data mining[1]. The automated, potential inspects recommended by data mining move beyond the analyses of past occurrences provided by backward-looking tools typical of decision support systems. Massive quantities of data already are collected and refined by most companies. Data mining techniques can be implemented promptly on existing software and hardware platforms to strengthen the value of existing information resources, and can be integrated with new products and systems as they are brought on-line.

Query optimization is a consequence of many relational database administration organizations. Query Optimization is the procedure of choosing the most systematic technique to accomplish a SQL statement. When the cost-based optimizer was provided for the first time with Oracle7, Oracle supported only standard relational data[2]. The introduction of objects enlarged the maintained data types and functions. The aim is to attempt them all out, but it requires deciding in what order. What interchange of tastes will maximize the comprehensive fulfillment of palate? Although much less pleasurable and instinctive, that is the type of problem that query optimizers are called to interpret. Given a query, there are many programs that a database management system (DBMS) can track to procedure it and manufacture its answer. All programs are identical in terminology of their final output but different in their value, i.e., the amount of time that they require to pass.

2. QUERY FLOW
The first step in processing a query submitted to a DBMS is to convert the query into a form usable by the query processing engine. High- level query languages such as SQL represent a query as a string, or sequence, of characters. Certain sequences of characters represent different types of tokens such as keywords, operators, operands, literal strings, etc[1][2].

The primary job of the parser is to extract the tokens from the raw string of characters and translate them into the corresponding internal data elements for example relational algebra operations and operands and structures for example query tree, query graph. The last job of the parser is to check the validity and syntax of the actual query string[3].

In second stage, the query processor applies instructions to the internal data structures of the query to convert these structures into complement, but more efficient representations. The rules can be based upon mathematical models of the relational algebra expression and heuristics, upon cost approximates of different algorithms or techniques applied to operations or upon the semantics within the query and the relations it necessitates[4][5]. Selecting the absolute rules to apply, when to apply them and how they are applied is the function of the query optimization engine.

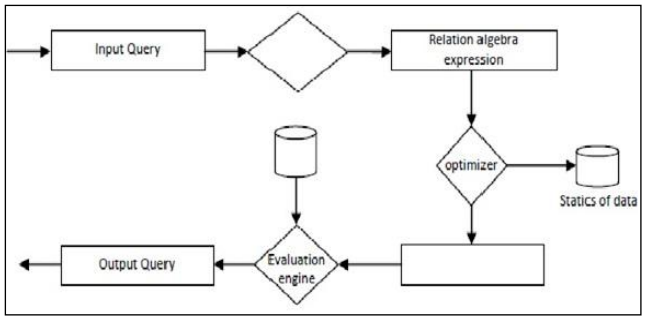


Figure 1: Steps in query processing

The final step in processing a query is the evaluation phase. The best estimation plan candidate generated by the optimization engine is chose and then accomplished. Besides processing a query in a simple sequential manner, some of a query's individual operations can be processed in parallel either as unconventional procedures or as interdependent pipelines of procedures or threads[6].

3. PROBLEM STATEMENT

With the dincreasing complexity of queries severely increase the execution cost of the queries and have a critical effect on performance and productivity of decision support systems , which increases time factor for database scanning as well time factor for execution of it which degrades the performance of queries.

In order to overcome these limitations it must require a schema object based view as our proposed approach for extremely fast retrieval of aggregate data, since it is precomputed and stored, at the expense of insert/update/delete so that it increase query performance than the ordinary view and table.

ANALYSE AN APPROACH BY EXAMPLE:

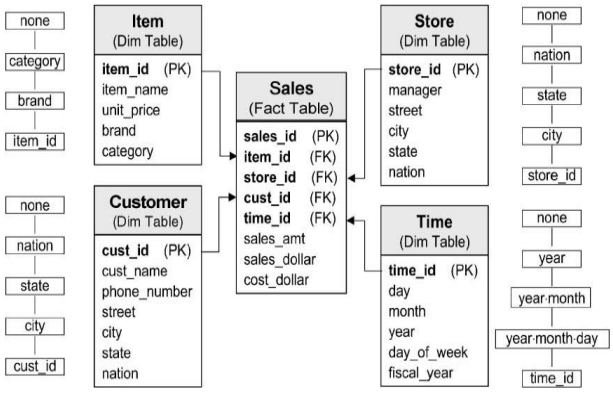


Figure 2: Analyse an Approach By Example[8]

We consider the following OLAP query Q1**,** which asks for the total sales of the stores in the USA or Canada from 1996 to 1999 by state and year.

Q1: SELECT state, year, SUM(sa1es-dollar)

FROM Sales, Store, Time

WHERE Sales.storeid = Store.storeid AND Sales.timeid = Time.timeid

AND (Store.nation = 'USA' OR Store.nation = 'Canada') AND Time.year *2* 1996 AND Time.year <= 1999 GROUP BY state ,Year

rw[Q1] has three query blocks, whose results are combined by union. Each query block contains a different MV and computes a part of the aggregate groups of Q1.

Specifically, the first query block computes from MVl the total sales of the stores in the USA or Canada from 1997 to 1999 by state and year. The second and the third one use MV2 and MV3 respectively to compute the total sales of the stores in the USA and Canada in 1996 by state. Since the three sets of groups are disjoint and the union of them is equal to the set of groups computed by Q1, we can obtain the same result of Q1 by taking the union of them as in rw[Q1].[9][10]

The above query Q1 leads to the following drawbacks :

- It directly uses on Fact Table Sales which contains a large amount of data in a complex structure so it required more time for database scanning
- It will not support recomputation of join and aggregation efficiently in order to get data with multi dimensional
- Using of Fact table decreases the performance of OLAP query.

4. WORKFLOW OF APPROACH

In order to achieve our goal of Optimizing performance of queries, OLAP queries must be rewritten using Materialized view and Dimension Hierarchies Lattice. This chapter focuses proposed work and Implementation strategies.

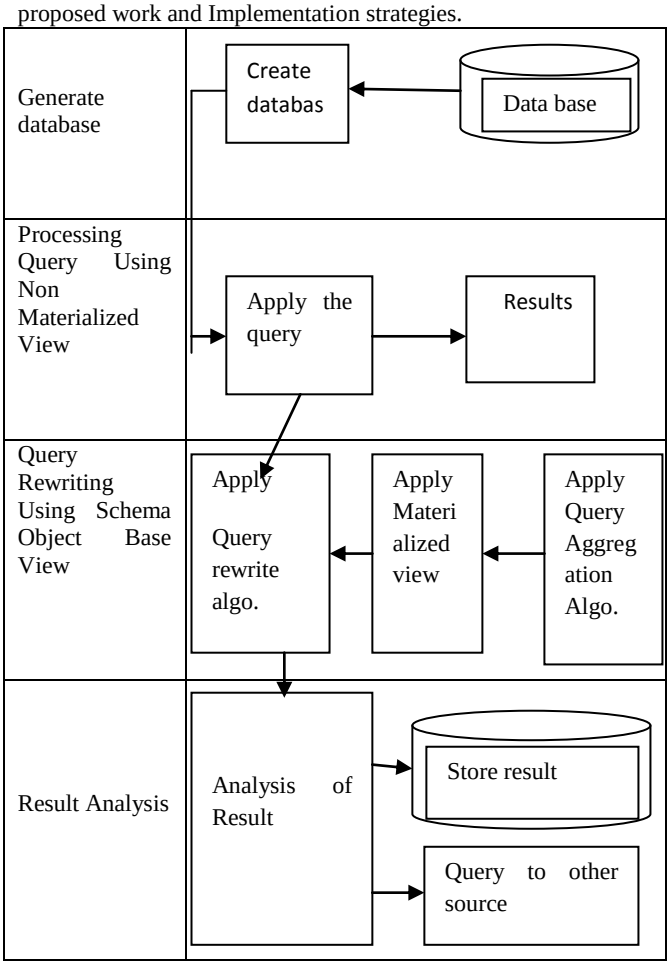


Figure 3: Work flow Approach

1. Creating Database
2. Processing Query Using Non Materialized View and Results analysis.

3. Apply The Query Rewriting Method using Materialized view to optimize query
4. Result Analysis after Applying Materialized view

5. IMPLEMANTATION AND RESULT

We are using different parameter for query optimization like as CPU cost, Input Output cost, Elimination of duplication data. We are work in cluster index by using this cluster index we are create Normal view and we are execute the query after execution we are analysis the query execution cost and its input and output cost. We are also analysis the query performance and query execution estimation cost. By using non schema object base view here is the different result for the query optimization.

5.1 Approx. CPU cost of Non-Schema object based view

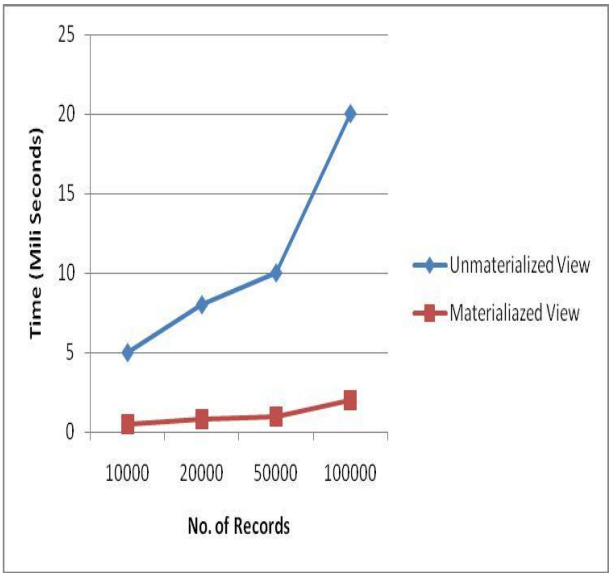
Table 1: CPU cost

No. of Records	Non materialized View
10000	5
20000	8
50000	10
100000	20

In cluster index we are creating a database. We are using non-materialize view (Non schema object base view) and fire the query and analysis the estimation for this query. We are also analysis the CPU estimation cost.

In this table we are enter the 10000, 20000, 50000 and 100000 records then its estimation cost is 5, 8, 10 and 12.It is the approx. CPU cost of non schema object base view.

5.1.1 Performance analysis of Non-Materialized view & Expected Materialized View in terms of CPU cost



Here display the query performance in graph .In this graph we are display the without using materialize view query performance graph. We are using non- materialize view and analysis the result then performance graph is increase and I expect we are using materialize view then query performance graph is decrees.

In this graph blue line is display the using non- materialize view Query performance CPU cost and red line is display Expected Materialized View in terms of CPU cost.

5.2 Approx. IO cost of Non-Schema object based view

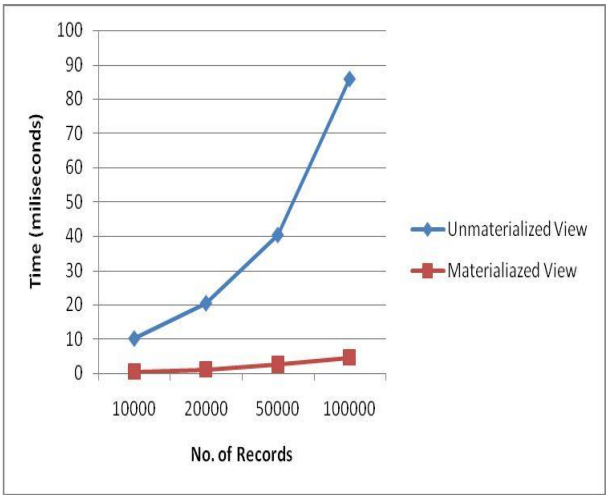
Table 2: Input/output cost

No. of Records	Non materialized View
10000	10.1
20000	20.4
50000	40.3
100000	85.73

In cluster index we are creating a database. We are using non-materialize view (Non schema object base view) and fire the query and analysis the estimation for this query. We are also analysis the input / output cost.

In this table we are enter the 10000, 20000, 50000 and 100000 records then its estimation cost is 10.1, 20.4, 40.3 and 85.73.It is the approx. Input / Output cost of non schema object base view.

5.2.1 Performance analysis of Non-Materialized view & Expected Materialized View in terms of IO cost



Here display the query performance in graph .In this graph we are display the without using materialize view query input / output cost in graph. We are using non- materialize view and analysis the result then performance graph is increase and I expect we are using materialize view then query input output cost is decrees.

In this graph blue line is display the using non- materialize view Query performance input / output cost and red line is display Expected Materialized View in terms of input / output cost.

6. CONCUSION AND FUTURE WORK

We proposed a new approach to rewrite a given query using schema object based existing in databases. We presented conditions for usability of MVs in rewriting queries and proposed a rewriting algorithm consisting of three main steps. In the first step, it selects MVs that will be used in rewriting and determines query regions for them. Previous approaches focus on optimization of query using aggregation and also works on single block of query while here we present usability of schema object as addition with existing work and will also works on multi block query. In the second step, it generates query blocks for the selected MVs using their query regions. The last step integrates the query blocks into a final rewritten query It utilizes a much broader class of MVs and yields more general types of rewritings than other previous approaches can do.

Future work involves the implementation of proposed approach and analyze the comparisons with the existing work .In this our future plans include extending the proposed rewriting method to deal with more general and complex queries and integrating the method with the process of query Optimization.

7. REFERENCE

[1] Amol Deshpande, Lisa Hellerstein "Flow Algorithms for Parallel Query Optimization" IEEE 2008.

[2] Joshi Janki, "An Analysis on Query Optimization in Distributed Database" International Journal of Modern Trends in Engineering and Research 2014.

[3] T.Nalini, Dr. A.Kumaravel, Dr.K.Rangarajan "A comparative study analysis of materialized view for selection cost" International Journal of Computer Science & Engineering Survey (IJCSES) Vol.3, No.1, February 2012.

[4] A. Lee, A. Nica, and E. Rundensteiner, "The EVE approach view synchronization in dynamic distributed environments", In IEEE Transactions and Data Engineering, 14, 2002.

[5] Garima Thakur, Anjana Gosain " A Comprehensive Analysis of Materialized Views in a Data Warehouse Environment" (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 5, 2011

[6] Deepika Kirti, Jaspreeti Singh "Assortment of Materialized View: A Comparative Survey in Data Warehouse Environment"International Journal of Computer Applications (0975 – 8887) Volume 96– No.7, June 2014

[7] Ravindra N. Jogekar, Ashish Mohod "Design and Implementation of Algorithmsfor Materialized View Selection and Maintenance in Data Warehousing Environment" International Journal of Emerging Technology and Advanced Engineering Volume 3, Issue 9, September 2013

[8] S. Chen, X. Zhang, and E. Rundensteiner, "A compensation based approach for view maintenance in distributed environments", In IEEE transactions and data engineering, 18, 2006.

[9] Madhu Bhan, T.V.Suresh Kumar, K.Rajanikanth "Materialized view size estimation using sampling" IEEE International Conference on Computational Intelligence and Computing Research 2013 IEEE

[10] Lijuan Zhou1,Min Xu ,Qian Shi ,Zhongxiao Hao , "Research on Materialized Views Technology in DataWarehouse" Beijing Educational Committee science and technology development plan project 2010.