Indian J.Sci.Res. 17(2): 54 - 57, 2018

ISSN: 0976-2876 (Print)
ISSN: 2250-0138(Online)

QUERY OPTIMIZATIONIN DATA WAREHOUSE AND DATA MINING : A CASE STUDY USING DWH AND DATA MINING TECHNIQUES

[1]P.Arpitha, [2] Dr. P V Kumar, [3] S.VijayaSree

[1] Department Of Computer Science, Rayalaseema University, Kurnool.
[2] Department Of computer Science and Engineering, Osmania University, Hyderabad
[3]Department Of MCA, Aurora's PG College, Moosarambagh, Hyderabad

***Abstract*-**Query optimization is the bottleneck of database application performance especially those which store history i.e. data warehouse. Numerous researches has been introduced in the area of optimizing query performance, however a lot of research focused on online transaction processing (OLTP) database applications rather than data warehouse applications. SQL is used as query language because most data warehouses are based on relational or extended relational database system. As the information requests of the users are likely to be very complex. In order to reduce the complexity of the query generation process and in order to preserve portability to other database systems proposed semantic query optimization architecture is very useful. First, we offer some guiding principles for query optimization: Query optimization is of great importance for the performance of a relational database, especially for the execution of complex SQL statements. A query optimizer determines the best strategy for performing each query. The query optimizer chooses, for example, whether or not to use indexes for a given query, and which join techniques to use when joining multiple tables. These decisions have a tremendous effect on SQL performance, and query optimization is a key technology for every application, from operational Systems to data warehouse and analytical systems to content-management systems. In present scenario data warehouses & mining turned out to be the common basis for the integration and analysis of data in modern enterprises. Data mining based applications are used to analyze data on the data base **.**using different data mining techniques to optimize the query in data warehouse(OLAP).

Keywords: Data warehouse , data mining, query optimizer,OLTP,SQL,OLAP.

I.Introduction

In today's era information and communication technology, every organization builds and maintains their vital data and information using powerful data base system that offers better performance and availability .apart from the typical transaction processing systems heavily for data warehousing and data mining to support different information systems of an organizations like decision support system ,executive support system, expert system and business intelligence systems that must run 24*7 around the globe.

II. Need And Importance Of Query Optimization

Query optimization is of great importance for the performance of a relational database, especially for the execution of complex SQL statements. A query optimizer determines the best strategy for performing each query. The query optimizer chooses, for example, whether or not to use indexes for a given query, and which join techniques to use when joining multiple tables. These decisions have a tremendous effect on SQL performance, and query optimization is a key technology for every application, from operational Systems to data warehouse and analytical systems to content-management systems. In present scenario data warehouses & mining turned out to be the common basis for the integration and analysis of data in modern enterprises. Data mining based applications are used to analyze data on the data base **.**

2.1 Understand How Your Database Is Executing Your Query

Nowadays all databases have their own query optimizer, and offers a way for users to understand how a query is executed. For example, which index from which table is being used to execute the query? The first step to query optimization is understanding what the database is doing. Different databases have different commands for this. For example, in MySQL, one can use "EXPLAIN [SQL Query]" keyword to see the query plan. In Oracle, one can use "EXPLAIN PLAN FOR [SQL Query]" to see the query plan.

2.2 Retrieve As Little Data As Possible

The more data returned from the query, the more resources the database needs to expand to process and store these data. So for example, if you only need to retrieve one column from a table, do not use 'SELECT *'.

2.3 Store Intermediate Results

[1]**Corresponding Author**

Sometimes logic for a query can be quite complex. Often, it is possible to achieve the desired result through the use of sub queries, inline views, and UNION-type statements. For those cases, the intermediate results are not stored in the database, but are immediately used within the query. This can lead to performance issues, especially when the intermediate results have a large number of rows.

The way to increase query performance in those cases is to store the intermediate results in a temporary table, and break up the initial SQL statement into several SQL statements. In many cases, you can even build an index on the temporary table to speed up the query performance even more. Granted, this adds a little complexity in query management (i.e., the need to manage temporary tables), but the speedup in query performance is often worth the trouble. Below are several specific query optimization strategies.

- **Use Index**

  Using an index is the first strategy one should use to speed up a query. In fact, this strategy is so important that index optimization is also discussed.

- **Aggregate Table**

  Pre-populating tables at higher levels so less amount of data need to be parsed.

- **Vertical Partitioning**

  Partition the table by columns. This strategy decreases the amount of data a SQL query needs to process.

- **Horizontal Partitioning**

  Partition the table by data value, most often time. This strategy decreases the amount of data a SQL query needs to process.

III.Objectives

The objectives and the purpose of query optimization is faster response to queries. The semantic optimizer knows more about its data rather than user. Therefore it can replace the user's query with a different query which will generate the same result set efficiently in less time. The new query is faster because it will do less work when extracting the selected result tuples from the data base.. we focus on a model of Optimizer Architecture for data Warehousing based (decision support systems) application. According to this model, the application generates a sequence of SQL statements, which is processed by the (OLAP Server) DWDBS.

IV. Methodology

SQL is used as query language because most data warehouses are based on relational or extended relational database system. As the information requests of the users are likely to be very complex. In order to reduce thecomplexity of the query generation process and in orderto preserve portability to other database systems proposed semantic query optimization architecture is very useful.

a)Partitioning:

partitioning is one of the technique used to optimize the performance of the query. Partitioning is done to enhance performance and facilitate easy management of data. Partitioning also helps in balancing the various requirements of the system. It optimizes the hardware performance and simplifies the management of data warehouse by partitioning each fact table into multiple separate partitions.

Partitioning is important for the following reasons –

1)For easy management: The fact table in a data warehouse can grow up to hundreds of gigabytes in size. This huge size of fact table is very hard to manage as a single entity. Therefore it needs partitioning.

2)To assist backup/recovery: we do not partition the fact table, then we have to load the complete fact table with all the data. Partitioning allows us to load only as much data as is

3) To enhance performance: If we do not partition the fact table, then we have to load the complete fact table with all the data. Partitioning allows us to load only as much data as is required on a regular basis. It reduces the time to load and also enhances the performance of the system.

To cut down on the backup size, all partitions other than the current partition can be marked as read-only. We can then put these partitions into a state where they cannot be modified. Then they can be backed up. It means only the current partition is to be backed up.To Enhance Performance by partitioning the fact table into sets of data the query procedures can be enhanced. Query performance is enhanced because now the query scans only those partitions that are relevant. It does not have to scan the whole data.

Database administration (e.g., adding new columns to a table, archiving **data**, recreating indexes, loading tables). **Data partitioning** involves splitting out the rows of a table into multiple tables (i.e., **horizontal partitioning**) or splitting out the columns of a table into multiple tables (i.e., **vertical partitioning**).

b)Data mining

Data Mining is defined as the procedure of extracting information from huge sets of data. In other words, we can say that data mining is mining knowledge from data. The data mining techniques are knowledge discovery, query language, classification and prediction, decision tree induction, cluster analysis.

The Data Mining Query Language (DMQL) was proposed by Han, Fu, Wang, et al. for the DBMiner data mining system. The Data Mining Query Language is actually based on the Structured Query Language (SQL). Data Mining Query Languages can be designed to support ad hoc and interactive data mining. This DMQL provides commands for specifying primitives. The DMQL can work with databases and data warehouses as well. DMQL can be used to define data mining tasks. Particularly we examine how to define data warehouses and data marts in DMQL.

Syntax for Task-Relevant Data Specification
Here is the syntax of DMQL for specifying task-relevant data –
Use database database_name or
Use data warehouse data-warehouse-name
In relevance to att or dim_list
From relation(s)/cube(s)[where condition]
Order by order_list
Group by grouping_list
Syntax for Specifying the Kind of Knowledge

Characterization

The syntax for characterization is –

Mine characteristics [as pattern_name]
Analyze {measures(s)}
The analyze clause, specifies aggregate measures, such as count, sum, or count%.
For example –
Description describing student attendance sheet.
Mine characteristics as student attendance
Analyze count %
Description describing student attendance sheet.
Mine characteristics as student attendance
Attendance count %**0**

Discrimination

The syntax for Discrimination is –
Mine comparison [as {pattern name]}
For {target _class} where {target-condition}
{versus {constrast _condition i}
Where {contrast_conditini}}
Analyze(measure(s))}

Association

The syntax for Association is−
Mine association[a {pattern_name}]
{matching {metapattern}}

V. Case Study

Let us consider a small case study of a student information system (SIS) implemented at college where the use of one optimization strategy called Partitioning can improve the overall performance of the system. The implementation issues on partitioning are considered for Data warehouse system.

SIS maintains information if students related to their attendance in the lecture ,their performance in the different exams , fine details, complaints , feedbacks etc. lectures can getting an information of the current semester.

Let us find out volume of data generated attendance of the students. There are 4 classes and 8 lectures a day , and 60 students per class .It means 4*8*60=1920 records of students are inserted every day just to keep track of student's attendance. It grows to approximately 57600 a month and 691200 in a semester. A normal query to get attendance information for the current semester takes 691200 records. There will be around 2073600 records of attendance for three years of students in the college.

In such cases we divide a table that stores attendance in three partitioning according to their use the constraints such as availability, maintainability ,performance and manageability factors can be improved. Even using Indexing we can reduce the complexity of query performance and boost up the overall performance of the query .

VI. Conclusion

Optimization is much more than transformations and query equivalence. .Despite many years of work, significant open problems remain. However, it is necessary for making effective contribution to the area of query optimization. Partioning helps greatly in achieving the major factors of data warehouse systems. It helps from reducing the cost of storage systems to reducing the overall time in executing the query .using different data mining techniques we can improve query optimimzation.

References

[1] Jiawei Han and MichelineKamber, "Data Mining: Concepts andv Techniques", 2 edition (4 Jun 2006)

[2] Gordon S. Linoff and Michael J. Berry, "Data Mining Techniques:v For Marketing, Sales, and Customer Relationship Management", 3rd Edition edition (1 April 2011)

[3] ReemaTheraja. Data Warehousing.Oxford University Press, 2009.

[4] Arun K Pujari, Data mining Techniques University press 2ndEdn. 2009.