Study of Performance Tuning Techniques

Srushti Patil[#1], *Pooja Damare*[#2], *Jagruti Sonawane*[#3], *Neeta Maitre*[*4]

[#] Student, Computer Department, [*] Assistant Professor, Computer Department,
Cummins College of Engineering for Women, Pune, Savitribai Phule Pune University, Pune, India.

Abstract- In the modern era, digital data is considered as the more valuable asset of an organization, and the organizations assign more significance to it than the software and hardware assets. Database systems are computer-based record keeping systems, which have been developed to store data for efficient retrieval and processing. Since data is produced and shared every day, data volumes could be large enough for the database performance to become an issue. In order to maintain database performance, identification and diagnosis of the root causes that may cause delayed queries is done. Poor query design can be one of the major causes of delayed queries. There are various methods available to deal with the performance issues. Database administrator decides the method or combination of methods that work best.

In this paper, we present some performance tuning techniques such as SQL Tuning, Indexing and Table Partitioning along with their advantages and limitations.

Keywords – Database performance, Tuning, Index, Optimization, QEP(Query Execution Plan).

INTRODUCTION-

Now-a-days, large amount of data gets generated during any organization's workflow. So, database management system has become a crucial part of the organization. With the constantly growing amount of data being accumulated and processed by organization's information systems, database performance issues become more likely. Considering the constantly changing requirements and expectations of user, delay in response time could considerably affect the organization's operations. So, here comes the need of database performance tuning or database performance optimization which aims to make database systems run faster.

To achieve database tuning, it is important to understand the causes of the problems and find the current bottlenecks as there can be various possible factors affecting the database performance. A first category targets 'hardware' factors which include memory, processor, disk and network performance. Other category includes the database related factors such as the database design, indexing, partitioning or locking. And sometimes application level problems can also be the cause of performance degradation. [8]

Hardware issues are usually easier to detect and solve than other causes. So, in this paper the focus will be on factors which are more directly related to database systems itself.

PERFORMANCE TUNING -

Performance tuning in database management system means enhancing the performance of database, i.e., minimizing the response time at a very optimum cost. As query response time is the no. one metrics when it comes to database performance, query optimization is one of the important aspects of performance tuning. [1]

The objective of query optimization is to provide minimum response time and maximum throughput with the efficient use of resources. Query optimization primarily means selection, followed by sequencing in specific order, of the different SQL clauses to formulate an efficient query from the multiple query plans by drawing a comparison of the query plans based on the cost of the resources involved and the response time. [1]

PERFORMANCE TUNING TECHNIQUES –

Following tuning techniques can be used for solving performance issues. [9]

- SQL TUNING
- INDEXING
- TABLE PARTITIONING

1.SQL TUNING-

SQL tuning is believed to have the largest impact on performance (more than 50%).SQL is a declarative language which only requires the user to specify what data is wanted. There might be hundreds or thousands of different ways to correctly process the query. Hence, it's very hard for the DBMS query optimizer to decide which access path should be used. The best execution plan chosen by the query optimizer is called the query execution plan (QEP). [10]

List of methods for SQL query tuning-

1.1 Gather statistics :

It is for Oracle DB. It relies on up to date statistics to generate the best execution plan. Updated statistics helps optimizer to select perfect execution plan for a query.

It can be resource consuming. It must plan accordingly before executing. [3]

1.2 Index Management :

Indexes are optional structures associated with tables and clusters that allow SQL statements to execute more quickly against these tables. Index created columns help queries to select using index instead of doing full table scan, which is usually expensive.

DML statements can be slow if there is a lot of indexes on the table. [5]

1.3 Table Reorganization :

It is used to improve the performance of queries or DML operations performed against these tables. All data blocks will be moved to be together and prevent fragmentation which can cause slowness.

It is usually time consuming and needs downtime. [6]

1.4 Prediction :

It involves estimating the space used of a table , estimating the space use of an index and obtaining Object Growth Trends. It is able to predict the problem before it happened.

Sometimes the predictions are not accurate because the data consumed is not increased in sequence. [5]

1.5 Data Mart :

A data warehouse is designed for data to be collected directly from the various sources. The database will be grouped according to schemas and departments. It is easy to maintain and improve the database performance.

It is applicable only to the schemas which are less than 100 GB. It is not optimum to use data mart for bigger database. [5]

1.6 Materialized View :

It provides access to table data by storing the results of a query in separate schema object. It results in fast synchronization between source and target. Data can be refreshed on preferred method.

Complex queries on Materialized View tables perform badly especially if there are joins with other tables. [5]

1.7 Partition Table :

It splits the table into smaller parts that can accessed , stored and maintained independent of one another. It improves performance when selecting data . It can be used easily for data pruning.

It is hard for the DBAs to do maintenance on partitioned table if it involves lots of partition in a table. Index creation will be slow if hash partition is used. [4]

1.8 Query Rewriting :

It consists of the compilation of an ontological query into an equivalent query against the underlying relational database. It improves the way, data are being selected. By adding hints in the SQL, it sometimes enhances the performance of individual queries.

It can be a troublesome job to change hardcoded queries. Queries that are tested thoroughly could cause slowness. [2,7]

1.9 Monitoring Performance :

It is used to determine possible problems, locate the root cause and provide recommendations for correcting them. It is able to identify the root cause of the problem.

Only DBA is able to do monitoring. [5]

1.10 Optimization :

Query optimization is the process of choosing the most efficient way to execute an SQL statement. It helps queries to run faster. Data retrieval can be improved. Parameter tuning and memory tuning enable the database to perform in optimum level.

It must be done with supervision and wrong parameter set can cause database to go down or perform badly. Memory leak is also possible. [2,5]

2. INDEXING -

Indexing a table is a way to search and sort the records in table. With the use of indexes the speed with which the records can be retrieved from the table can be greatly improved.

Indexing can be thought of as a two-dimensional matrix independent of the table on which the index is being created. The two dimensions can be a column that would hold which is sorted extracted from the table on which index is being created and an address field that identifies the location of the record in the database. After creating the indexes it is important to collect statistics about the indexes using the RUNSTATS utility.

When a table is referenced in a database query and there is no index a table scan must be performed on that table. The larger the longer it would take for the table. However if indexing is used an index scan would be performed. An index scan is much faster than a table scan. Indexed files are smaller and require much less time to be read than a table especially when the table grows bigger.

A database manager uses a B+ tree structure. In this structure top level is called the root node. The bottom level consists of leaf nodes, where the actual key index values are stored, as well as a pointer to the actual row in the table. Levels between the root and leaf node levels are called intermediate nodes. In looking for a particular index key value, Index Manager searches the index tree, starting at the root node. The root contains one key for each node at the next level. The value of each of these keys is the largest existing key value for the corresponding node at the next level.

Indexes can reduce time significantly; however, indexes can also have adverse effects on performance. Before creating indexing, consider the effects of multiple indexes on disk space and processing time:

- Each index takes up a certain amount of storage on disk space. The exact amount is dependent on the size of the table and the size and number of columns included in the index.
- Each INSERT or DELETE operation performed on a table requires additional updating of each index on that table. This is also true for each UPDATE operation that changes an index key.
- The LOAD utility rebuilds or appends to any existing indexes.
- Each index potentially adds an alternative access path for a query, which the optimizer will consider, and therefore increases the query compilation time.

Indexing is a very important thing for fine tuning databases. It has to be done very judiciously. But the disadvantage is it can backfire also. So, consideration of other factors like disk space, I/O is also important.

3. TABLE PARTITIONING-

Huge data size is one of the common contributions to SQL bottleneck. In this case , the huge table will be analyst for partition where huge tables and index are split into smaller parts. Each partition is independent object with its own name. After performing table partitioning, database performance will drastically increase as less data is to be scanned and processed. Thus, it will be easier to maintain and manage database system with faster data retrieval.

At the same time maintenance including data purging, rebuilding index and backup can run faster. However, this method causes maintenance overhead for DBA in the case involving large amount of partitioned tables. In addition, index creation will be very slow if hash partition is used.[5]

CONCLUSION-

In this paper, an attempt is made to present a review of database performance tuning techniques is made. This paper focuses on tuning techniques which are directly related to database design. The main purpose behind this study is to understand major factors that can lead to database performance improvement. As query response time is the number one metrics when it comes to database performance, SQL tuning is one of the widely used tuning technique. SQL tuning aims to decrease response time and increase system throughput.

REFERENCES-

[1] A. Hameurlain, “Evolution of Query Optimization Methods: From Centralized Database Systems to Data Grid Systems”, Proceedings of the 20th International Conference on Database and Expert Systems Applications.

[2] Shi. Jiyuan, 2010. “Research and Practice of SQL Optimization In ORACLE.” Information Processing (ISIP), 2010 Third International Symposium on. IEEE.

[3] Hu, Ling, et al. “QueryScope: visualizing queries for repeatable database tuning.” Proceedings of the VLDB endowment.

[4] Cao, Wei and Dennis Shasha. “Tuning in action”. Proceedings of the 16th International Conference on Extending Database Technology.ACM.

[5] Herodotou, Herodotos, Nedyalko Borisov and Shivnath Babu. “Query optimization techniques for partitioned tables.”Proceedings of the 2011 ACM SIGMOD International Conference on Management of data. ACM.

[6] Oracle,2013. Oracle Database Administrator’s Guide 11g Release 1 (11.1).

[7] Hobbs, L. and P. Tsien. Oracle Database 10g Release 2 Online Data Reorganization and Redefinition.

[8] JONES, D. “The definitive guide to SQL Server Performance Optimization”, available online at www.realtimepublishers.com , pp 60-69.

[9] DUNHAM, J. “A guide to large-database tuning”, UNIX Review’s Performance Computing, San Francisco, pp. 35-41.

[10] SHASHA, D., “Tuning Databases for high performance”, AMC Computing Surveys, Baltimore, Vol. 28.