Database Query and Its Optimization: A Conclusive Report

Numptik Voratas
Asian College of Engineering, Thailand.

Query is a statement or group of statement that adequately execute some basic database operations viz. ‘Read’, ‘Write’, ‘Delete’, and ‘Update’. It plays a consequential role in managing and retrieving data. In general, distributed queries are more complex and complicated as compared to centralized queries. Queries can be categorized as data creation and data destruction, Data management queries, Data control quarry, OLTP and DSS quaries. In data creation and data destruction quaries create, insert and drop quaries are used. In data management quarry data is managed and manipulate, data can be insert, delete and update. In data control query, one can save data using commit command; permission can be granted using grant command [1][2][3].

In online transaction processing (OLTP) the work analysis and query optimization is done. In decision support system (DSS) queries used to retrieve data from large database. The execution time is not predictable in DSS query. Decision support system (DSS) queries are more complex as compare to online transaction processing queries (OLTP). The running time of DSS queries are unpredictable as compare to OLTP. The process of optimization in Decision support system (DSS) queries is complex as compare to OLTP queries. A distributed DSS query is used to retrieve data from multiple sites. In online transaction processing system (OLTP); real updates are performed. However, DSS queries execute batches as compared to real time updates. Online transaction processing (OLTP) database applications are optimal for managing changing data; these applications typically have many users who are performing transaction at the same time that change real time data, in other words OLTP is a live database. On other side the tables in a decision support database are heavily indexed and the raw data is frequently preprocessed and organized to support various types of queries to be used. The OLTP and DSS queries can be differentiated on the basis of different parameters as mentioned below [1][4][5][6]:

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Parameter</th>
<th>OLTP</th>
<th>DSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Abbreviation</td>
<td>Online transaction processing.</td>
<td>Decision support system.</td>
</tr>
<tr>
<td>2</td>
<td>Nature of Data</td>
<td>Common routine work.</td>
<td>Aggregate or subroutine information.</td>
</tr>
<tr>
<td>3</td>
<td>Execution Time</td>
<td>Less</td>
<td>More</td>
</tr>
<tr>
<td>4</td>
<td>Amount of Time</td>
<td>Predictable</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>5</td>
<td>Resource consumption</td>
<td>Less</td>
<td>More</td>
</tr>
</tbody>
</table>

A number of heuristics have been applied in recent times, which proposed new algorithms for substantially improving the performance of a query[1][2][3]. As stated by Manik Sharma et al. (2015) there are two major types of database queries called DSS and OLTP queries. To optimize a DSS query on the basis of usage of system resources, one has to find an optimal query execution plan which minimizes the Total Costs of a query. For finding the optimal query execution plan, the costs of
different performance metrics viz. Input-Output Costs, Processing Costs and Communication Costs, Total Costs and Runtime of DSS query should be computed and analyzed. Here, Total Costs represents the usage of the system resources required to execute a query. It is also defined as the sum of Local Processing Costs (LPC) and Communication Costs (CMCT). Several authors have used different techniques like exhaustive enumeration, dynamic programming, genetic algorithm, restricted genetic algorithm; entropy based restricted genetic algorithm and other hybrid evolutionary algorithms to optimize the distributed queries. Authors designed DSS queries using one of the DSS benchmark. The designed queries were then examined and analyzed on the basis of execution time, total costs and the quality of the solution[6][7][8][9][10]. Authors have analyzed the effect of varying Input output to communication costs ratio over the total costs of the DSS query. Authors found that by reducing the I/O to communication costs from 1:1.6 to 1:1, one is able to drastically reduce the communication costs[11]. Some of the authors have also statistically analyzed the performance of the query optimizer [12].

It is found that both OLTP and DSS need different types of query optimizer. Moreover, several techniques have been used to optimize the query. However, little work has been done using emerging evolutionary techniques. One should employ latest evolutionary techniques to optimize the OLTP and DSS queries.

References
