Comparative Analysis of Joins and Semi Joins in Distributed Query Optimization

Amit Verma[#], Rajinder Singh[*]

[#] *Department of Computer Science & Engineering, Guru Nanak Dev University*

Amritsar

[1]amitmtech4031@gmail.com

[2]tovirk@yahoo.com

Abstract— **In this paper the focus is given on computing and analyzing the performance of joins and semi joins in distributed database system. Database is defined as collection of files or table, where as DBMS stands for Database Management System which is collection of unified programs used to manage overall activities of the database. The two dominant approaches used for storing and managing database are centralized database management system and distributed database management system in which data is placed at central location and distributed over several locations respectively. Independent of the database approach used, one of the foremost issue in the database is the retrieval of data by using multiple table from central repository in centralized database and from number of sites in distributed database. Joins and semi joins are primitive operations used to extract required information from one, two or multiple tables. The various metrics that will be considered while analyzing performance of join and semi join in distributed database system are Query Cost, Memory used, CPU Cost, Input Output Cost, Sort Operations, Data Transmission, Total Time and Response Time. In short the intention of this study is analyze the performance and behavior of join and semi-join approach in distributed database system.**

I. INTRODUCTION

A distributed database system is the combination of two different technologies used for data processing: Database Systems and Computer Networks. In a distributed database environment, data is stored at different sites connected through a network. A distributed database management system (DDBMS) supports the creation and maintenance of distributed databases. An objective of a DDBMS is to present a simple and unified interface to the users so that they can access the databases as if there were a single database. In this way, there is no need for a user to be familiar with the underlying local database management systems. Distributed systems are a collection of independent cooperating systems, which enables storage of data at geographically dispersed locations, based on the frequency of access by users local to a site. The distributed database also enables combining of data from these dispersed sites by means of queries .[1]

A. *Query processing*

Query processing in a distributed database requires transfer of data from one computer to another through a communication network. Query at a given site might require data from remote sites. In query optimization, a cost is associated with each query execution plan. Cost is the sum of local cost (I/O cost, CPU cost at each site) and the cost of transferring data between sites. The complexity and cost increases with the increasing number of relations in the query. Further, minimizing the amount of data transmission is important to reduce the query processing cost. Due to the large number of parameters affecting query execution cost, a single query can be executed in several different ways. A query execution strategy or plan is required to minimize the cost of query processing. The key problem for query optimization in a distributed database is selection of the most cost effective plan to execute a query.[1]

The performance of a distributed database query depends on how fast and efficiently data is retrieved from multiple sites. Faster retrieval of data in a distributed database system is a complex problem since multiple sites are involved. Several factors impact the performance of distributed query processing. several factors impact the performance of distributed query processing. This factors are selection of appropriate site (when same data is replicated at multiple sites), order of operation (like select, project and join) and selection of join method (like semi join, natural join, equi join etc.). Due to the large number of factors involved, there could be multiple execution plans for a single query. Each plan is associated with a cost and the objective of a distributed query optimizer is to find a plan with lowest possible cost (optimal plan). The execution cost is expressed as a sum of I/O, CPU and communication cost.[2]

B. Query optimization

Query optimization involves two main tasks, search space generation and finding an optimal plan from the search space, using search strategies and cost model. The search space is a set of alternative execution plans for the input query. A given query is represented as query trees (Join trees) in a search space using transformation rules. If a given query involves many operators and many relations, then the search space will become very large, because it will contain a large number of equivalent query trees for the given query. Investigating a large search space may increase optimization time and cost. Hence, query optimization imposes some restriction on the size of the search space. To reduce the size of the search space, a restriction is imposed on the shape of a query tree. Several classes of join trees like linear and bushy trees exist. The problem of finding an optimal join tree from set of alternatives (search space) is solved by using several search strategies. The search strategy is used to explore the search space in order to find an optimal plan from set of alternatives using a cost model.[2]

Query optimization could be static or dynamic. In a static optimization strategy, a given query is parsed, validated, optimized once and stored in the database. Dynamic optimization strategy works in bottom-up way by building more complex plans from the simpler plans.[11]

C. Join

Join is one of the most imperative operations in database theory that is used to extract information from two or more than two tables. Technically join operation is one of the special cases of Cartesian product. In join unlike Cartesian product before concatenation the tuples of the join tables are checked against specified condition. There are various types of joins like equi-join, self join, inner join, outer join etc. Independent of type all of these are used to extract data from two or more tables.[4]

A natural join is an equijoin of two relations over a specified attribute, more specifically, over attributes with the same domain. A natural join is denoted by R.

D. Semi-joins

A semi-join is one of the important operations in relation theory that is used to optimize a joins query. Semi-join is used to reduce the size of relation that is used as an operand. A semi-join from Ri to Rj on attribute A can be denoted as Rj⋉ Ri . Research shows that semi joins are very helpful in optimizing the join query by reducing the quantity of data exchanged. But one of the darken side of using semi join is that it increases the local processing cost as well as number of message. It returns rows that match an EXISTS sub-query without duplicating rows from the left side of the predicate when several rows on the right side satisfy the norms of the sub-query.[4]

II. Analysis Of Distributed Cost Model

Distributed database system, provides data distribution transparency by hiding the data distribution details from the users. [5] Whenever a distributed query is generated at any site of a distributed system, it follows a sequence of phases namely query decomposition, query fragmentation, global query optimization and local query optimization.

The allocation of data considers a set of fragments , a set of locations in a network , and a set of applications placed at L. These applications need to access the fragments which should be allocated in the locations of a network. The allocation problem consists on finding an optimal distribution of F over L [8]. Thus, distributed cost model includes cost functions to predict the cost of operators, database statistics, base data, and formulas to calculate the sizes of intermediate results.

A. Cost Functions

In a distributed system, the cost of processing a query is expressed in terms of the total cost measure or the response time measures .The total cost measure is the sum of all cost components. If no relation is fragmented in the distributed system and the given query includes selection and projection operations, then the total cost measure involves the local processing cost only. However, when join and semijoin operations are executed, communication costs between different sites may be incurred in addition to the local processing cost. Local processing costs are usually evaluated in terms of the number of disk accesses and CPU processing time, while communication costs are expressed in terms of the total amount of data transmitted. [4] For geographically dispersed computer networks, communication cost is normally the dominant consideration, but local processing cost is of greater significance for local networks. But in wide area networks the local processing cost is mostly ignored and an emphasis is made on minimizing the communication cost.

III. OBJECTIVE OF THE STUDY

- To understand the significance of the Query Processing in the Distributef Database Management System.

- To Design an Algorithm which is used to analyze the Query in the DistributedDatabase Management System.
- To compute and analyze different metrics of Query in Distributed Database Management System.

Distributed query processing is to translate a high-level query on a single logical distributed database (as seen by the users) into a low-level language on physically distributed local databases. In distributed query processing, the total cost should be minimized for executing a distributed query. [5] The query execution involves only a local processing cost when no relation is fragmented in a distributed DBMS. On the other hand, if relations are fragmented, a communication cost is incurred in addition with the local processing cost. The aim of distributed query processing is to minimize the total execution cost of the query which includes the total processing cost (sum of all local processing costs in participating sites) of the query and the communication cost. The local processing cost of a distributed query is evaluated in terms of the number of disk accesses (I/O cost) and CPU cost. The CPU cost is incurred when performing data operations in the main memory in participating sites. The I/O cost can be minimized by using efficient buffer management technique. In a distributed query execution, the communication cost is required to exchange data between participating sites. Hence, the communication cost depends on several factors such as the amount of data transfer between participating sites, the selection of best site for query execution, the number of message transfer between participating sites, and the communication network. In case of high-speed wide area networks (with a bandwidth few kilobytes per second), the communication cost is the dominant factor and the optimization of CPU cost and I/O cost can be ignored in such cases. The optimization of local processing cost is of greater significance in case of local area networks.

IV. CONCLUSION

In this paper we discussed about the factors on which the processing of the query depends. The comparative analysis of the Query Processing using joins and semi joins in Distributed Database Management System is discussed. The use of joins and semi joins affects the cost of Query Processing in Distributed DBMS very much and it the cost can be reduced using joins and semi joins in different conditions.

V. AKNOWLEDGEMENT

Author is highly indebted to Dr. Rajinder Singh Virk, Associate Professor, DCSE, Guru Nanak Dev University, Amritsar for his precious guidance from time to time.

VI. BIBLIOGRAPHY

[1] “Principles of Distributed Database Systems” second edition by M. TAMER OZSU.

[2] “Database Concepts” seventh edition by Navathe.

[3] “Study of Query of Distributed Database Based on Relation Semi Join” paper proposed by Xiaofeng Li, Dong Le, Hong Zhi Gao, Lu Yao in 2010 International Conference On Computer Design And Appliations IEEE in July2010.

[4] “Analysis of Joins and Semi Joins in a Distributed Database Query ” by Dr. Rajinder Singh Virk , Manik Sharma, Dr. Gurdev Singh in July2012 International Journal of Computer Applications (0975 – 8887) Volume 49– No.16.

[5] “Distributed Database System Query Optimization Algorithm Research” by Fan Yuanyuan and MiXifeng in 2010 IEEE.

[6] “Combining Join and Semi-Join Operations for Distributed Query Processing ” by Ming-Syan Chen and Philip S. Yu in June 2007 IEEE.

[7] “Review of Dynamic Query Optimization Strategies in Distributed Database” by Pankti Doshi and Vijay Raisinghani.

[8] “Using Join Operations As Reducers In Distributed Query Processing” by Ming-Syan Chen and Philip S. Yu.

[9] “Why Not Semi-joins for Streams, When Distributed? ” by Tri Tran , Byung Suk Lee , Matthew W. Bovee.

[10] “Review of Dynamic Query Optimization Strategies in Distributed Database” by Pankti Doshi Vijay Raisinghani.

[11] “Distributed Database System Query Optimization Algorithm Research” by Fan Yuanyuan and MiXifeng.

[12] “A Method for Processing Distributed Database Queries” by WILLIAM PERRIZO.

[13] “Optimal Query Processing for Distributed Database Systems” by WESLEY W. CHU, FELLOW, IEEE, AND PAUL HURLEY.