International Journal of Engineering Technology Science and Research
IJETSR
www.ijetsr.com
ISSN 2394 – 3386
Volume 2 Issue 8
August 2015

Optimizing Join Query In Distributed Databases

Rudrani sharma[1], Er. Pushpneel Verma[2], Er. Sachin Chaudhary[3]
Department of Computer Science & Engineering
Bhagwant Institute of Technology, Muzaffarnagar

Abstract— *Query Optimization is to use the best plan for the query that improves the performance of the query. Query Optimization is difficult in distributed databases as compared to centralized databases. Queries in distributed databases are effected by factors such as insertion methods of the data into the remote server and transmission time between servers. Response time of the query depends upon the transmission time, local processing speed.*

Keywords—distributed database, query optimization, objectives and factors of join query, conclusion

INTRODUCTION

A distributed database as a collection of multiple, logically interrelated databases distributed over a computer network. A distributed database management system (distributed DBMS) is then defined as the software system that permits the management of the distributed database and makes the distribution transparent to the users. Sometimes "distributed database system" (DDBS) is used to refer jointly to the distributed database and the distributed DBMS. The two important terms in these definitions are "logically interrelated" and "distributed over a computer network."

Benefits of using distributed databases are:

i. **Improved Performance**: Because the data is stored on multiple sites, so the overhead on one machine decreases which improves the performance.

ii. **Localization**: means the data is present as close to the site where it is needed, therfore data can be accessed in less time and data transfer time also reduces.

iii. **Availability and Reliability**: In distributed database systems, the availability of the data increases because the replicas of the data are distributed at different sites. It also increases the reliability because if the one site fails, then data can be accessed from the other site where its replica is present. So, in distributed environment, failure of one site does not result in unavailability of the data.

iv. **Reduced communication overhead:** Communication overhead reduces in distributed environment because a relation is available at each site locally that contains the replicas of the data.

v. **Easier System Expansion***:* The capacity of the distributed database can be increased easily by adding the computers to the network [7]. In the distributed environment, because system can associate and coordinate a number of small machines so it gains the power equal to the power of a supercomputer.

QUERY OPTIMIZATION

query optimization aims at choosing the "best" point in the solution space of all possible execution strategies. An immediate method for query optimization is to search the solution space, exhaustively predict the cost of each strategy, and select the strategy with minimum cost. Query Optimization is to operate the query in different way so that it gives the same result but the speed to retrieve the data increases. The queries should be efficient so that data can be retrieved in less time or accessing to database became fast. There are alternative ways to perform the query that give the same result. The way to perform the query should give the result in minimum time and should increase the performance of the query. In distributed systems, the cost of a query plan is given by the sum of the transmission cost and the local processing cost [5]. The transmission cost is in factors of speed to transfer the data from one machine to another machine and the local processing cost is in terms of CPU cycles, disk I/O. Query Optimizer determines:

- ✓ Number of alternative plans
- ✓ Cost of every plan using cost model
- ✓ Selects the plan with the lowest cost

Join Query in distributed databases is used to join the data from multiple sites. Optimization of join query in centralized databases is simple as compared to distributed databases. More work is done on join query in centralized databases and more optimization is required in distributed databases.

International Journal of Engineering Technology Science and Research
IJETSR
www.ijetsr.com
ISSN 2394 – 3386
Volume 2 Issue 8
August 2015

RELATED WORK ON JOIN QUERY OPTIMIZATION IN DISTRIBUTED DATABASES

There are different methods to optimize the queries in the databases. These methods improve the performance of the query and decrease the cost. The optimizer determines that in which order the queries (e. g. joins, selects, and projects) should be executed. Related work on join query optimization in distributed databases is to calculate the size of the data on two different machines and then to send the table having smaller size to another site and then perform the join query [5].

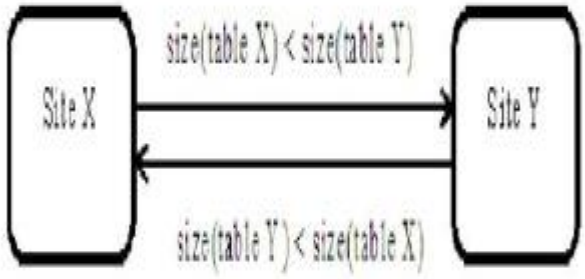

Figure 1: Minimize size of transmission data [5]

Another method for join query in distributed databases is the parallel query processing. Parallel processing doesn't focus on minimizing the quantity of transmission data but rather maximizing the number of simultaneous transmissions [7].

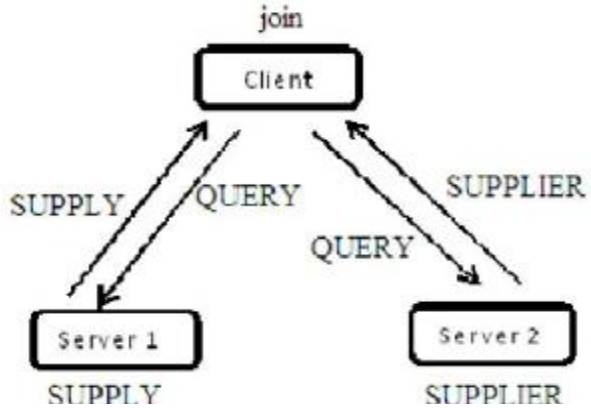

Figure 2: Parallel processing of join query

Client sends the request for the data from server 1 and server 2 by the queries. After that, server 1 sends the SUPPLY data and server 2 sends the SUPPLIER data to client. Then client inserts the data into its database and performs the join query on the data from two servers.

If server 1 contains the SUPPLY relation as:

SUPPLY(SUPPLY_NO, FROM_PLACE, TO_PLACE)

And

server 2 contains the SUPPLIER relation as:

SUPPLIER(SUPPLY_NO, S_NAME, S_ADDRESS)

and

client wants the join of the SUPPLY and SUPPLIER relation from server 1 and server 2 respectively and want to perform the query Q.

Q: SELECT *FROM SUPPLY S, SUPPLIER sr WHERE s.SUPPLY_NO = sr.SUPPLY_NO

In distributed databases, query Q can be divided into three parts:

1. SELECT *FROM SUPPLY
2. SELECT *FROM SUPPLIER
3. SELECT *FROM SUPPLY S, SUPPLIER sr WHERE s.SUPPLY_NO = sr.SUPPLY_NO

Queries 1 and 2 select the data from two source tables. Because this data resides on the remote machines, the executions of these two queries do not require data transmission. Query 3 is the join query which can not be executed until the data on the remote sites have been transferred to the same sites.

OBJECTIVES OF JOIN QUERY IN DISTRIBUTED DATABASES

To perform the distributed join query that is accessing the data from the remote sites, objectives for the optimization are:

- ✓ *Size of transmitted data*: It is the amount of the data that is to be transmitted. Size of the transmitted data should be small so that less time will be required for transmission.
- ✓ *Transmission speed*: it depends upon the network speed. For the wide area networks, transmission speed more affects the query.
- ✓ *Local processing costs*: it consists of CPU cost, I/O cost. Local processing costs can vary with the machine processing speed.

To increase the performance of join query, these costs should be less and operations should be performed in efficient manner for optimization of the query.

FACTORS FOR PARALLEL PROCESSING OF JOIN QUERY

Parallel processing of join query in distributed database depends upon the following factors:

International Journal of Engineering Technology Science and Research
IJETSR
www.ijetsr.com
ISSN 2394 – 3386
Volume 2 Issue 8
August 2015

Time for query execution for requesting the data: time for the client to send the request to servers for the data that is placed at servers.
Time for transmitting data: The transmission time increases as the quantity of transmitted data increases. It depends upon the network speed between client and server. Response time to transmit the data is given by:
Max(size(SUPPLY), size(SUPPLIER))
Time for inserting data: time taken to insert the data into the
client database from the servers.
Different insertion methods can be used:

- ✓ Row-by-row insertion
- ✓ Bulk insertion

Bulk insertion is better to use than the row-by-row insertion. Another type of insertion methods can be used to optimize the insertion of data.
Time for join execution: time taken to perform the join query at client side that joins the tables from server 1 and server 2.
Optimization of join query can be done by using:

- ✓ different join orders
- ✓ alternative "where" clause that will give the same result
- ✓ different join methods

join also depends upon the local processing cost of query such as CPU cost and I/O cost.
To perform the distributed join query that is accessing the data from the remote sites, costs should be less so that performance of join query increases. If a machine wants the result of join query of data that is present at different machines, then transmission costs and the insertion costs are very important. Insertion of the server's data into client database takes the more time than the transmission of the data. Therefore, the important objective is to improve the performance of the join query that is accessing the data from two different machines by using the different insertion methods that take less time.

PROPOSED WORK

First method for the join query is first to transfer the data from servers to client and then insert the data into client database, after that results are shown by performing join query at the client site that takes the data from its own database . Time for this method will equal to the addition of time to fetch the data from server sites, time to insert the data into client database and join query processing time.
Second method that is my proposed work will show the results by performing the join query on the client site without inserting the data into its database. The join query in my proposed work will directly take the data from server sides. Time for proposed method will depend upon the time to fetch the data from server sides and join query execution time. Then compare both the methods based upon their performance. In the proposed method, insertion time of data into client database will be deducted. Therefore, the join query will be optimized in distributed databases.

CONCLUSION

This paper presents the join query optimization in distributed databases. One method for the join query is first to transfer the data from servers to client site and then insert the data into client database, after that join query is performed. Proposed method will directly perform the join query on the client site after fetching from servers site and it will not insert the data into client database. By the proposed method, insertion time of data into client database will be deducted. So, this method will optimize the join query in distributed databases.

REFERENCES

[1] Aljanaby Alaa , Abuelrub Emad, and Odeh Mohammed , " A Survey of Distributed Query Optimization", The International Arab Journal of Information Technology, Vol. 2, No. 1.
[2] Mullins Craig S., "Distributed Query Optimization", Technical Support .
[3] Nicoleta Iacob , "Distributed Query Optimization", PhD Student, University of Piteşti, Issue 4/2010
[4] Ioannidis Y. E. and Kang Y. C. , "Randomized Algorithms for Optimizing Large Join Queries", in Proceedings of the ACM SIGMOD Conference on Management of Data, Atlantic City, USA, pp. 312-321.
[5] Ghaemi Reza, Amin Milani Fard, Tabatabaee Hamid, and Sadeghizadeh Mahdi ," "Evolutionary Query Optimization for Heterogeneous Distributed Database Systems", World Academy of Science, Engineering and Technology 19.
[6] Mor Jyoti, Kashyap Indu, Rathy R. K., "Analysis of Query Optimization Techniques in Databases", International Journal of Computer Applications (0975 – 888) Volume 47– No.15.
[7] Jiang Shun , "Optimizing Join Query in Distributed Database", University of North Carolina, Wilmington.

[8] Valduriez Patrick , "Join Indices", ACM Transactions on Database Systems, Vol. 12, No. 2

[9] Ioannidis Y. E. and Kang Y. C. , "Left-Deep vs. Bushy Trees: An Analysis of Strategy Spaces and its Implications for Query Optimization", SIGMOD Conference 1991: 168-177.

[10] Sukheja Deepak, Singh Umesh Kumar (July 2011), "A Novel Approach of Query Optimization for Distributed Database Systems", IJCSI International Journal of Computer Science Issues, Vol.8,Issue4,No1.

[11] M. Tamer Özsu • Patrick Valduriez Principles of Distributed Database Systems Third Edition

[12] International Journal of Scientific and Research publications, volume3, Issue 5, May 2013 ISSN 2250-3153