Analysis of Joins and Semi Joins in a Distributed Database Query

Manik Sharma
Assistant Professor & Head
Department of Comp. Science
Sewa Devi S.D College TT

Dr. Gurdev Singh
Professor
Department of IT
VIET, Banur, Rajpura

Rajinder Virk
Phd,Associate Professor
DCSE
Guru Nanak Dev University

ABSTRACT
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. In this paper the focus is given on computing and analyzing the performance of joins and semi joins in distributed database system. 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.

Keywords
Distributed Database, Data Transmission, Response Time, Total Time, Join, Semi join etc.

1. INTRODUCTION
Data is one the vital entity in the database is managed by two using two major database approaches known as Centralized Database Management Approach and Distributed Database Management Approach. Centralized database management is one the traditional approach of database management, in which all of the data in database is sited on central location. Centralized database approach has overcome several limitations of file oriented approach of prior times. Since in centralized database approach the data is placed on central repository hence it is easy to access or extract data from multiple tables as compare to distributed database approach where data is distributed over several sites. In centralized database the database query can be easily transformed into set of relational algebra's operation, but in distributed database system one has to put more effort to analyze the amount of data exchange in addition to corresponding set of relational algebra's operations. Distributed database system [1] [4] [8] is defined as collection of logically interrelated data distributed over several sites.

The number of nodes in distributed system is connected either by using wired or wireless network media. In other words distributed database system [3][10] is defined as the convergence of database system and Computer Network. One of the major issues in the distributed database design is the placement of data and program across the number of computer or site available in the system.

After making placement of data and application program one has to focus on transforming a distributed query into equivalent low level query so that actual implementation and execution strategy of the query can be carried out. In distributed database system, it is obvious the database query will extract data from several different sites, so in this case the important factor is to reduce to amount of data transmission to maximum extent.

2. OBJECTIVE OF STUDY
The various objectives of this study are:

- To understand the significance of joins and semi join in distributed database.
- To compute and analyze different metrics of query using join and semi join operations in distributed database system.
- To compute the cost of query using cost based query optimizer and provides some variant alternate for the query.
- To compute and analyze the data transmission from one site to another in processing query using joins and semi joins approach in distributed database.
- To compute and analyze Total Time and Response Time of query implemented with join and semi join approach in distributed database.

3. JOINS AND SEMI JOINS
Before proceeding further let us first understand the concept of Join and Semi joins.

Join [6] 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. The center of attraction in this study will be equi-join one of the most frequently used type of join.

A semi-join is one of the important operations in relation theory that is used to optimize a joins query. Semi join [3] 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 $Rj \ltimes Ri$. 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. The research has shown that Semi-join and anti-

join transformation cannot be done if the sub-query is on an OR branch of the WHERE clause.
The objective of semi join in distributed database is to reduce the data transmission [2] from one site to another.
The semi join can be implemented by using different join methodology. The following algorithm explains the working of semi joins in nested loop.

```
Open table1
While not end of table1
        Read tuple from table1
        Success=false
        Open table2
        While not end of table2
                Read tuple from table2
                If  (table1.tuple==table2.tuple) then
                        Success=true
                        Exit loop
                End if
End while
Close table2
End while
Close table1
```

ragged.

4. EXPERMENTAL ANALYSIS

The processing of distributed query is different from centralized query. One of the vital parameter in distributed query processing is the amount of data transmission required for getting required result. To analyze the working and performance of joins and semi joins operation in centralized as well as in distributed database system the following tables EMP and DEPT are to be considered. While analyzing the performance in centralized database system it is obvious that EMP and DEPT table are placed on same site or location. On the other hand while analyzing the performance in distributed database it is assumed that EMP and DEPT tables are placed at site1 and site2 respectively. The complete structure of the above said table is as given below:
Dept Table

```
Create table DEPT
(Deptno number (2) constraint pk_dept primary key,
Dname varchar2 (14) ,
Loc varchar2 (13) ) ;
```

Emp Table

```
Create table EMP
(Empno number(4) constraint pk_emp primary key,
Ename varchar2(10),
Job varchar2(9),
Mgr number(4),
Hiredate date,
Sal number (7, 2),
Comm number (7, 2),
Deptno number (2) constraint fkey_deptno references dept);
```

It is assumed that both relations are not fragmented. Suppose EMP table has total 14 tuples and each tuple consumes 51bytes. Similarly DEPT table has 4 tuple and each tuple consumes 29bytes. So total memory consumed by EMP and DEPT table is 714bytes and 116bytes respectively. It is further assumed that the following query is requested at site3.
Find the name, Dname, Deptno and location of the employee where he/she works.

```
Select emp.ename, dept.dname, emp.job, dept.deptno from
EMP, Dept where emp.deptno=dept.deptno;
```

4.1 Query Processing Using Joins in Distributed Database System

Since reduced cost and advanced communication technology gives birth to the idea of Distributed Database Management Systems that turn out to be an integral part of many computer applications. Distributed Database [7] system is cluster of distributed computers that are coupled with one another with the help of some communication media (like twisted pair, coaxial cable, fiber optics, satellite etc.) on which a database is allocated and placed. It is obvious that a query may have different equivalent transformation that lead to different resource consumption. So in distributed database system one has to keep in mind the consumption of resources while selecting the execution strategy for the query.
So while execution distributed query one has to keep in mind various factors like equivalent relational algebra's operations, placement of data and application programs, ordering of relation algebra's operations, bytes transferred from one site to another, Total_Time, Response_Time etc.
Now let us understand the meaning and significance of Total Time and Response Time. In the distributed cost model [8] [9] total time which is computed by adding all the cost components (Local Processing Cost and Communication Cost) of a query, whereas Response Time is computed as an elapsed time from the starting to completion of query. Mathematically the Total_Time and Response_Time are computed as follow:
Total_Time =

TCPU * # Instructions + TIO * IO + +TMessage * #Messages +TTCost * #Bytes …. Eq-I

Response Time

TCPU * # SInstructions + TIO *SIO + +TMessage * # S_Messages +TTCost * #SBytes Eq- II

The original query that extract data from two tables EMP and Dept in distributed database system can be implemented and executed in three different ways as given below in Case I, II and III.
Case I: In this case to implement and execute the query one has to transfer both join table EMP and DEPT to the resultant location i.e. at site 3. The following diagram shows the query plan of above said case.

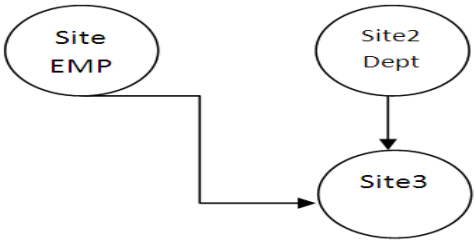

Figure1 : Data Transmission

The total number of bytes transferred in this case will be computed as follow:

14*51+29*5

=714+116

=830bytes

Total_Time (TM) =2TMessage+TTCost (1,05,000Bytes)

Response_Time =Max (TMessage+TTCost*714Bytes), TMessage+TTCost*116Bytes)

Case II: transfer EMP table to Site 2 where dept table is available. Apply join operation here i.e. at site 2 and transmit the required result at site 3.
The following diagram shows the query plan of above said case.

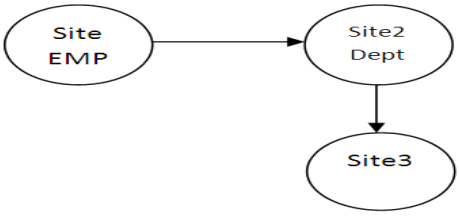

Figure2: Movement of Data

The total number of bytes transferred in this case is as follow:

51*14+35*14

=714+490

=1204bytes

Total_Time (TM) =2TMessage+TTCost(1204Bytes)

Response_Time =Max (TMessage+TTCost*714Bytes), TMessage+TTCost*490Bytes)

Case III: This is just reverse case of Case II; in this case Dept table will be transmitted at Site 1 where EMP table is already available. Now apply the join operation here and transmit the required result to site 3. The following diagram shows the query plan of above said case.

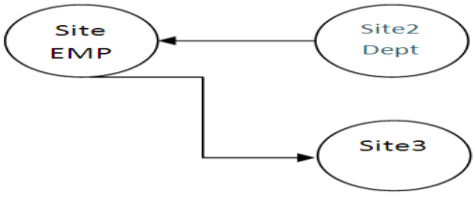

Figure3: Movement of Data

The total number of bytes transferred in this case will be computed as follow:

29*4+35*14

=116+490

=606bytes

Total_Time (TM) =2TMessage+TTCost(606Bytes)

Response_Time =Max(TMessage+TTCost*116Bytes), TMessage+TTCost*490Bytes). The following table shows the analysis of different metrics.

ID	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	Instance	In-Out
0	Select Statement		14	518	7 (15)	00:00:01		
1	Hash Join		14	518	7 (15)	00:00:01		
2	Table Access Full	EMP	14	238	3 (0)	00:00:01		
4	Remote	D	4	80	3(0)	00:00:01	Desktop	R->s

4.2 Query Processing Using Semi Joins

The above said query when implemented with semi join approach will look like as follow:

Select emp.ename, scott.d.dname@desktop job, scott.d.deptno@desktop from emp,scott.d@desktop where scott.d.deptno@desktop in (select scott.d.deptno@desktop from scott.d@desktop where emp.deptno=scott.d.deptno@desktop)

In case of semi join the joining attribute of table T1 located at site S1 is send to the site S2 where other joining table T2 is placed. The joining attribute is then joined with the available

join table T2. After this the projection operation is implemented on the resultant table Temp 1 and is transmitted back to the original site S1, where the resultant is joined back with table T1. Now project the join attribute deptno of Department table located at Site2 and transmit it to Site 1. The total data transmission in this case is

2*4=8Bytes Shipped from Site 2 to Site1

Now apply join operation at Site 1 of table1 with Transmitted join attribute and then apply the projection operation to extract attributes (Empno, Name, Job and Deptno). The resultant table after join is again transmitted back to Site 2. So the data transmission at this point is:

35*14=490Bytes

Total_Time (TM) =2TMessage+TTCost(498Bytes)

Response_Time =Max(TMessage+TTCost*8Bytes), TMessage+TTCost*490Bytes).

ID	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	Instance	In-Out
0	Select Statement		14	770	11 (19)	00:00:01		
1	Hash Join		14	770	11 (19)	00:00:01		
2	Hash Join		4	184	8 (25)	00:00:01		
3	View	Vw_SQ_1	4	104	4 (25)	00:00:01		
4	Hash Unique		4	52	4 (25)	00:00:01		
5	Remote	D	4	52	3(0)	00:00:01	Desktop	R->s
6	Remote	D	4	80	3 (0)	00:00:01	Desktop	R->s
7	Table Access Full		14	126	3 (0)	00:00:01		

From the above analysis one come to conclude that semi joins gives its best when one want to reduce the amount of data transmission from one site to another.

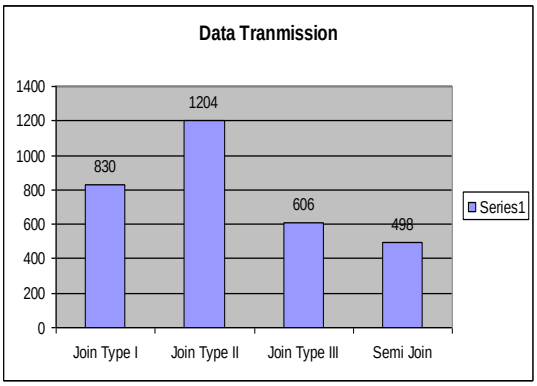

Figure 4: Analysis of Data Transmission

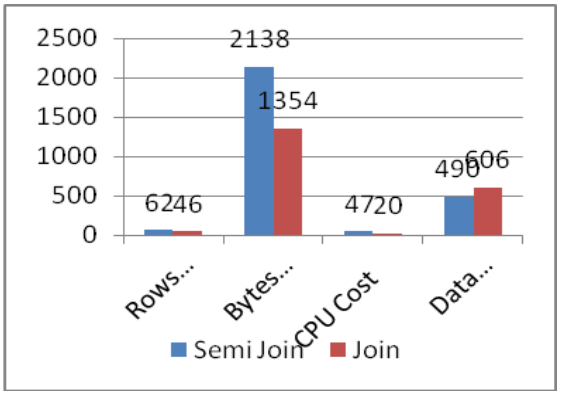

Figure: Analysis of Join and Semi Join Operations

5. JOINS VERSUS SEMI JOINS

One of the interesting questions is when the query has to be executed with Join and when with semi join. The selection of join and semi joins in distributed system is directly depends upon the data transmission from one site to another. In this study the major fact that came out is that semi joins is found more useful than join when the data transmission from one site to another is more. The following pseudo-code will explain the case when join or semi join will be selected for execution of query.

Assumption: it is assumed that table T1, T2 are placed over site S1,S2 and the query is requested and resulted on Site S3. In the following pseudo-code Scost refers to cost of semi join operation, JA is joining attribute, R is resultant table compiled after joining join attribute with joining table followed by selection and projection operation if required, JCost is cost of join operation. Here cost means transmission cost only.

Step1: Read Table T1

Step2: Read Table T2

Step3: Read Operation

Step4: Project JA from the required table

Step5: Scost=Cost (JA) + Cost (π(JA JT) + Cost (R)

Step6: Jcost=Cost (T1) + Cost (T2)

Step7: IF (SCost<JCost) Then

Execute Operation with Semi Join

Else

Execute Operation with Join

End IF.

6.CONCLUSION

From the above analysis in distributed database system the analysis shows that join approach gives its best in data transmission when a relation having lower cardinality is transmitted to the location where a relation of upper cardinality and larger tuple size is placed. In regard to total time it is clear from above analysis that the query executed with semi join possess lesser total time when data transfer is more. It is very difficult to conclude which one is better in join and semi joins. From the above study it is clear that the data transmission in a distributed query using semi join is always lesser than the data transmitted in distributed query using joins operation however data accessed using semi join may be larger than join operation. No doubt semi joins implement more operation as compare to join, but it reduces the number of bytes transferred from one site to another to great extent. Further one is able to conclude that semi joins are beneficial if the transmission cost is of main consideration, otherwise joins will be preferred.

6. ACKNOWLEDGMENTS

Authors are highly indebted to Dr. Gurvinder Singh, Associate Professor and Head, DCSE, Guru Nanak Dev University, Amritsar for his precious guidance from time to time.

7. REFERENCES

[1] Nilarun Mukherjee, Synthesis of Non Replicated Dynamic Fragment Allocation Algorithm in Distributed Database System", Published in Proceeding of international conference on advances in Computer Science , 2010

[2] Ramez Elmasri, Shamkant B. Navathe, "Fundamentals of Database System", Fifth Edition, Pearson Education, Second Impression, pp 894, 2009.

[3] M. Tamer Ozsu, Patrick Valduries, "Principles of Distributed Database System", Second Edition, Pearson Education, pp 169.

[4] T.V. Vijay Kumar, Vikram Singh, "Distributed Query Processing Plans Generation Using GA", International Journal of Computer Theory and Engineering, Vol 3. No.1, Feb 2011.

[5] Narasimhaiah Gorla, Suk-Kyu Song, "Subquery allocation in Distributed Database using GA", JCS & T, Vol. 10, No.1.

[6] Deepak Shukla, Dr. Deepak Arora, "An Efficient Approach of Block Nested Loop Algorithm based on Rate of Block Transfer", IJCA, Vol.21, No.3, May 2011.

[7] Swati Gupta, Kuntal Saroha, Bhawna, "Fundamental Research in Distributed Database", IJCSMS, Vol. 11, Issue 2, Aug 2011.

[8] Reza Ghaemi, Amin Milani Fard, Hamid tabatabee, "Evolutionary Query Optimization For Hetrogenous Distributed Database System", World Academy of Science, Engineering and Technology, 43, 2008.

[9] Johann Christoph Freytag, "The Basic Principles of Query Optimization in Relational Database Management System", Internal Report, IR-KB-59, March 1989.

[10] T.V. Vijay Kumar, Vikram singh, Ajay Kumar Verma, "Distributed query Processing Plans Using GA", IJCTE, Vol 3. , No.1, 2011.