Analysis of Joins and Semi-joins in Centralized and Distributed Database Queries

Manik Sharma
Assistant Professor & Head, Computer Department
Sewa Devi S.D. College
Tarn Taran, Punjab, India
manik_sharma25@yahoo.com

Dr. Gurdev Singh
Professor (IT)
Gurukul Vidyapeeth Institute of Engg. & Technology
Banur, Rajpurar, Punjab, India
singh.gndu@gmail.com

***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 centralized as well as in distributed database system. The various metrics that will be considered while analyzing performance of join and semi join in centralized database and 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 compare and contrasts the behavior join and semi-join approach in centralized and distributed database system.**

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

I. 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.

II. OBJECTIVE OF STUDY

The various objectives of this study are:

- To understand the significance of joins and semi join in centralized and distributed database.
- To compute and analyze different metrics of query using join and semi join operations in centralized and 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, total time, response time in processing query using joins and semi joins approach in distributed database.

III. 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

978-0-7695-4817-3/12 $26.00 © 2012 IEEE
DOI 10.1109/ICCS.2012.15

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 R_i to R_j on attribute A can be denoted as $Rj \ltimes R_i$. 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.

IV. Experimental 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. 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;

Query Processing Joins in centralized DB

In centralized database system all of the data is placed over a central location to avoid any redundancy in the database. Further this approach also improves the security in the database and help in managing concurrent transaction also. As discussed above it is easy to implement and execute query in centralized database system as compare to distributed database system. To analyze the query using Joins and semi joins in centralized database system, the focus is given on computing different metrics based on cost based optimizer of query optimization like Cost of Query, Bytes Accessed, Cardinality and optimizer used. The following table shows how the values of different metrics varies when the above said query is implemented and executed in centralized database system using join approach.

Table 1: Metric Analysis Using Join Approach

Operation	Object	Cost	Bytes	CPU Cost	IO Cost
Select Statement		4	476	156447	4
Nested Loops		4	476	156447	4
Table Access	EMP	3	294	39667	3
Table Access	DEPT	1	13	8341	1
Index	PK_DEPT	0		1050	0

The same query that extract data from two tables EMP and DEPT when implemented with semi join approach show different behavior in regard to computed metrics as given below:

Table 2: Metric Analysis using Semi Join Approach

Operation	Object	Cost	Bytes	CPU Cost	IO Cost
Select Statement		4	462	171147	4

Nested Loops		4	462	171147	4
Nested Loops		3	280	54367	3
Table Access	DEPT	3	238	39667	3
Index	PK_DEPT	0	3	1050	0
Table Access	EMP	1	13	8341	1
Index	PK_DEPT	0		1050	0

From the above computed table one is able to produce the following graphical analysis of different metrics.

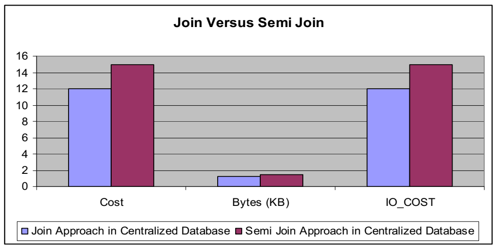


Figure 1: Comparison of join and semi join

It is commonly said that semi join approach helps in reducing the disk access in centralized database management approach. However, from the above tables it is found that in centralized database system semi join increases the Cost of query, Bytes accessed, and Input Output cost of query as compare to join. Further the above analyzed table shows the CPU Cost of the query using semi join approach is also more as compare to CPU Cost of query using join approach.

The following chart shows the variation in CPU Cost.

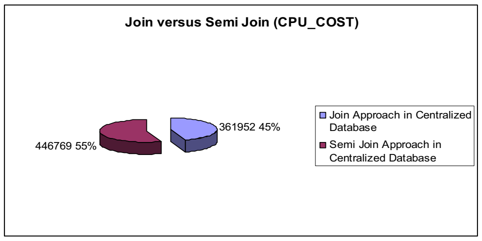


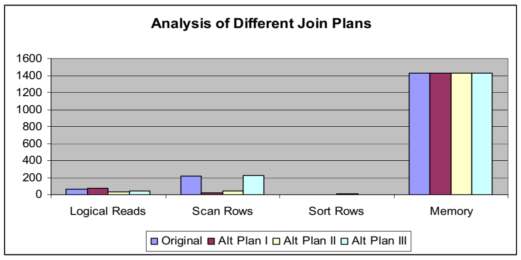
Figure 3: Comparison of CPU Cost

The above said query when analyzed by using one of important performance analysis application software Toad for oracle, it will analyze the original query and provide some other alternated query plans with variation in computed metrics. Toad suggest several different query plan, out of which best three alternate plan abbreviate below as ALT1 (Alternate I), Alt II (Alternate II), Alt III (Alternate III) that shows variation in computed metrics are selected as shown in the following table.

Query Plan	Plan Cost	Logical Reads	Scan Rows	Sort Rows	Memory	Equivalent Query
Original	4	61	215	0	1424	Select emp.ename, dept.dname, emp.job, dept.deptno from EMP, Dept where emp.deptno = dept.deptno
Alt Plan I	3	68	20	0	1424	Select /*+ INDEX_JOIN(EMP) */ emp.ename, dept.dname, emp.job, dept.deptno from EMP, Dept where emp.deptno = dept.deptno
Alt Plan II	6	27	39	14	1424	Select /*+ NO_USE_NL(DEPT) */ emp.ename, dept.dname, emp.job, dept.deptno from EMP, Dept where emp.deptno = dept.deptno
Alt Plan III	7	39	228	0	1424	Select /*+ USE_HASH(DEPT,EMP) */ emp.ename, dept.dname, emp.job, dept.deptno from EMP, Dept where emp.deptno = dept.deptno

Table: Analysis of different query plans using Join Approach

The following chart shows the graphical analysis of different query plan with above computed metrics. The graph shows how a query behaved when executed with different suggested query plans. The different metrics taken in chart are logical reads, scanned rows, sorted rows and memory consumed by different query plans.



.

Figure2: Analysis of Different Join Plans

The following table shows the analysis of semi join approach and its alternate plan with different computed metrics. The different metrics that are studied are same as in above case.

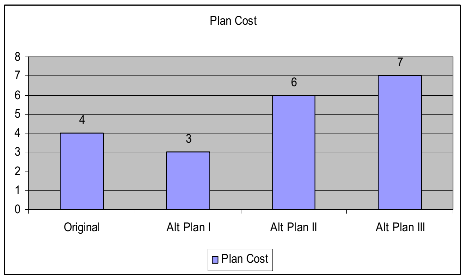


Figure4: Analysis of Plan Cost of different Join plan

Table 3: Analysis of Different Query Plans for semi join

Query Plan	**Plan Cost**	**Logical Reads**	**Scan Rows**	**Sort Rows**	**Memor y**	**Equivalent Query**
Original	4	75	209	0	1488	Select emp.ename, dept.dname, emp.job, dept.deptno from emp, dept where dept.deptno in (select deptno from dept where emp.deptno = dept.deptno)
Alt Plan I	3	114	14	32	1552	Select /*+ INDEX_JOIN(EMP) */ emp.ename, dept.dname, emp.job, dept.deptno from emp, dept where dept.deptno in (select deptno from dept where emp.deptno = dept.deptno)
Alt Plan II	4	32	23	0	1488	Select /*+ INDEX_JOIN(EMP) */ emp.ename, DEPT1.dname, emp.job, DEPT1.deptno FROM emp, dept DEPT1, (SELECT DEPT2.deptno COL1, DEPT2.deptno COL2 from dept DEPT2 GROUP BY DEPT2.deptno, EPT2.deptno) TEMP0 where emp.deptno = EMP0.COL1 AND DEPT1.deptno = TEMP0.COL2
Alt Plan III	18	74	332	14	1552	Select /*+ FIRST_ROWS */ emp.ename, DEPT1.dname, emp.job, DEPT1.deptno from emp, dept DEPT1 where (emp.deptno, DEPT1.deptno) IN (SELECT DEPT2.deptno + 0, deptno + 0 from dept DEPT2)

The following diagram shows the graphical analysis of different query plans as given by TOAD with an analysis of original query plan. The graph shows how a query behaved when executed with different suggested query plans. The different metrics taken in chart are logical reads, scanned rows, sorted rows and memory consumed by different query plans. The graph clearly suggests that the query shows significant change when executed with different query plans.

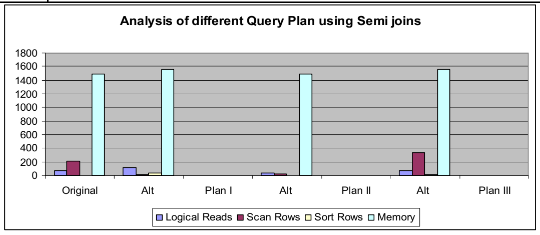


Figure 5: Analysis of different Query plans

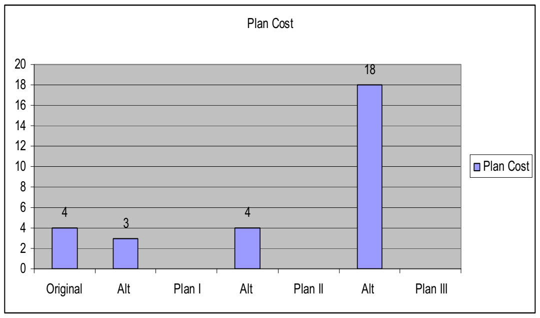


Figure 6: Analysis of Query Cost

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.

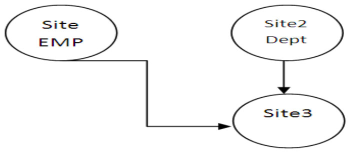


Figure7 : Data Transmission

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

14*51+29*5

=714+116

=830bytes

Total_Time (TM) =2$T_{Message}$+T_{TCost} (1,05,000Bytes)

Response_Time =Max ($T_{Message}$+T_{TCost}*714Bytes), $T_{Message}$+T_{TCost}*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.

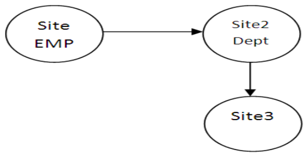


Figure8 Movement of Data

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

51*14+35*14

=714+490

=1204bytes

Total_Time (TM) =2$T_{Message}$+T_{TCost}(1204Bytes)

Response_Time =Max ($T_{Message}$+T_{TCost}*714Bytes), $T_{Message}$+T_{TCost}*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.

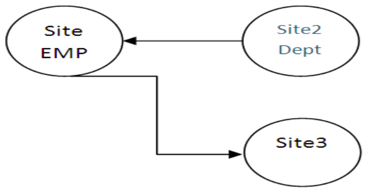


Figure9: Movement of Data

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

29*4+35*14

=116+490

=606bytes

Total_Time (TM) =$2T_{Message}+T_{TCost}$(606Bytes)

Response_Time =Max($T_{Message}+T_{TCost}$*116Bytes), $T_{Message}+T_{TCost}$*490Bytes)

Query Processing Using Semi Joins

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

Select emp.ename, dept.dname emp.job, dept.deptno from emp,dept where dept.deptno in (select deptno from dept where emp.deptno=dept.deptno);

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

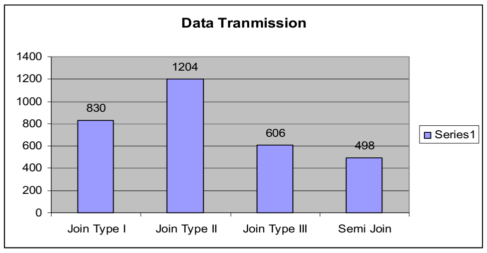
2*4=8Bytes Shipped from Site 2 to Site1

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) =$2T_{Message}+T_{TCost}$(498Bytes)

Response_Time =Max($T_{Message}+T_{TCost}$*8Bytes), $T_{Message}+T_{TCost}$*490Bytes)



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.

Conclusion

From the above analysis it is clear that Semi join does not show any improvement in the query execution in centralized database system. Rather the study show that when the query is executed using semi join it has higher query cost , input output cost and CPU cost as compare to when query is implemented using join approach. Further semi join approach in centralized database system does not help in reducing the accessing of bytes required for query execution. 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. However it is not mandatory in all cases. 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. 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.

Acknowledgment

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

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.