Abstract—In this paper an optimum method is introduced and described for running Global Query on distributed database system. We restate the general method of running Global Query in the environment distributed on database, and compare it with the suggested method. In the offered method, contrary to the general one, to find a solution to Global Query less interplay is required on the fragmented databases distributed among different systems.

Index Terms—global query; query decomposition; distributed databases; sub query.

I. INTRODUCTION

Progress in network as well as database systems technology led to development of DDS in 1970. Though many definitions are there for the system, none is standard. A DDS comprises of a distributed database managing system (DDBMS), a distributed database, and an interconnection network. Two different technologies govern a DDS, i.e. database system and computer network [1,2]. A DDS is a collection of several databases distributed logically in a computer network. DDBMS is a software authorizing management of distributed database and which makes a transparency in distribution for users. Problems one encounters with in database systems are excessive complications in the distributed setting. Firstly, it is possible that data is repeated in a distributed environment[3,4]. A distributed database may be designed for a whole database or a part of it set for different sites of computer networks. DDS is responsible to:

(1) choosing one of the saved copies of requested data for access in a retrieve case;

(2) ensuring in effect of an update reflected for each data and its copy.

Secondly, if failure is occurred in some sites or links while an update is running, the system should guarantee the effect and reflection on the data, and the system should be retrieved.

Thirdly, it is not possible for any site to have instantaneous information on right actions occurring in other sites. Also, the concurrence of transactions upon several sites is much more difficult than in the central database system. The present article gives an optimum method in running Global Query on distributed database system. Focus is on the end that the number of interactions on fragmented databases containing the results of sub-queries is less than the case for the general method in running sub-queries of Global Query to find a solution. Hereby, a share of running is dedicated to all the tables required to reach a general solution, with less works done in the suggested method relative to the usual method. The outline of the rest of this paper is as follows. Section 2 describes fragmentation in distributed database. Section 3 describes query decomposition. Section 4 describes general method in running global query. Section 5 presents suggested method in running global query. Section 6 describes comparison between general and suggested method. In the last section also will be discussed conclusion.

II. FRAGMENTATION IN DISTRIBUTED DATABASE

Subdividing a general relation R to fragments R1, R2, …Rn possessing enough information to recover the original relation R, and each saved in any site on the network computer is called fragmentation[5,6]. There are three strategies for fragmentations; horizontal: subdivision of a relation into fragments or tuples as in Figure. 1 is named a horizontal fragmentation.

Vertical: dividing a relation based on its characteristics into subsets or fragments as in Figure. 2 is called a vertical fragmentation.

Manuscript received April 2015
Mohammadjavad Hosseinpoor, Member of faculty in Department of Computer Engineering and Member of Young researchers and elite club, Estahban Branch, Islamic Azad University, Estahban,Iran.
Hossein Monem, Member of faculty in Information Technology Department Paramedical Sciences School Shiraz University of Medical Sciences, Shiraz, Iran.
Mortaza Zolfpour- Arokhlo, Member of faculty in Department of Computer Engineering, Sepidan Branch, Islamic Azad University, Sepidan,Iran.
A. Fragmentation Advantages

*It makes a parallel processing of data possible.

*The tuples are set where they are accessible repeatedly often.

When the tables are horizontally fragmented in a database, they must be “unioned” to regain the original table, whereas in the vertical case they are “joined” to recover the original table. Each of the fragments are distributed among these systems.

III. QUERY DECOMPOSITION

In a DDS, Query is propounded in a high level of global database, whilst the data necessary for the solution of Global Query may be located in different databases. In this case, Global Query should be decomposed to sub-queries which are run locally in different systems containing the result of Global Query [6,7]. After running of sub-queries in different fragmented databases the returned results are combined according to the rules of section 3 to obtain the solution. When a Global Query enters the system it would be sent to the Query Decomposition part to be divided into sub-queries which are to be run on relevant different fragmented tables. In the follow-up, the general method of running the sub-queries in DDS is studied along with the optimum method and they are compared.

IV. GENERAL METHOD[7]

We explain, through an example, the general method of running sub-queries obtained from Query Decomposition on DDS as well as their return of results to get the solution. The general tables in this article are according to Figure. 4 S, P, and SP with data thereof. Now, we subdivide the tables by the hybrid method into fragmented tables and distribute them between two databases set in two disjoint systems according to Figure. 5. Here, the tables S, P, and SP are divided, respectively into tables S1, S2, S3, S4, P1, P2, P3, P4, and SP1, SP2, SP3, SP4. The fragmented tables S1, S2, P1, P2, SP1, and SP2 are located in the first database, and S3, S4, P3, P4, SP3, and SP4 are positioned in the second. The combinations of fragmented tables to reach the original tables. as stated in section 3 are as follows:

\[
\text{s} = (s1 \cup s2) \text{Join} (s3 \cup s4)
\]

\[
\text{p} = (p1 \cup p2) \text{Join} (p3 \cup p4)
\]

\[
\text{sp} = (sp1 \cup sp2) \text{Join} (sp3 \cup sp4)
\]

The Global Query to be run on the DDS would be as:

\[
\text{GQ: select sname from s,sp where (s.s#=sp.s# and p.p#= sp.p# and p.pname='labtop' and price> 700$)}
\]

The way this query is run in the system through the general method is as follows:

First, the relevant tree is made as in Fig. 6. Replacing the fragmented tables of S, P, and SP with the original tables gives rise to the tree of Fig. 7 according to which the sub-queries are:

Subq1:

Select p# from p1 where (p1.pname='labtop')

Subq2:

Select p# from p2 where (p2.pname='labtop')

Subq3:

Select s#,p# from sp1 where (sp1.price> 700$)

Subq4:

Select s#,p# from sp2 where (sp2.price> 700$)

Subq5:

Select s#,sname from s1

Subq6:

Select s#,sname from s2

All these sub-queries correspond to the first system and the first database, so are entirely sent to the first system. After their running the returned solutions are combined as in tree of Figure. 6 to give the principal solution Sname=javadi. The details are that after returning of results of sub-query running to the first database, first the results of subq1 and subq2 are “unioned”. Second, those of subq3 and subq4 are “unioned”. Then the Union happens for those of subq5 and subq6. The results of second and third Unions Join to each other, and the outcome is Joined with the first Union to reach at last the main result sname=javadi. The process is illustrated in Figure. 7.
V. SUGGESTED METHOD

In this method the tree of the query is made as in the general method according to Fig. 7, and then the sub-queries are produced as before from subq1 to subq6, however, they are not run immediately, but the sub-queries of a database are “Joined” and/or “Unioned” with each other and then run on the systems. Since all the sub-queries correspond to one database they are “joined” or “unioned” based on the suggested method prior to be sent to the database. In the method subq1, subq3, and subq5 combine to yield the following subquery:

**Subq1:**

\[
\text{Select sname from s1,p1,sp1 where ( s1.s#=sp.s# and p1.p#=sp.p# and p1.pname = 'labtop' and sp1.price> 700$ )}
\]

Also, combination of subq2, subq4, and subq6 makes the following subquery:

**Subq2:**

\[
\text{Select sname from s2,p2,sp2 where ( s2.s#=sp.s# and p2.p#=sp.p# and p2.pname = 'labtop' and sp2.price> 700$ )}
\]

Now, since these two sub-queries relate to the first database, and the tables S1, S2, P1, P2, SP1, and SP2 are in the sub-query, then the two sub-queries could be “Unioned” as:

**Subq:**

\[
\text{Select sname from s1,p1,sp1 where ( s1.s#=sp.s# and p1.p#=sp.p# and p1.pname = 'labtop' and sp1.price> 700$ )}
\]

Union

\[
\text{Select sname from s2,p2,sp2 where ( s2.s#=sp.s# and p2.p#=sp.p# and p2.pname = 'labtop' and sp2.price> 700$ )}
\]

Running the sub-query on the first database gives the desired result sname=javadi.

VI. COMPARISON BETWEEN GENERAL AND SUGGESTED METHODS

In this section we compare the two methods of running Global Query. The general method runs the six sub-queries on the related tables in the database. Then, the returned results based on the produced tree render three Unions at the first stage, one Join at the second stage, and another Join at the third reach the main solution. So
the number of jobs = running of 6 sub-queries + 3 Unions + 2 Joins.

If we undertake the suggested method, after subdivision of Global Query to six sub-queries these are not run on the relevant tables, but the tactic is that the sub-queries of a database are combined to each other according to the produced tree. Hence, because all the six sub-queries are in the first database, only one sub-query is made as stated in section 6 and run in the database. So

the number of jobs here = running of 2 sub-query + 1 Union.

As it could be seen, the number of operations in the suggested method is much less than the case for the general method, and the less interactions cause reduction of run time.

VII. CONCLUSION

In this article an optimum method was presented for running of Global Query on a distributed database system. According to the results obtained from the general and suggested methods, it was inferred that the suggested method is more favorable as regards the number of operations whence run time. As soon as the suggested method is used, both the number of running sub-queries and the Union-Join operations would be reduced.

REFERENCES


