2017 8th International Conference on Information Technology (ICIT)

A comparative Analysis of Data Fragmentation in Distributed Database

Asma H. Al-Sanhani, Amira Hamdan, Ali B. Al-Thaher, Ali Al-Dahoud
Faculty of Science & I.T., Al-Zaytoonah University of Jordan, P.O.Box 130,.Amman (11733), Jordan.
asma.hussein.san@gmail.com, amira.hamdan922@yahoo.com, ali_Bassam_AlThaher@yahoo.com,
aldahoud@zuj.edu.jo

Abstract—**Distributed database technology is supposed to have a remarkable impact on data processing in the next years. This paper overviews Data Fragmentations in distributed database system. We discussed in briefly distributed database environment, fragmentations and distributed databases design. We compared between the horizontal fragmentations, vertical fragmentations, and mixed fragmentations. The Correctness rules of fragmentation have been used; the best type of Data Fragmentations for distributed databases design has been suggested.**

Keywords—Distributed database, Fragmentations, Horizontal Fragmentations (HF), Vertical Fragmentations (VF), and Mixed Fragmentations (MF).

I. INTRODUCTION

Networks of computers are found in everywhere. The Internet is one, as are the many networks of which it is composed. Mobile phone networks, corporate networks, factory networks, campus networks, home networks, in-car networks – all of these type of networks, both separately and in combination, share the essential characteristics that make them relevant subjects for study under the heading distributed systems [1]. Distributed systems are an important development in computing technology, which is concerned with the delivery of constantly expanding data to points of query. Collections of data in the forms of partitions or fragments can be distributed or replicated over multiple physical locations [2]. When an organization is geographically dispersed, it may choose to store its databases on a central database server or to distribute them to local servers (or a combination of both). A **distributed database** is a single logical database that is spread physically across computers in multiple locations that are connected by a data communications network. A distributed database (DDB) is a collection of data that logically belongs to the same system but is spread over the sites of a computer network. Distribution of data is a collection of fragmentation, replication and allocation processes. The sites of the distributed database may be allocated in the same space and have the same network address but the communication between them is done over a network instead of sharing memory.

As communication technology, software and hardware, advance rapidly and the equipment prices falls every day, developing DDBS become more and more feasible. Design of efficient distributed database is one of the hot topics in database and information technology areas. DDBS is the integration of DDB and DDBMS. This integration achieved by merging the database and networking technologies, or it can be defined as, a system that runs on a collection of nodes that do not have shared memory, and give the impression to the user as working in a single machine [4]. Figure 1 shows the Environment of the distributed Database and how it works with communication network.

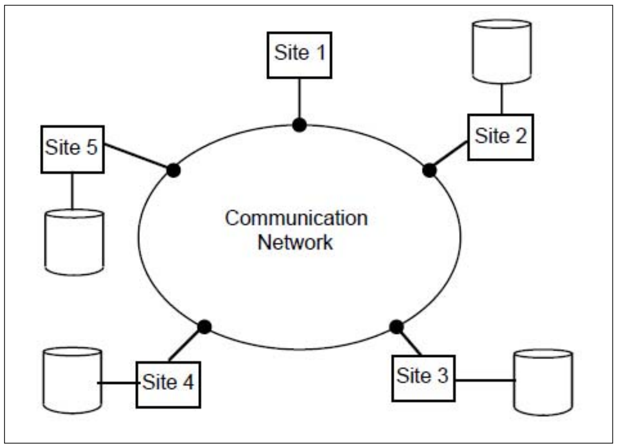

Figure 1. A distributed Database Environment

The distributed database design process consists of three phases, namely, initial design, redesign and materialization of the redesign. The initial design consists of the fragmentation and allocation algorithms that minimize the total transaction processing cost for a given set of transactions. The redesign problem consists of generating new fragmentation and allocation schemes from current fragmentation and allocation schemes by taking into consideration the logical and physical changes in the distributed database environment. The materialization of redesign is accomplished by a sequence of operations to materialize the new fragmentation and allocation scheme from the current design [5]. The rest of this paper is organized as follows, related work is discussed in Section two. Section three provides a discussion of **Fragmentations.** In Section four, we present Data Fragmentations with Correctness Rules of Fragmentation followed by our conclusions and future work in Section five.

II. Related Works

The distributed data processing is an effective way to improve reliability, availability and performance of a database system. Most of the current research deals with coming up with a better set of algorithms for fragmentation schemes and allocation schemes. Many researchers have been studied the fragmentation in a distributed database system. Seung-Jin Lim (2016) presented different approaches for vertical fragmentation of relations that are referenced by rules and an allocation strategy for rules and fragment in a distributed deductive database system (DDDBS). The potential advantages of the proposed fragmentation and allocation scheme include maximal locality of query evaluation and minimization of communication cost in a distributed system. It also formulated the mathematical interpretation of the proposed vertical fragmentation and allocation algorithm.

Another researcher, (Van Nghia Luong et el.2015) proposed an algorithm that built the initial equivalence relation based on the distance threshold. This threshold was also based on knowledge-oriented clustering techniques for both of horizontal and vertical fragmentation. Similarity measures used in the algorithms were the measures developed from the classical measures. Experimental results carrying on the small data set matched fragmented results based on the classical algorithm. Execution time and data fragmentation significantly reduced while the complexity of our algorithm in the general case was stable.

(Akashkumar Patel, et el.2014) used fragmentation methods. There are several fragmentation methods reviewed in this article. They utilized the resources and thus it must select an accurate and efficient fragmentation methodology to enrich the power of distributed database system.

On the other side, (**Bhuyar**, et el.2012) had introduced a fragmentation technique that could be applied at the first stage as well as in last stages of distributed database system. The Allocation of fragments was done also in the algorithm. Result illustrated that the proposed technique could be used to solve the initial fragmentation problem of the relational databases for distributed systems decently.

Nicoleta - Magdalena Iacob (2011) concentrated on data allocation problem with the aim to assure an optimal distribution of data in the process of the distributed database design in correlation with data fragmentation. They analyzed the cost of fragmentation and replication. Distributed databases have appeared as a necessity, because they improved availability and reliability of data and assured high performance in data processing by allowing parallel processing of queries, but also reduced processing costs.

Adrian Runceanu (2007) was presented a general approach of the vertical fragmentation issue of the dates from a distributed database. The implementation of a heuristic algorithm conceived before that used an objective function who took over information about the administrated dates in a distributed database and it evaluated all the scheme of the database vertical fragmentation.

Another research (HABABEH, et al. 2003) developed a strategy for distributed database design that was simple and useful to achieve the objectives of data fragmentation, allocation, and replication. It had been designed to fragment and allocate data in a distributed relational database system, minimized the communication cost between sites, and improved the performance in a heterogeneous network environment system.

(Navathe, et al. 1995) presented algorithms for generating candidate vertical and horizontal fragmentation schemes and proposed a methodology for distributed database design using these fragmentation schemes. When applied together these schemes form a grid. This grid consisting of cells was then merged to form mixed fragments to minimize the number of disk accesses required to process the distributed transactions. They had implemented the vertical and horizontal fragmentation algorithms and developed a Distributed Database Design Tool to support the proposed methodology.

III. FRAGMENTATIONS

Distributed database system design primary concern is to make fragmentation of classes in case of object oriented databases, or the relations in case of relational database, replication and allocation of the fragments in different sites of the distributed system, and local optimization in each site. Fragmentation is a technique to split a single class or relation of a database into two or more partitions, also; the combination of the partitions supports the original database without any loss of information. That means, it reduces the amount of irrelevant data accessed by the applications of the database, thus reducing the number of disk accesses [4]. Each fragment may be stored at any site over a computer network. Fragmentation aims to improve:

a. Reliability.
b. Performance.
c. Balanced storage capacity and costs.
d. Communication costs.
e. Security.

The following information is used to decide fragmentation:

- Quantitative information: cardinality of relations, frequency of queries, site where query is run, selectivity of the queries, etc.
- Qualitative information: predicates in queries, types of access of data, read/write, etc.

The Advantages and Disadvantages of Fragmentation are shown below:

Advantages	**Disadvantages**
Since data is stored close to the site of usage, efficiency of the database system is increased.	When data from different fragments are required, the access speeds may be very high.
Local query optimization techniques are sufficient for most queries since data is locally available.	In case of recursive fragmentations, the Job of reconstruction will need expensive **techniques.**
Since irrelevant data is not available at the sites, security and privacy of the database system can be maintained.	Lack of back-up copies of data in different sites may render the database ineffective in case of failure of a site.

Table 1. Fragmentation; Advantages and disadvantages

Fragmentation types

Fragmentation can be horizontal, vertical or mixed.

1. *Horizontal fragmentation*
   Horizontal fragmentation (HF) allows a relation or class to be partitioned into disjoint tuples or instances. Each fragment is stored at a different node, and each fragment has unique rows. However, the unique rows all have the same attributes (columns). Intuition behind horizontal fragmentation is that Every site should hold all information that is used to query at the site and the information at the site should be fragmented so the queries of the site run faster [4].

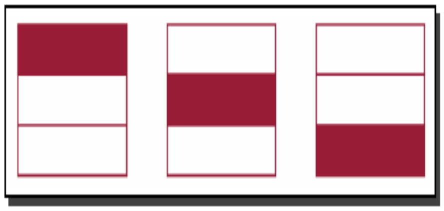

Figure 2. Horizontal fragmentation

2. *Vertical Fragmentation*(VF)*:*
   A class or relation in VF will be partitioned into separate sets of columns or attributes except the primary key. Each set must include the primary key attribute(s) of the table. This arrangement can make sense when different sites are responsible for processing different functions involving an entity.
   The main target of vertical fragmentation is to divide a relation into a set of smaller relations, so that in only one fragment many of the applications will run [4].

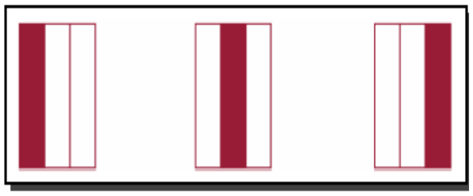

Figure 3. Vertical fragmentation (VF)

3. *Mixed fragmentation:*
   Combination of Horizontal and vertical fragmentations give the mixed fragmentations (MF). The table in this type of fragmentation scheme is divided into blocks that based on the needed requirements. Each fragmentation may be allocated on a specific site. Mixed fragmentation is the most complex one; it needs more management. Mixed fragmentation contains a vertical fragment followed schema will not be sufficient to satisfy the requirements by a vertical fragmentation, or a horizontal fragmentation followed by a vertical fragmentation [4].

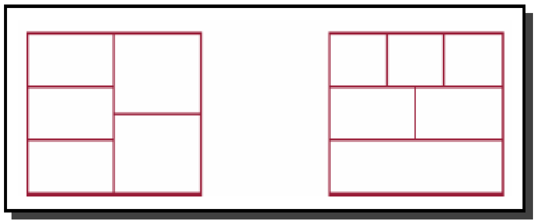
Figure 4. Mixed Fragmentation

IV. DATA FRAGMENTATIONS

Data fragmentation allows breaking a single object into two or more fragments or segments. The object might be a system database, table, or a user's database. The fragment could be stored at any site over a computer network.

To illustrate the fragmentation strategies, let's use the Human Resources table for the first type that called Horizontal Fragmentation depicted in Table 2 below, intuition behind horizontal fragmentation:

1. Every site should hold all information that used to query the site.
2. The information at the site should be fragmented so that the queries of the site run faster.

For example, Human Resources table is having 1000 records can be horizontally fragmented into ten fragments, each fragment having 100 unique records.

EMP_ID	F_NAME	JOB	DEPT_ID	Salary
Fragment 1				
Fragment 2				
. .				
Fragment n				

Table 2. Horizontal Fragmentation

Secondly, vertical fragmentation is fragmenting of table into columns known as set or site, where every site must have at least one column in common such as the primary key attribute column (so that when the fragmented sites when needed can again be formed to a whole (parent) table using the common column). Vertical fragmentation of a relation R produces fragments R1,R2, . . . , each of which contains a subset of R's attributes.

For example, in the Human Resources table; the attributes as EMP_ID (PRIMARY KEY), F_NAME, JOB, SALARY, and DEPT_ID could be Vertically fragmented into two sites such as site1 and site2 as shown in table 3.

EMP_ID	F_NAME	JOB	EMP_ID	SALARY	DEPT_ID
Fragment 1			Fragment 2		

Table 3. Vertical Fragmentation

Finally, mixed fragmentation refers to a combination of horizontal and vertical strategies. In other words, a table may be divided into several horizontal subsets (rows), each one having a subset of the attributes (columns). Mixed fragmentation needs two-steps procedures. First of all, horizontal fragmentation is introduced for each site. The horizontal fragmentation yields the subsets of Human Resources tuples (horizontal fragments) that are located at each site. Vertical fragmentation is used within each horizontal fragment to divide the attributes. For example, table 4 shows how to fragment data by mixed fragmentation.

EMP_ID	F_NAME	JOB	DEPT_ID	Salary
Fragment 1			Fragment 2	
...		Fragment n		

Table 4. Mixed Fragmentation

V. CORRECTNESS RULES OF FRAGMENTATION

1. Completeness

- Decomposition of relation R into fragments R1,R2, . . . ,Rn is complete iff each data item in R can also be found in some Ri.

2. Reconstruction

- If relation R is decomposed into fragments R1, R2, . . . , Rn, then there should exist some relational operator ∇ that reconstructs R from its fragments, i.e., R = R1∇. . .∇Rn:

∗ Union to combines horizontal fragments.
∗ Join to combine vertical fragments.

3. Disjointness

- If relation R is decomposed into fragments R1, R2, . . . , Rn and data item d_i appears in fragment Rj , then di should not appear in any other fragment R_k, $k \neq j$ (exception: primary key attribute for vertical fragmentation)
  - ∗ for horizontal fragmentation, data item is a tuple.
  - ∗ for vertical fragmentation, data item is an attribute [4].

For each fragment of a relation R, the comparison of fragmentation strategies is shown in table 5.

I. Condition C = True (all tuples are selected).
II. List (L = ATTRS(R)) = True (all attributes are included in the list).

	VF	HF	MF
C	True (all tuples)	False (not all tuples)	False (not all tuples)
L	False (not all columns)	True (all columns)	False (not all columns)

Table 5. Comparison of fragmentation strategies

VI. Conclusion

In this paper; an introduction about distributed database with distributed database design has been presented. Data Fragmentation with all of its types has been explained. Comparison of fragmentation strategies is shown for each fragment of a relation R. Advantages and disadvantages for each type also have been presented.

Finally; this work could be of a great help to new researchers who would like to work with distributed databases.

References

[1] J. D. T. K. G. B. George Coulouris, DISTRIBUTED SYSTEMS, United States of America: Addison-Wesly, 2012.

[2] W. A. S. A.-A. a. A. Y. Azzam Sleit, "A Dynamic Object Fragmentation and Replication Algorithm In Distributed Database Systems," American Journal of Applied Sciences, Vols. 613-618,, 2007.

[3] P. V. M. Tamer Özsu†, "DISTRIBUTED DATABASE SYSTEMS: WHERE ARE WE NOW?," IEEE Computer, Vols. 24, No. 8,, 1991.

[4] D. P. A. Ms. P. R. Bhuyar, "Horizontal Fragmentation Technique in Distributed Horizontal Fragmentation Technique in Distributed Horizontal Fragmentation Technique in Distributed H Horizontal Fragmentation Technique in Distributed Database," International Journal of Scientific and Research Publications, vol. 2, no. 5, 2012.

[5] K. K. M. R. Shamkant B. Navathe, "A Mixed Fragmentation Methodology For Initial Distributed Database Design," Journal of Computer and Software Engineering, Vols. 395-426., 1995.

[6] N. -. M. Iacob, "Fragmentation and Data Allocation in the Distributed Environment," 2011.

[7] A. Runceanu, "TOWARDS VERTICAL FRAGMENTATION IN DISTRIBUTED DATABASES," 2007.

[8] S.-J. a. Y.-K. N. Lim, "Vertical fragmentation and allocation in distributed deductive database systems," Information Systems, 1997.

[9] I. O. N. I. C. H. O. L. A. S. B. a. M. U. T. H. U. R. Hababeh, "AN INTEGRATED STRATEGY FOR DATA FRAGMENTATION AND ALLOCATION IN A DISTRIBUTED DATABASE DESIGN," ta Fragmentation and Allocation in A Distributed Database Design."

Proceedings of the International Conference on Information Technology and Natural Science (ICITNS), 2003.

[10] R. H. V. D. Akashkumar Patel1, "A Review on Fragmentation Techniques in Distributed Database," *International Journal of Modern Trends in Engineering and Research,* 2014.