



International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

www.iasir.net

A Study on Distributed Database System

Swati Gupta

Assistant Professor, Department of Computer Science,
Amity University Haryana, Gurgaon, Haryana, INDIA.

Abstract: A Database is a collection of data describing the activities of one or more related organizations with a specific well defined structure and purpose. A Database is controlled by Database Management System (DBMS) by maintaining and utilizing large collections of data. A Distributed System is the one in which hardware and software components at networked computers communicate and coordinate their activity only by passing messages. In short a Distributed database is a collection of databases that can be stored at different computer network sites. This paper presents an overview of Distributed Database System along with their advantages and disadvantages. This paper also provides various aspects like replication, fragmentation and various problems that can be faced in distributed database systems.

Keywords: Deadlock, Distributed Database Management System, Fragmentation, Replication.

I. Introduction

The rapidly growing business environment has a need for distributed database which is reliable, scalable and easily accessible [2, 3]. Distributed database systems provide an improvement on communication and data processing due to its data distribution throughout different network sites. DDB not only makes data access faster, but it has resolve the problem of single-point of failure as the data is being distributed across multiple sites. In short a distributed database can be defined as a database environment consisting of a collection of data with different parts under the control of separate DBMSs running on independent computer systems [7,8]. All the computers are interconnected with the network and each system has an autonomous processing capability serving local applications. Each system participates, in the local as well as in the execution of one or more global applications. Such applications require data from more than one site. The distributed nature of the database is hidden from the users. A distributed database management system (DDBMS) is the software that manages the DDB and consists of a collection of sites, each of which maintain a local database system and provides access mechanisms that make the distribution transparent to the user[10,11].

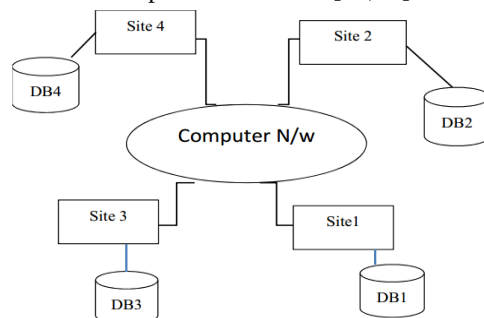


Fig. 1: A distributed Database

II. Types of Distributed Database

There are two types of distributed database

I) Homogeneous distributed database

- All sites have identical software
- The sites are aware of each other and agree to cooperate in processing user requests. All the site surrenders their part of its autonomy in terms of right to change schemas or software
- It appears to user as a single system

II) Heterogeneous distributed database

- Different sites may use different schemas and software
- Sites may not be aware of each other
- Providing limited facilities for cooperation in processing the transaction.

III. Advantages of Distributed Databases

The distributed database offers several advantages:

- Robust: A problem in one part of the database will not affect other part of database.
- Security: Access to the database can be restricted to only authorize user.
- Network traffic is reduced, thus reducing the bandwidth cost
- High Performance: Queries and updates are largely local so that there is no network bottleneck.

IV. Disadvantages of Distributed Databases

The following are the various disadvantages of distributed databases [9, 10]:

- Complexity: A distributed database is more complicated to setup and maintain as compared to central database system.
- Security: There are many remote entry points to the system compared to central system leading to security threats.
- Data Integrity: In distributed system it is very difficult to make sure that data and indexes are not corrupted.
- In distributed database systems, data need to be carefully placed to make the system as efficient as possible.
- Distributed databases are not so efficient if there is heavy interaction between sites.

V. Distributed Database: Characteristics

The distributed database consists of the following properties:

A) Fragmentation

Fragmentation [11]: Fragmentation consists of breaking a given relation into smaller relations or fragments and then storing the fragments, possibly at different sites.

Fragmentation of data in distributed database has three major advantages:

- Efficiency: Data are stored close to where they are used and separate from other data used by other users or applications.
- Local optimization: Data can be stored to optimize performance for local access.
- Ease of querying: Combining data across horizontal partitions is easy because rows are simply merged by unions across the partitions.

There are three types of fragmentation namely–

A1. Horizontal fragmentation

Horizontal fragmentation (HF)[1,2] allows a relation to be partitioned into disjoint tuples or instances. The given relation is fragmented row wise. The purpose of such type of fragmentation is quick and easy access of the data so that query when fired from the given site runs faster. Horizontal fragmentation is defined as selection operation, $\sigma_p(R)$.

For example, the following relation

EMPLOYEE (eid, fname, lname, site, pos, salary)

Eid	Fname	Lname	Site	Pos	Salary
Fragment1					
Fragment2					
Fragment3					

Fig. 2: Horizontal fragmentation

A2. Vertical Fragmentation

Vertical fragmentation (VF) [4,5] allows a relation or class to be partitioned into disjoint sets of columns or attributes except the primary key. Each vertical fragment must include the primary key attribute(s) of the table.

In vertical fragmentation we fragment the given relation R into small fragments R1,R2..Rn, this fragment are being formed column wise. Vertical fragmentation is defined using the projection operation of the relational algebra:

$\Pi_{A1,A2,...,An}(R)$

Eid	Fname	Lname	Site	Eid	Pos	Salary
Fragment1				Fragment2		

Fig. 3: Vertical fragmentation

A3. Hybrid fragmentation

Combination of horizontal and vertical fragmentations is mixed or hybrid fragmentations (MF)[10,15]. In this type of Fragmentation scheme, the table is divided into arbitrary blocks, based on the needed requirements. Each

fragmentation can be allocated on to a specific site. This type of fragmentation is the most complex one, which needs more management, in most cases simple horizontal or vertical fragmentation of DB applications. Mixed Fragmentation is defined using the selection and projection operations of relational algebra:

$$\Pi_p(\sigma_{A_1, \dots, A_n}(R))$$

$$\sigma_{A_1, \dots, A_n}(\Pi_p(R))$$

Salary	Eid	Fname	Lname	Site	Pos
Fragment1				Fragment2	
Fragment3		Fragment4			

Fig. 4: Mixed Fragmentation

B) Replication [12, 13]:

In Replication several copies of a relation are stored at different sites. Replication will help in increasing reliability, locality and performance. Various advantages of replication are as follows:

- Reliability: If one of the sites containing the relation (or database) fails, a copy can always be found at another site without network traffic delays
- Fast response: Each site that has a full copy can process queries locally, so queries can be processed rapidly.
- Possible avoidance of complicated distributed transaction integrity routines:

Replicated databases are usually refreshed at scheduled intervals, so most forms of replication are used when some relaxing of synchronization across database copies is acceptable.

There are two types of replication which are as follows:

1. Synchronous Replication: All copies of a modified relation (fragment) must be updated before commit. Here, the most up to date value of an item is guaranteed to the end user.
2. Asynchronous Replication: Asynchronous replication allows different copies of the same object to have different values for short periods of time. Data is updated after a predefined interval of time.

VI. Problems in Distributed Database Systems

One of the major problems in distributed systems is deadlock. A deadlock is a state of the system where a set of processes request resources that are held by other processes in the set and none of the process can be completed [14, 15, 16] thereby creating a circular wait condition.

If the sequence of the allocations of resources to the processes is not controlled, deadlocks can occur. Hence we focus on deadlock detection and removal.

- Deadlock Detection
In order to detect deadlocks, in distributed systems, deadlock detection algorithm must be used[15,16]. Each site maintains a local wait for graph. If there is any cycle in the graph, there is a deadlock in the system. Even though there is no cycle in the local wait for graph, there can be a deadlock. This is due to the global acquisition of resources. In order to find the global deadlocks, global wait for graph is maintained. This is known as centralized approach for deadlock detection. The centralized approach to deadlock detection, while straightforward to implement, has two main drawbacks. First, the global coordinator becomes a performance bottleneck, as well as a single point of failure. Second, it is prone to detecting non-existing deadlocks, referred to as phantom deadlocks.
- Deadlock Recovery
Deadlock always involves a cycle of process creating a circular wait condition. The general approach for deadlock recovery is process termination. In this method, nodes and edges of the resource graph are eliminated.

In Process Termination, the simplest algorithm is to terminate all processes involved in the deadlock. This approach is unnecessarily wasteful, since, in most cases, eliminating a single process is sufficient to break the deadlock. Thus, it is better to terminate processes one at a time, release their resources, and check at each step if the deadlock still persists. Before termination of process following parameters need to be checked: (a) The priority of the process, (b) The cost of restarting the process, and (c) The current state of the process

VII. Conclusion

In the current scenario of the fast changing world, distribution of data became the necessity. Distribution of data has its own advantages and disadvantages. This paper presents a complete review on distributed databases. It is clear from the study that distribution of data involves the problem of deadlock. We need to find out the methods

by means of which process can access the data which can leads to minimization of deadlock and thus resulting in proper utilization of resources.

References

- [1] www.businessdictionary.com/definition/database.html
- [2] en.wikipedia.org/wiki/Database
- [3] ib2012itgs.wikispaces.com/file/.../ITGS%20databasde%20homework.pdf.
- [4] www.webopedia.com/TERM/D/distributed_database.html
- [5] www.inf.unibz.it/dis/teaching/DDB/ln/ddb01.pdf
- [6] www.teachict.com/as_a2_ict.../A2.../distributed_database/
- [7] www.webopedia.com/TERM/D/distributed_database.htm
- [8] wps.pearsoned.co.uk/wps/media/objects/.../Chapter%2012_WEB.pdf
- [9] https://www.dlsweb.rmit.edu.au/toolbox/Database/.../dist_sys_def.htm
- [10] www.csc.liv.ac.uk/~dirk/Comp332/COMP332-DDB-notes.pdf 214 Parul Tomar & Megha
- [11] <http://stackoverflow.com/questions/5777234/horizontal-vs-verticalfragmentation-in-distributed-database-management-systems>.
- [12] www.cse.iitb.ac.in/~sudarsha/db-book/slide-dir/ch22.ppt
- [13] Xiangning Liu, Bharat K. Bhargava, "Data Replication in Distributed Database Systemsover Large Number of Sites", Computer Science Technical Reports. Paper 1229
- [14] en.wikipedia.org/wiki/Deadlock
- [15] X. M. Chandy and J. Misra, "A Distributed Algorithm for Detecting Resource Deadlocks in Distributed Systems" in ACM, 1982.
- [16] B. M. M. Alom, F. Henskens, and M. Hannaford, "Deadlock Detection Views of Distributed Database," in International conference on Information Technology & New Generartion (ITNG- 2009) Las Vegas, USA: IEEE Computer Society, 2009.