CONTENTS

Syllabus Advanced Database System
Semester I

Unit I: Distributed Database Concepts
Definition of Distributed databases and Distributed Database Management System (DDBMS), Distributed transparent system. DDBMS Architecture: DBMS standardization, Global, Local, External, and Internal Schemas, Architectural models for DDBMS. Distributed database design: Design problem of distributed systems, Design, strategies (top-down, bottom-up), Fragmentation, Allocation and replication of fragments. Query Processing Overview, Query Optimization.

Unit II: Transaction Processing in Distributed databases and Parallel databases
Transaction Management: Definition and examples, formalization of a transaction, ACID properties, classification of transaction. Concurrency Control: definition, execution schedules, examples, locking based algorithms, timestamp ordering algorithms, deadlock management. DBMS reliability: Definitions and Basic Concepts, Local Recovery Management, In-place update, out-of-place update, Distributed Reliability Protocols, Two phase commit protocol, Three phases commit protocol. Parallel Database System: Definition of Parallel Database Systems. Parallel query evaluation: Speed up and scale up, Query Parallelism: I/O Parallelism (Data Partitioning) Intra-query Parallelism, Inter –Query Parallelism, Intra Operation Parallelism, Inter Operation Parallelism.

Unit III: Object Oriented, Temporal and Spatial Databases:
Object Oriented Database: Object Identity, Object structure, Type Constructors, Encapsulation of Operations, Methods, Persistence, Type and Class Hierarchies, Inheritance, Complex Objects, Object-oriented DBMS , Languages and Design: ODMG Model, Object Definition Languages (ODL), Object Query Languages (OQL). Temporal and Spatial Database: Introduction to Temporal Database: Time ontology, structure, and granularity, Temporal data models, Temporal relational algebras. Introduction to Spatial Database: Definition, Types of spatial data, Geographical Information Systems (GIS), Conceptual Data Models for spatial databases, Logical data models for spatial databases: rastor and vector model. Physical data models for spatial databases: Clustering methods (space filling curves), Storage methods (R-tree). Query processing.

Unit IV: Deductive, Active, Multimedia and XML Databases
Deductive Database: Introduction to recursive queries, Datalog Notation, Clause Form and Horn Clauses, Interpretation of model: Least Model semantics, The fixed point operator, safe Datalog program, recursive query with negation. Active Database: Languages for rule specification: Events, Conditions, Actions. XML and Database: Structure of XML Data, XML Document Schema, Querying and Transformation, Storage of XML Data. Introduction to multimedia database systems.

Text book:

- Distributed Database; Principles & Systems By Publications, Stefano Ceri and Giuseppo Pelagatti,, McGraw-Hill International Editions (1984)
- Database Management Systems, 3rd edition, Raghu Ramakrishnan and
- Johannes Gehrke, McGraw-Hill (2002).
- Fundamentals of Database Systems, 6thEdition, Elmasri and Navathe, Addison. Wesley (2003).
- Unifying temporal data models via a conceptual model, C.S. Jensen, M.D. Soo, and R.T. Snodgrass: Information Systems, vol. 19, no. 7, pp. 513-547, 1994.
- Spatial Databases: A Tour by Shashi Shekhar and Sanjay Chawla, Prentice Hall, 2003 (ISBN 013-017480-7)
- Principles of Multimedia Database Systems, Subramanian V. S. Elsevier Publishers, 2013.

References:

- Principles of Distributed Database Systems; 2nd Editied By M. Tamer Ozsu and Patrick Valduriez, Person Education Asia.
- Database System Concepts, 5^{th} edition, Avi Silberschatz , Henry F. Korth , S. Sudarshan: McGraw-Hill (2010)
- Database Systems: Concepts, Design and Applications, 2nd edition, Shio Kumar Singh, Pearson Publishing, (2011).
- Multi-dimensional aggregation for temporal data. M. Böhlen, J. Gamper, and C.S. Jensen. In Proc. of EDBT-2006, pp. 257-275, (2006).
- Moving objects databases (chapter 1 and 2), R.H. Güting and M. Schneider: Morgan Kaufmann Publishers, Inc., (2005)
- Advanced Database Systems, (chapter 5, 6, and 7), Zaniolo et al.: Morgan Kaufmann Publishers, Inc., (1997).

Module - I

1

DISTRIBUTED DATABASE CONCEPTS

Unit Structure

1.0 OBJECTIVE:

After going through this unit, you will be able to:

- understand what Distributed database is.
- define what is Distributed Database Management System
- describe features of DDBMS its advantages and disadvantages
- Illustrate Distributed transparent system
- Classify Distributed transparent System.

1.1 INTRODUCTION:

For appropriate working of any business/organisation, there's a requirement for a well-organised database management system. In the past databases used to centralize in nature. But, with the growth of globalization, organisations lean towards expanded crosswise the world.

Because of this reason they have to choose distributed data instead of centralized system. This was the reason concept of Distributed Databases came in picture.

Distributed Database Management System is a software system that manages a distributed database which is partitioned and placed on different location. Its objective is to hide data distribution and appears as one logical database system to the clients.

1.2 DISTRIBUTED DATABASE CONCEPT:

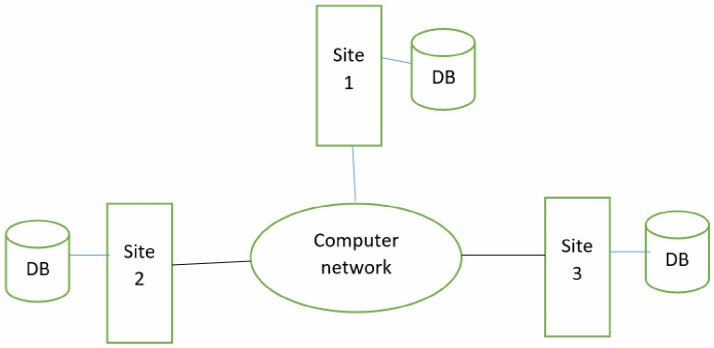
Distributed Database is database which is not restricted to one system only. It is a group of several interconnected databases. These are spread physically across various locations that communicate through a computer network. Distributed Database Management System (DDBMS) manages the distributed database and offers mechanisms so as to make the databases clear to the users. In these systems, data is intentionally distributed among multiple places so that all computing resources of the organization can be optimally used.

1.2.1. Definition of Distributed Databases and Distributed Database Management System (DDBMS)

The concept that is most important to the DDBMS is location clearness, meaning the user should be unaware of the actual location of data.

"A distributed database management system (DDBMS) can be defined as the software system that permits the management of the distributed database and makes the distribution transparent to the users.":- ***M. Tamer Özsu***

A Distributed Database Management System allows end users or application programmers to view a pool of physically detached databases as one logical unit. In another word, we can say distributed database is, where different data stored among multiple locations but connected via network, and for user it represent as a single logical unit.



Distributed Database Management System

1.2.1.1 Features ofDistributed Database Management System

Some features of Distributed Database Management system are as follows:

- DDBMS software maintain CRUD (create, retrieve, Update, Delete) functions.
- It covers all application areas where huge volume of data are processed and retrieved simultaneously by n number of users.
- It ensure that data modified at any location update universally.
- It ensures confidentiality and data integrity which is important feature in transaction management.
- It can handle heterogeneous data platforms.

1.2.1.2 Advantages of Distributed Database Management System:

Some of the advantages of DDBMS are as follows:

- **Reliable:**

  Incase of centralized DBMS if database fails entire system comes to a halt whereas in DDBMS when a component fails may be reduce performance but it will not stop fully.

- **Easy Expansion**

  In centralized database system if system needs to be expanded, the implementation require extensive efforts and interruption in the existing functionality. However in DDBMS no disturbance in current functioning.

- **Faster Response**

  In centralized database all queries are passing through central data repository because of that response time is more although in DDBMS data is distributed in well-organized, so it runs faster response onqueries.

1.2.1.3 Disadvantages ofDistributed Database Management System:

- **Complex and Expensive**

  DDBMS provides data transparency and work on different sites so it may require complex and expensive software for proper working.

- **Overheads**

  Simple and complex operation and queries may require large communication and calculation. Responsiveness is largely dependent upon appropriate data distribution. Improper data distribution often leads to slow response to user requests.

- **Integrity**

  As data is on multiple sites it may create problem in updating data and maintaining data integrity.

1.2.2. Reasons to Boosting DDBMS

The following Reasons inspire moving towards DDBMS −

- **Distributed Nature of Structural Units** – Now a days most organizations are partitioned into several units that are physically scattered over the world. Each unit needs its own set of local data. Thus, the total database of the organization converts into distributed.
- **Data sharing Need** −The several organizational divisions often need to interact with each other and share data and resources. This demands common databases or simulated databases that should be used in a co-ordinated manner.
- **Provision for OLTP and OLAP**–Online Analytical Processing (OLAP) and Online Transaction Processing (OLTP) works on diversified systems. Distributed database systems supports and both OLAP and OLTP.
- **Database Retrieval** − One of the common methods used in DDBMS is imitation of data across different locations. Replication of data spontaneously helps in data recovery if database in any site is broken. Users can access data from other sites while the damaged site is being rebuilt. Thus, database disaster may convert inconspicuous to users.
- **Usefulin Multiple Application Software** − Most organizations use a variant of application software and each is having different database support. DDBMS provides anidentical functionality for using the same data among diversified platforms.

1.2.3 Databases Types:

1.2.3.1. Homogeneous Database:

In a homogeneous database, all diverse sites collect data identically. At all the sites same operating system, database management system and the data structures used is being used. Therefore, they are easy to manage.

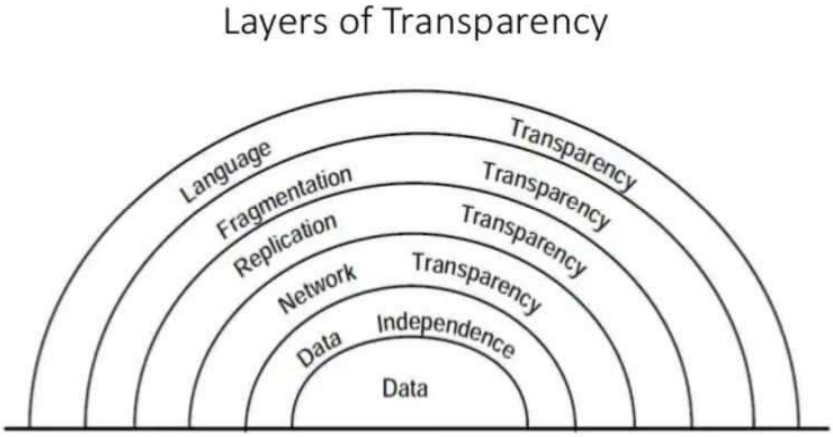
1.2.3.2 Heterogeneous Database:

In a heterogeneous distributed database, different sites can use dissimilar schema and software that can lead to glitches in transactions and query processing. Also, a particular site might be completely uninformed of the other sites. Diverse computers may use a different operating system, different database application. They possibly will even use changed data models for the database. Therefore, conversions are compulsory for different sites to interconnect.

1.3 DISTRIBUTED TRANSPARENT SYSTEM:

One of the property of Distributed Database Management System is Distributed transparent system. Because of this feature internal details of

the distribution is hidden from the users. DDBMS hides all the distributed complexities and allow users to feel that they are working on single and centralized database.



Different Layers of transparencies

1.3.1 Levels of Distributed Transparent System:

DDBMS is supporting transparency at three levels:

1.3.1.1 Fragmentation Transparency

In Fragmentation transparency, fragments are created to store the data in distributed wayand should stay transparent. In this all the data administration work necessarily control by the system, not by the user. In this job, when a user sets a query, the global query is distributed in many sites to get data from fragments and this data is place together at the end to produce the result.

1.3.1.2 Location Transparency

Location transparency confirms that the user can fire query on any relation or fragment of a relation like they are stored locally on user's place. But the table or its fragments are kept at isolated site in the distributed database system, should be completely unaware to the user. The address and access mechanism of the remote site are completely hidden.

In order to integrate location transparency, DDBMS must have access to restructured and perfect data dictionary and DDBMS directory which contains the details of locations of data.

1.3.1.3 Replication Transparency

Replication transparency certifies that duplication of databases are concealed from the users. It permits users to query upon a relation as if only a single copy of the table is in place.

Replication transparency is connected with concurrency transparency and failure transparency. At any time a user updates a data element, the update is replicated in all the replicas of the table. Though,

this process should not be identified to the user. This is known as concurrency transparency.

In case of let-down of a site, the user can still progress with his queries using replicated copies without any information of failure then this is failure transparency.

1.4 SUMMARY

Distributed Database Management System (DDBMS) software which manages number of databases raised at different locations and connected with each other through a computer network. It offers mechanisms so that the delivery remains unaware to the users, who see the database as a single database. Its internal details hidden from users with transparency feature.

1.5 LIST OF REFERENCES AND BIBLIOGRAPHY AND FURTHER READING

- Principles of Distributed Database Systems; 2nd Editied By M. Tamer Ozsu and Patrick Valduriez, Person Education Asia.
- Distributed Database; Principles & Systems By Publications, Stefano Ceri and Giuseppo Pelagatti,, McGraw-Hill International Editions (1984)
- https://cs.uwaterloo.ca/~tozsu/publications/distdb/distdb.pdf
- https://www.tutorialspoint.com/distributed_dbms/index.htm
- https://www.geeksforgeeks.org/distributed-database-system/

1.6 MODEL QUESTIONS:

1. Explain Distributed Database Management System? Where we can use it instead of DBMS?
2. Write and explain problem areas of distributed data base system.
3. Write advantages and disadvantages of DDBMS.
4. What is Distributed Transparent System? Explain its types.
5. Explain reasons for advancement of DDBMS.
6. Write a short note on:
    - Fragmentation Transparency
    - Location Transparency
    - Replication Transparency

2

DDBMS ARCHITECTURE

Unit Structure

2.0 OBJECTIVES

After going through this Chapter, you will be able to:

- understand Distributed database management system architecture
- define what is Global, Local, External, and Internal Schemas
- describe different architectural model for DDBM

2.1 INTRODUCTION

In any system architecture defines its structure. This means that the components of the system are identified, the purpose of each element is specified, and the interrelationships and interactions among these components are defined. The specification of the architecture of a system requires identification of the various units, with their connections and relationships, in terms of the data and control flow over the system.

2.2 DBMS STANDARDIZATION

Data standardization is the acute method of fetching data into a collective layout that allows for combined research, large-scale analytics, and sharing of refined tools and procedures

2.3 DDBMS ARCHITECTURE

Database systems comprise of complex data structures. Thus, to make the system efficient for retrieval of data and reduce the complexity of the users, developers use the method of Data Abstraction.

2.3.1.Factors for DDBMS Architecture:

DDBMS architectures are commonly developed dependent on three factors –

2.3.1.1. Distribution–Itstates the physical dispersal of data crosswise the different sites. Autonomy refers to the distribution of control, the distribution aspect of the classification deals with data. The user sees the data as one logical group. There are a numeralways DBMS have been distributed. We abstract these alternatives into two classes:

- client/server distribution
- peer-to-peer distribution (or full distribution).

2.3.1.2 Autonomy

Autonomy, in this perspective, refers to the distribution of mechanism, not of data. It identifies the distribution of regulator of the database system and the degree to which each component DBMS can work independently. Autonomy is a function of a quantity of factors such as whether the module systems interchange information, whether they can independently accomplish transactions, and whether one is certified to modify them. Requirements of an autonomous structure have been stated as follows:

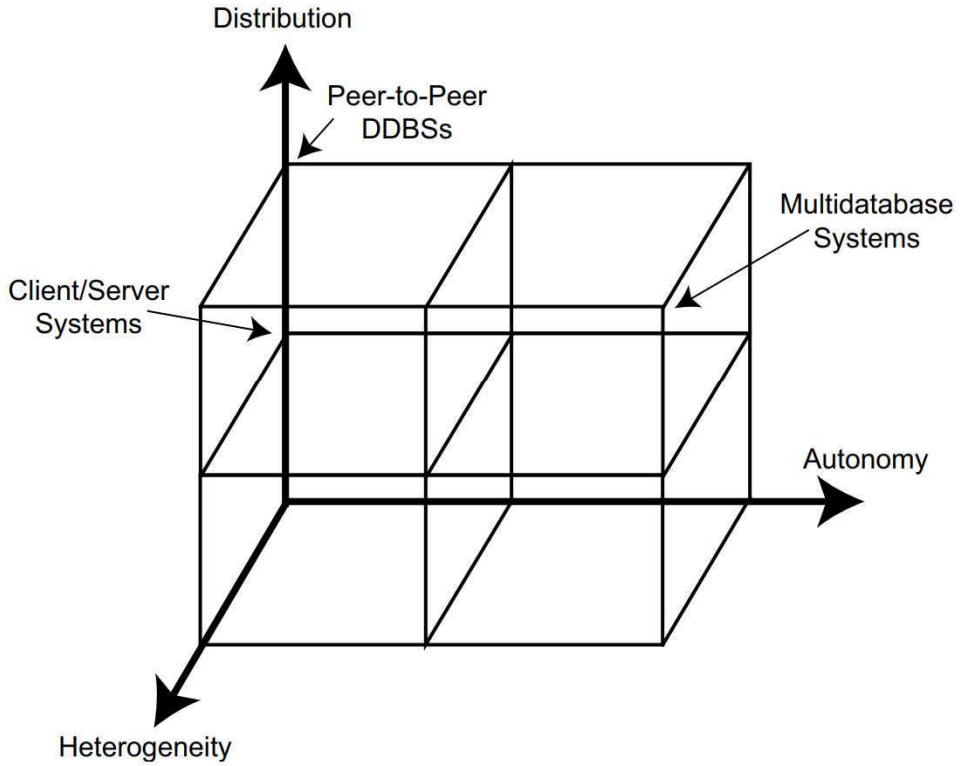
- The local procedures of the individual DBMSs are not affected by their involvement in the distributed system.
- The method in which the individual DBMSs develop queries and optimize them should not be affected by the accomplishment of global queries that access multiple databases.
- System regularity or operation should not be negotiated when individual DBMS join or leave the distributed system.

On the other hand, the proportions of autonomy can be stated as follows:

Design autonomy: Individual DBMS are permitted to use the data models and transaction management systems that they desire.

- **Communication autonomy:** To each of the discrete DBMS is free to make its own decision as to what type of information it wants to offer to the other DBMS or to the software that controls their global execution.
- **Execution autonomy:** Each DBMS can implement the transactions that are submitted to it in any way that it wants to.

2.3.1.3. Heterogeneity– It refers to the uniformity or variation of the data models, system tools and databases. Heterogeneity may happen in various forms in distributed systems, ranging from hardware heterogeneity and dissimilarities in networking protocols to distinctions in data managers. Representing data with different modelling tools creates heterogeneity because of the inherent expressive powers and limitations of individual data models. Heterogeneity in query languages not only involves the use of completely different data access paradigms in different data models (set-at-a-time access in relational systems versus record-at-a-time access in some object-oriented systems), but also covers differences in languages even when the individual systems use the same data model. Although SQL is now the standard relational query language, there are many different implementations and every vendor's language has a slightly different flavour.



DBMS Implementation Alternatives

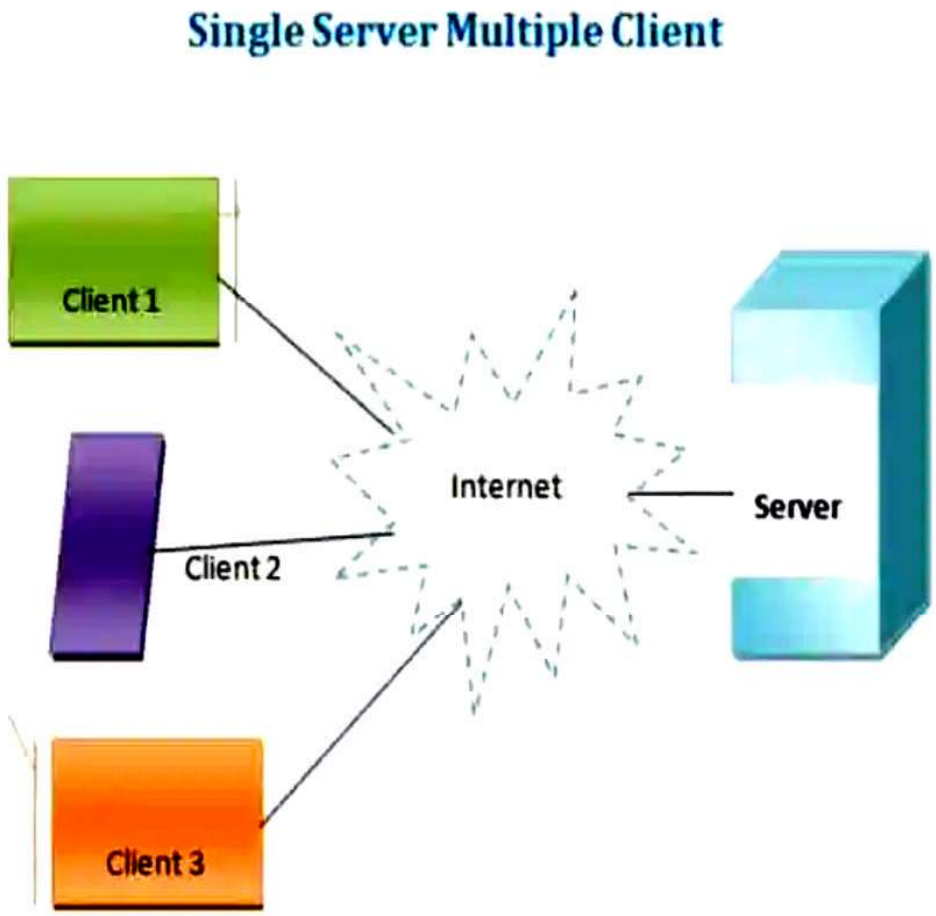
2.3 ARCHITECTURAL MODELS OF DISTRIBUTED DBMS:

2.4.1 Client-Server Architecture:

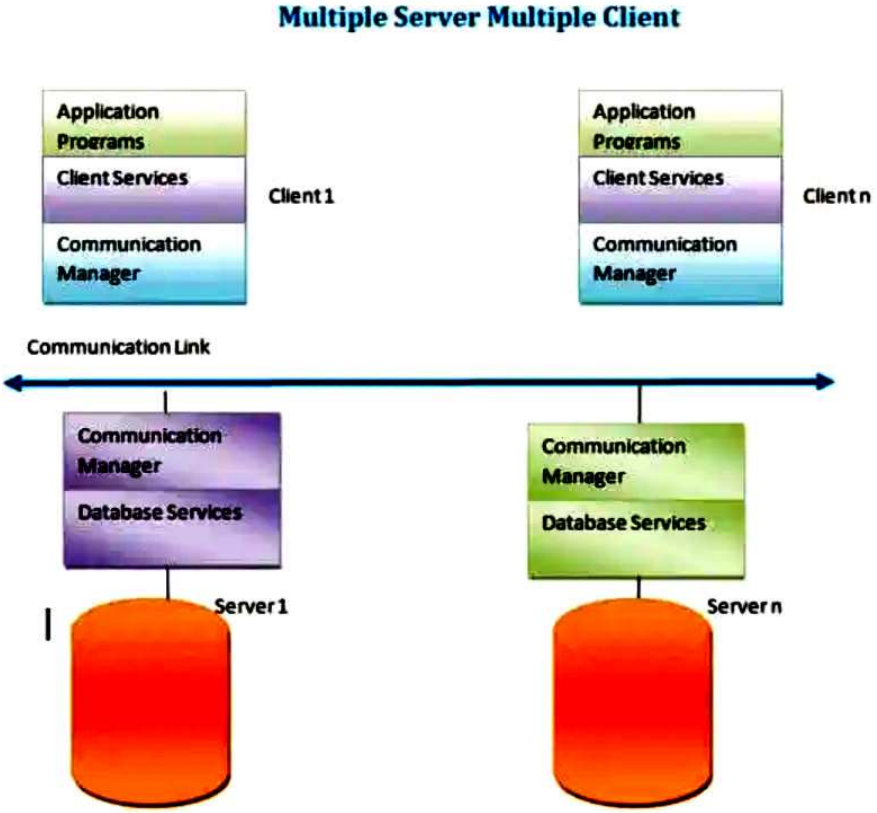
Client-Server architecture is a two-level architecture where the functionality is distributed into servers and clients. The server functions mainly comprise data management, query handling, transaction management and optimization. Client functions contain mainly user interface. Nevertheless, they have some functions resembling consistency checking and transaction management.

The two different types of clients – server architecture are as follows:

- **Single Server Multiple Client**



- **Multiple Server Multiple Client**:



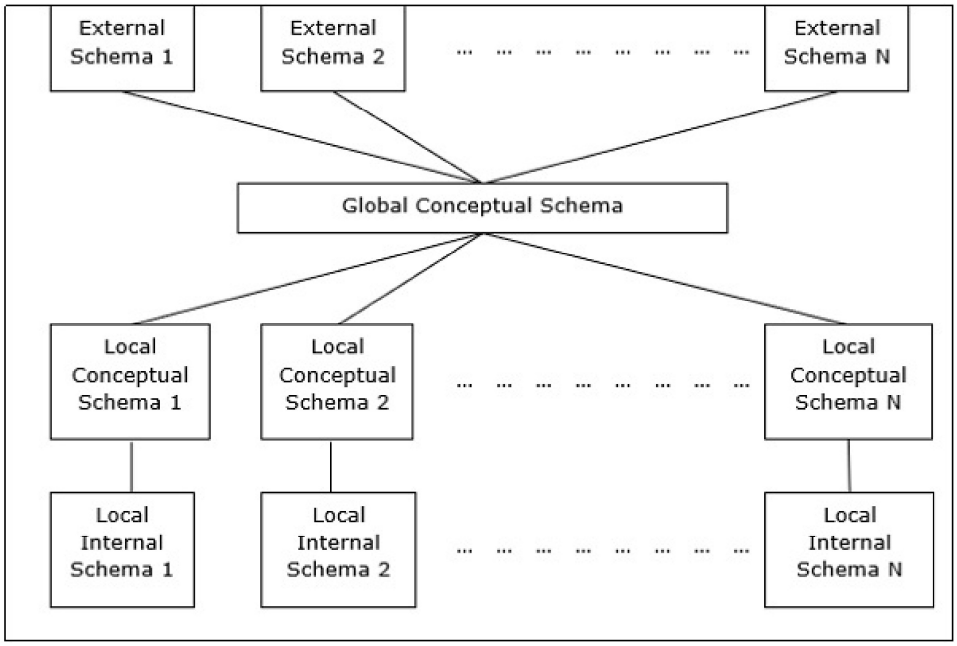
2.4.2 Peer- to-Peer Architecture for Distributed DBMS

In this systems, each peer actions both as a client and a server for instructing database services. The peers share their source with other peers and co-ordinate their actions.

This architecture in general has four levels of schemas −

2.4.2.1 Global, Local, External, and Internal Schemas:

- **Global Conceptual Schema** –Global Conceptual Schema represents the global logical view of data. It represents the logical explanation of entire database as if it is not circulated. This level encloses definitions of all units, relationships among entities and security and integrity facts of whole databases kept at all sites in a distributed system.
- **Local Conceptual Schema** –Local Conceptual Schema Show logical data organization at individual location.
- **Local Internal Schema** –Local Internal Schema represents physical record at each site.
- **External Schema** –External Schema Describes user's vision of facts and figures.



2.4.3 Multi - DBMS Architectures

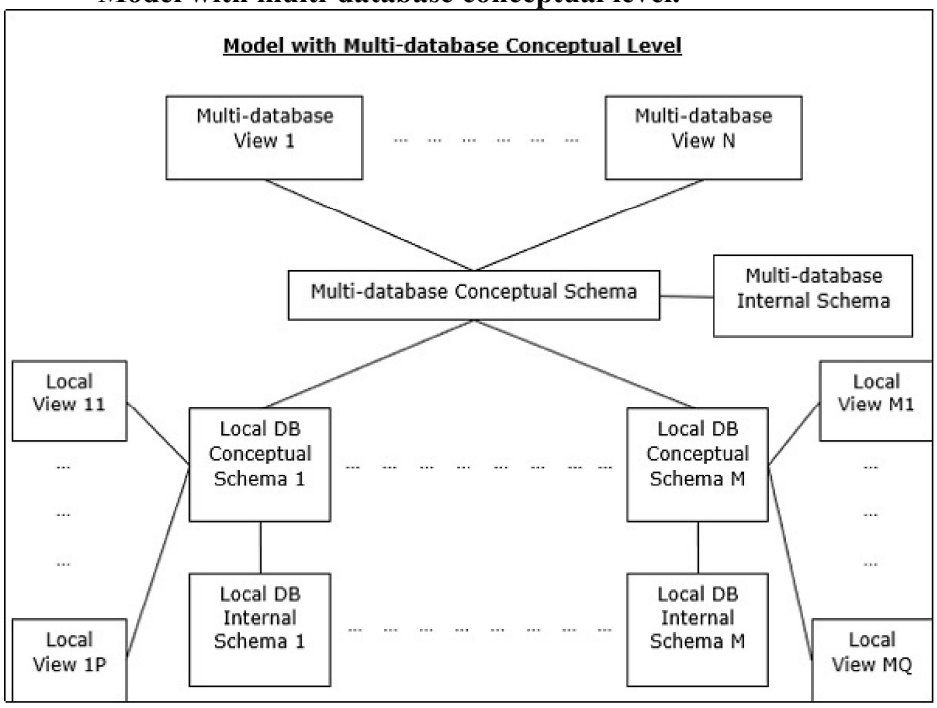
This is an integrated database system formed by a collection of two or more autonomous database systems.

Multi-DBMS can be expressed through six levels of schemas −

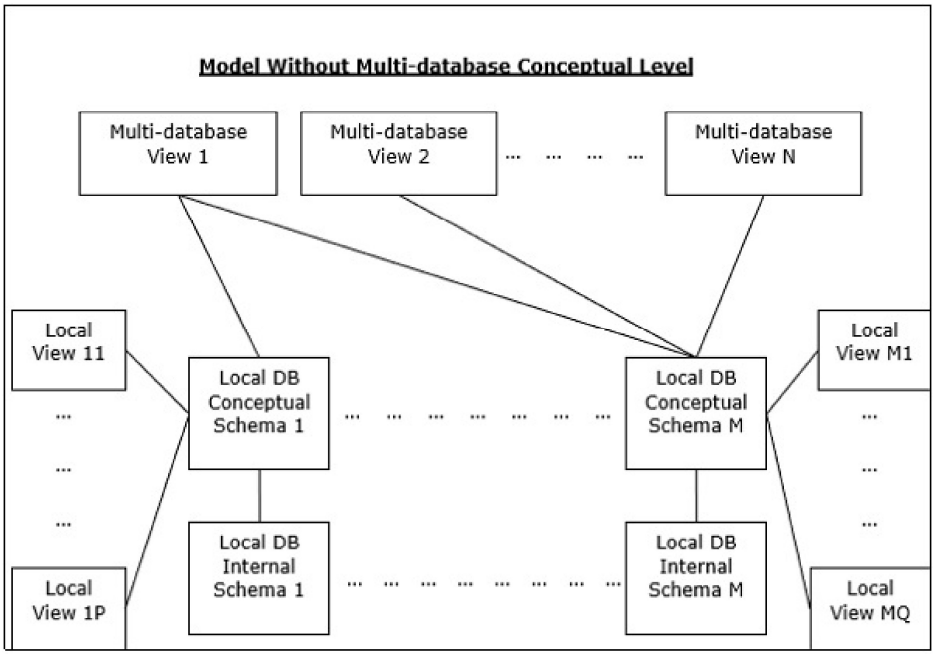
- **Multi-database View Level** − Describes multiple user views including of subsets of the integrated distributed database.
- **Multi-database Conceptual Level** − Shows integrated multi-database that comprises of global logical multi-database structure definitions.
- **Multi-database Internal Level** − Illustrates the data distribution across different sites and multi-database to local data mapping.
- **Local database View Level** − Give a picture of public view of local data.
- **Local database Conceptual Level** − Describes local data organization at each site.
- **Local database Internal Level** − Shows physical data organization at each site.

There are two design alternatives for multi-DBMS −

- **Model with multi-database conceptual level.**



- **Model without multi-database conceptual level.**



2.5 SUMMARY

There is different types of distributed databases. Distributed databases can be classified into homogeneous and heterogeneous databases having further divisions. Distributed architecture can be classified in various types namely client – server, peer – to – peer and multi – DBMS.

2.5 LIST OF REFERENCES AND BIBLIOGRAPHY AND FURTHER READING

- https://www.csitweb.com/distributed-dbms-features-needs-and-architecture/
- https://www.ohdsi.org/data-standardization/
- Principles of Distributed Database Systems; 2nd Editied By M. Tamer Ozsu and Patrick Valduriez, Person Education Asia.

2.7 MODEL QUESTIONS:

1. What is Distributed Database Management System Architecture? Explain
2. Explain different architectural model for DDBMS
3. Explain Peer- to-Peer Architecture for Distributed DBMS. Write Short Notes on the following:

- Global Schema
- Local Schema
- External Schema
- Internal Schemas

3

DISTRIBUTED DATABASE DESIGN

Unit Structure

3.0 OBJECTIVES

After going through this Chapter, you will be able to:

- understand Design of Distributed System
- Know Top-down and Bottom-up Strategies of Database Design
- describe Fragmentation and Allocation and replication of fragments
- gain knowledge about Query processing and Query Optimization

3.1 INTRODUCTION

The design of a distributed computer system contains making conclusions on the placement of data and programs through the sites of a computer network, as well as probably designing the network itself. In Distributed DBMS, the distribution of applications includes two things:

- Distribution of the distributed DBMS software
- Distribution of the application programs that run on it.

3.2 DESIGN PROBLEM OF DISTRIBUTED SYSTEMS

The distributed information system is defined as *"a number of interdependent computers linked by a network for sharing information among them"*. A distributed information system comprises of multiple independent computers that transfer or exchange information via a computer network.

- **Heterogeneity:**

Heterogeneity is functional to the network, computer hardware, operating system and execution of different developers. A crucial component of the heterogeneous distributed structure client-server environment is middleware. Middleware is a set of facilities that permits application and end-user to interrelate with each other across a heterogeneous distributed system.

- **Openness:**

The openness of the distributed system is determined mainly by the point to which new resource-sharing facilities can be made offered to the users. Open systems are considered by the fact that their key interfaces are circulated. It is based on a uniform communication tool and published interface for access to pooled resources. It can be built from varied hardware and software.

- **Scalability:**

Scalability of the system should persist efficient even with a important increase in the number of operators and resources coupled.

- **Security:**

Security of information system has three mechanisms confidentially, integrity and availability. Encryption defends shared resources, preserves delicate information secrets when communicated.

- **Failure Handling:**

When some errorsarise in hardware and the software suite, it may produce incorrect results or they may stop before they have completed the predicted computation so corrective techniques should to implement to handle this case. Failure control is challenging in distributed systems because the let-down is incomplete i.e. some components fail while others come to an end.

- **Concurrency:**

There is a chance that several users will attempt to access a common resource at the similar time. Multiple users create requests for the same resources, i.e. read, write, and update. Each source must be safe in a parallel environment. Any item that signifies a shared resource a distributed system must confirm that it operates properly in a concurrent setting.

- **Transparency:**

Transparency confirms that the distributed system should be observed as a single object by the users or the application programmers somewhat than the pool of autonomous systems, which is work together. The user should be uninformed of where the services are situated and the transmitting from a local machine to anisolated one should be transparent.

3.3DESIGN, STRATEGIES (TOP-DOWN, BOTTOM-UP)

It has been recommended that the group of distributed systems can be scrutinized along three scopes
1. Level of Sharing
2. Behaviour of access forms
3. Level of information on access pattern behaviour

To follow all extents some proper method has to be there to grow distributed database design. There are two methods for developing any database, the top-down method and the bottom-up method. Although these approaches appear completely different, they share the mutual goal of employing a system by relating all of the communication between the processes.

3.3.1 Top-down design Strategy

The top-down design structure starts from the common and transfers to the specific. In other words, you start with a universal idea of what is required for the system and then work your method down to the more specific particulars of how the system will work together. This process contains the identification of diverse entity types and the definition of each entity's characteristics.

3.3.2 Bottom – up design Strategy

The bottom-up approach begins with the specific details and moves up to the general. This is complete by first recognizing the data elements and then alliance them collected in data sets. In other words, this technique first identifies the aspects, and then groups them to form objects.

3.4 FRAGMENTATION

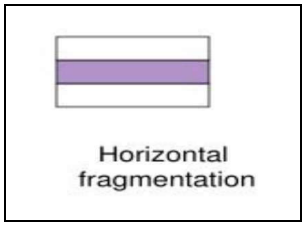
Data fragmentation is a procedure used to break up entities. The item might be a user's database, a system database, or a table. It permits you to breakdown a single object into two or more sectors, or fragments. Each fragment can be put in storage at any site over a computer network. In designing a scattered database, you must decide which portion of the database is to be put in storage where. One method used to break up the database into logical entities called fragments. Facts about data fragmentation is kept in the distributed data catalog(DDC), from which it is retrieved by the TP to process user requests. Fragmentation information is deposited in a distributed data catalogue which the dealing out computer uses to process a user's demand.

3.4.1 Data Fragmentation Strategies:

Data fragmentation strategies, are established at the table level and comprise of dividing a table into logical fragments. There are three forms of data fragmentation strategies: horizontal, vertical, and mixed.

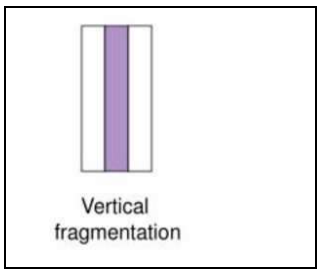
3.4.1.1 Horizontal fragmentation

This kind of fragmentation refers partition of a relation into fragments of rows. Each fragment is kept at a different workstation or node, and each fragment comprises unique rows. Each horizontal fragment may have a changed number of rows, but each fragment must have the identical attributes.



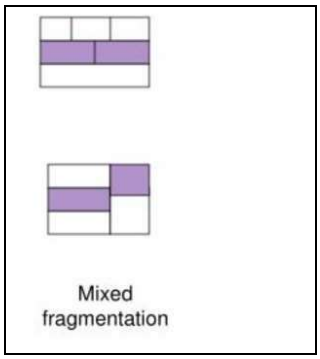
3.4.1.2 Vertical fragmentation

This type of fragmentation refers to the partition of a relation into fragments that contain a collection of attributes. Each vertical fragment must have the same number of rows, but can have dissimilar attributes depending on the key.



3.4.1.3 Mixed fragmentation

This type of fragmentation is a two-step procedure. First, horizontal fragmentation is completed to obtain the essential rows, then vertical fragmentation is done to distribute the attributes between the rows.



3.5 ALLOCATION AND REPLICATION OF FRAGMENTS

3.5.1 Data Allocation

Data allocation is a procedure of deciding where to accumulate the data. It also comprises a decision as to which data is stored at what location. Data provision can be centralised, partitioned or replicated.

3.5.1.1. Centralised

The entire database is stored at one place. No distribution happens.

3.5.1.2 Partitioned

The database is distributed into several fragments that are deposited at numerous sites.

3.5.1.3 Replicated

Copies of one or added database fragments are kept at several sites.

3.5.2 Data Replication

Data replication is the storage of data replicas at numerous sites on the network. Fragment copies can be stored at several site, thus increasing data availability and reply time. Replicated data is subject to a common consistency rule. This rule involves that all replicas of the data fragments must be same and to ensure data consistency among all of the imitations.

Although data replication is favourable in terms of availability and response periods, the maintenance of the replications can turn into complex. For example, if data is simulated over multiple sites, the DDBMS needs to decide which copy to access. For a query process, the nearest copy is all that is necessary to satisfy a transaction. Though, if the operation is an update, at that time all copies must be selected and restructured to satisfy the common consistency rule.

A database can be moreover fully replicated, partially replicated or not replicated.

3.5.2.1 Full replication

Stores multiple copies of each database fragment at various sites. Fully replicated databases can be unlikely because of the amount of overhead forced on the system.

3.5.2.2 Partial replication

Stores multiple copies of some database fragments at multiple sites. Most DDBMS can hold this type of replication precise well.

3.5.2.3 No replication

Stores each database section at a single site. No repetition arises.

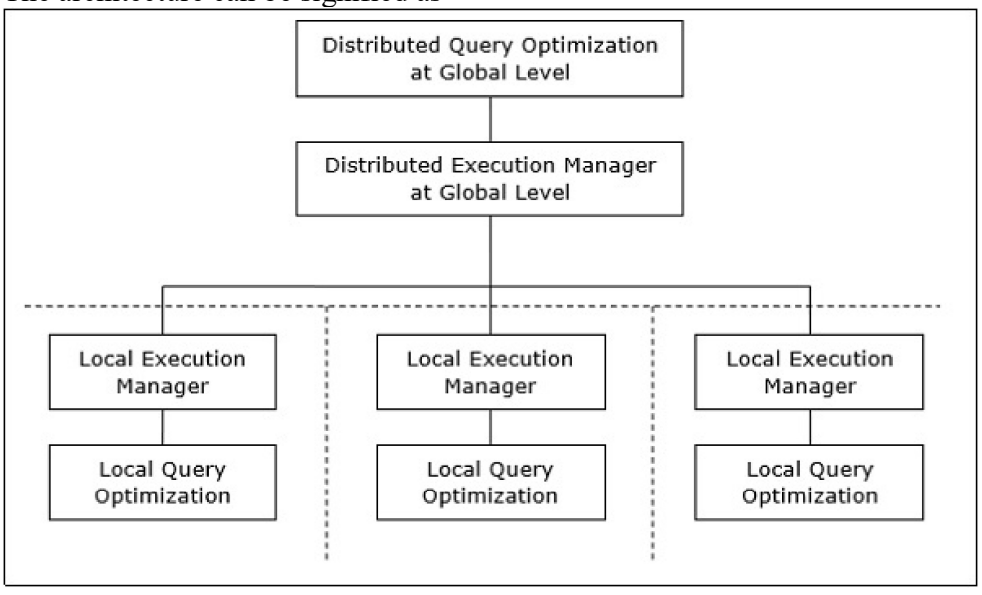
Data replication is mainly useful if usage frequency of remote data is great and the database is fairly huge. Another advantage of data replication is the opportunity of restoring lost data at a specific site.

3.6 QUERY PROCESSING OVERVIEW

A Query processing in a distributed database management system needs the transmission of data among the computers in a network. A distribution approach for a query is the ordering of data diffusions and local data processing in a database system. Usually, a query in Distributed DBMS entails data from multiple sites, and this need for data from different sites is termed the transmission of data that causes communication costs. Query processing in DBMS is unlike from query processing in centralized DBMS due to this communication cost of data transmission over the network. The transmission cost is small when sites are joined through high-speed Networks and is pretty significant in other networks.

In a distributed database system, handling a query comprises of optimization at both the world-wide and the local level. The query move in the database system at the client or supervisory site. Here, the user is legalised, the query is checked, translated, and enhanced at a global level.

The architecture can be signified as −



Mapping Global Queries into Local Queries

The procedure of mapping global queries to local ones can be recognised as follows −

- The tables essential in a global query have fragments distributed crosswise multiple sites. The local databases have data only about limited data. The supervisory site uses the global data dictionary to collect information about the distribution and recreates the global vision from the fragments.

- If there is no duplication, the global optimizer tracks local queries at the sites where the fragments are kept. If there is replication, the global optimizer selects the site based upon communication cost, workload, and server speed.

- The global optimizer produces a distributed execution proposal so that least amount of data allocation occurs across the sites. The plan shapes the location of the fragments, order in which query steps wishes to be executed and the processes involved in transferring transitional results.

- The local queries are optimized by the local database servers. Finally, the local query effects are merged together through blending operation in case of horizontal fragments and join process for vertical fragments.

3.7QUERY OPTIMIZATION

Distributed query optimization needs evaluation of anenormous number of query trees each of which produce the necessary results of a query. This is primarily due to the occurrence of large volume of replicated and fragmented data. Hence, the goal is to find an optimal solution instead of the finest solution.

The main concerns for distributed query optimization are −
- Optimal consumption of resources in the distributed system.
- Query trading.
- Decrease of solution space of the query.

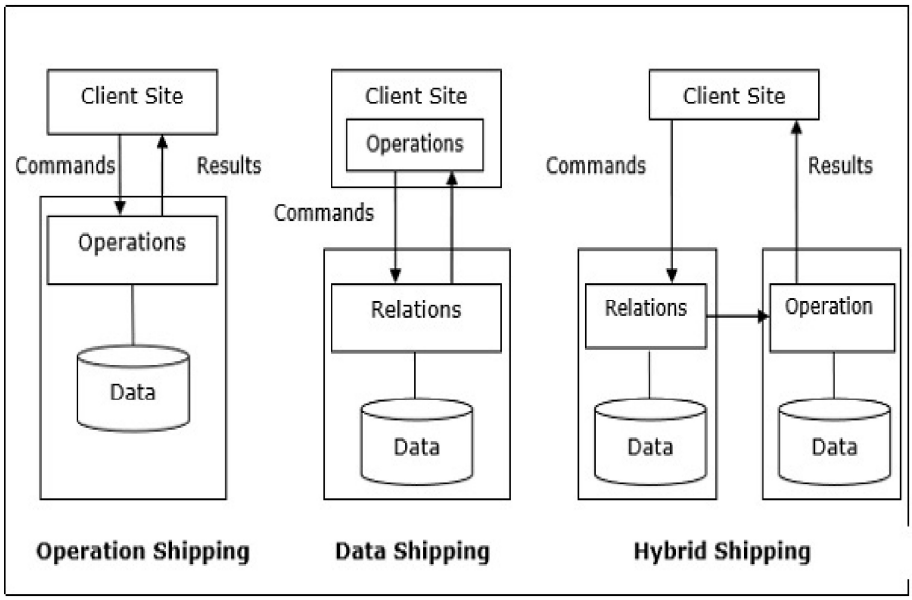
3.7.1 Optimal Utilization of Resources in the Distributed System

A distributed system has a number of database servers in the various sites to perform the actionsbelong to a query. Following are the approaches for optimal resource utilization −

- **Operation Shipping** − In operation shipping, the process is run at the location where the data is kept and not at the client site. The results are then transported to the client site. This is applicable for operations where the operands are presented at the same site. i.e. Select and Project operations.

- **Data Shipping** − In data shipping, the facts fragments are transported to the database server, where the processes are executed. This is used in procedures where the operands are distributed at diverse sites. This is also suitable in systems where the communication overheads are low, and local processors are abundant slower than the client server.

- **Hybrid Shipping** − This is a mixture of data and operation shipping. At this point, data fragments are transmitted to the high-speed processors, where the process runs. The results are then lead to the client site.



3.7.2 Query Trading

In query trading system for distributed database systems, the controlling/client site for a dispersed query is called the buyer and the locations where the local queries execute are entitled sellers. The buyer expresses a number of options for choosing sellers and for restructuring the global results. The goal of the buyer is to reach the optimal cost.

The algorithm jumps with the buyer allocating sub-queries to the vender sites. The best plan is created from local improved query plans proposed by the sellers joined with the communication cost for renovating the final result. Once the global optimum plan is framed, the query is performed.

3.7.3 Reduction of Solution Space of the Query

Optimal solution normally involves reduction of clarification space so that the cost of query and data relocation is reduced. This can be attained through a set of experimental rules, just as heuristics in centralized structures.

Some of the rules are as follows:

- Implement selection and projection tasks as early as promising. This eases the data flow over communication web.
- Streamline operations on horizontal fragments by removing selection conditions which are not applicable to a particular site.
- In case of join and union procedures comprising of fragments sited in multiple sites, transfer fragmented data to the site where utmost of the data is present and implement operation there.
- Use semi-join process to qualify tuples that are to be combined. This decreases the amount of data relocation which in turn reduces communication cost.
- Combine the common leaves and sub-trees in a dispersed query tree.

3.5 SUMMARY

The improvement in technology has opened the locks for unlimited volumes of data to transfer into the system. Distributed database technology is certainly one of the key growths in the field of database systems. Though, with the remarkable amount of data driving in from various sources and in many formats, it may become relatively a difficult task for a business to stock, process and manage this data. Choosing the services of a database expansion company that provides tradition database development solutions provider may support to meet the specific experiments of the business by keeping data well-organized, protected and easily accessible for approved users.

3.6 LIST OF REFERENCES AND BIBLIOGRAPHY AND FURTHER READING

- https://www.dlsweb.rmit.edu.au/Toolbox/knowmang/content/distributed_sys/ddms_design.htm
- http://www.myreadingroom.co.in/notes-and-studymaterial/65-dbms/559-database-design-concepts.html
- https://www.geeksforgeeks.org/design-issues-of-distributed-system/

3.7 MODEL QUESTIONS

1. Explain Design problem of distributed systems.
2. What is Query Optimization? Explain Types.
3. Explain Query Processing. Differentiate Global Queries into Local Queries.
4. Explain Data Fragmentation Procedure.
5. Explain Design Problem of Distributed System.