① DISTRIBUTED DATABASE — M. Tamer Ozsu, Patrick ①

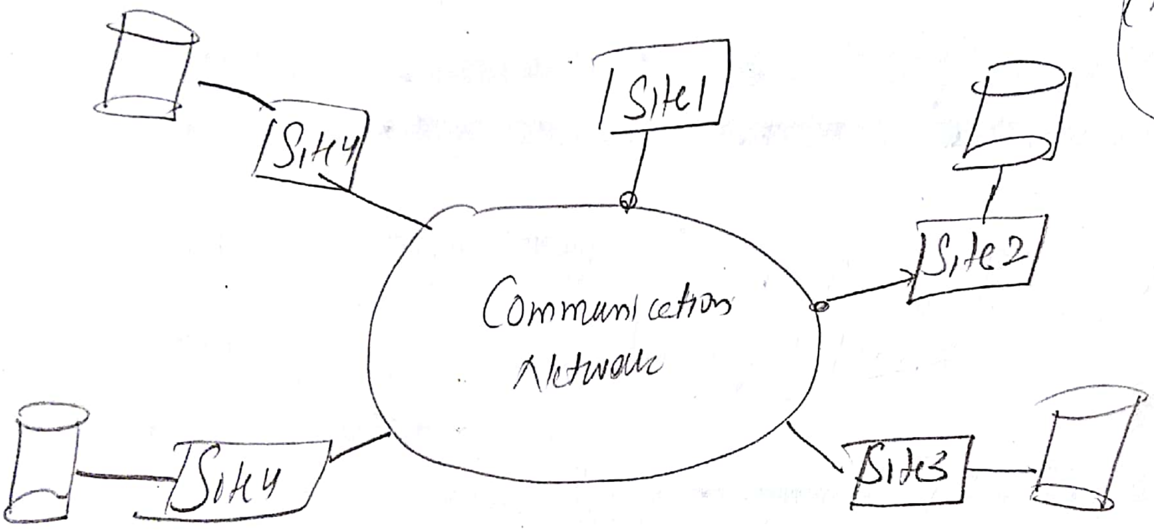
Distributed Database is a collection of multiple logically interrelated databases, distributed over a Computer Network

relationships (Foldr/ng)

2 DISTRIBUTED DATABASE MANAGEMENT SYSTEM.

DDBMS is a software System that manages a distributed database ~~while~~ and ~~making~~ makes the distribution transparent to the user.

(implementation details, communication network, kahde, file location te data store, only o/p dikhane)



DDBS ENVIRONMENT

3 Conditions for Distributed Database

For a database to be called distributed the following minimum conditions should be satisfied.

1. Connection of Database nodes over a Computer Network.

- There are multiple computers known as sites or nodes. These sites must be connected by a communication network to transmit data

• Promises / Features of DDBMS

(1) Transparency (2) Autonomy (3) Reliability & Availability.

a) Transparency

It refers to the idea of hiding implementation details from end users. The advantage of fully transparent DBMS is the high level of support that it provides for the development of complex applications.

Types of Transparency

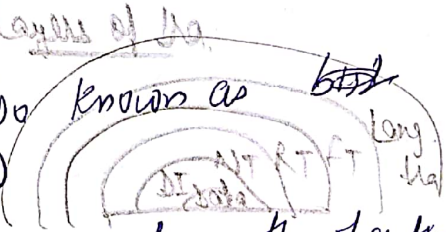
(1) Transparent mngmt of dis. ed & rep. ed data
eg. Emp, Proj
→ Data Indepen.
→ n/w trans. → location
(distri trans.) → naming
↓ unique name for each object (locality of refer)

(1) Data Organisation Transparency.
(2) Replication Transparency
(3) Fragmentation Transparency.
(4)

(2) Reliability thru Dis. ed trans.
(3) Improved perf → each site handles own data
(4) easier eg. expansion → Localization

→ replin trans.
→ frag. trans → how rel

Layers of DS



(1) Data Organisation Transparency: Also known as distribution or n/w transparency. This refers to freedom for the user from the operational details of the network & placement of the data in the distributed system. It is further divided into location & naming transparency.

(1.1) Location Transparency

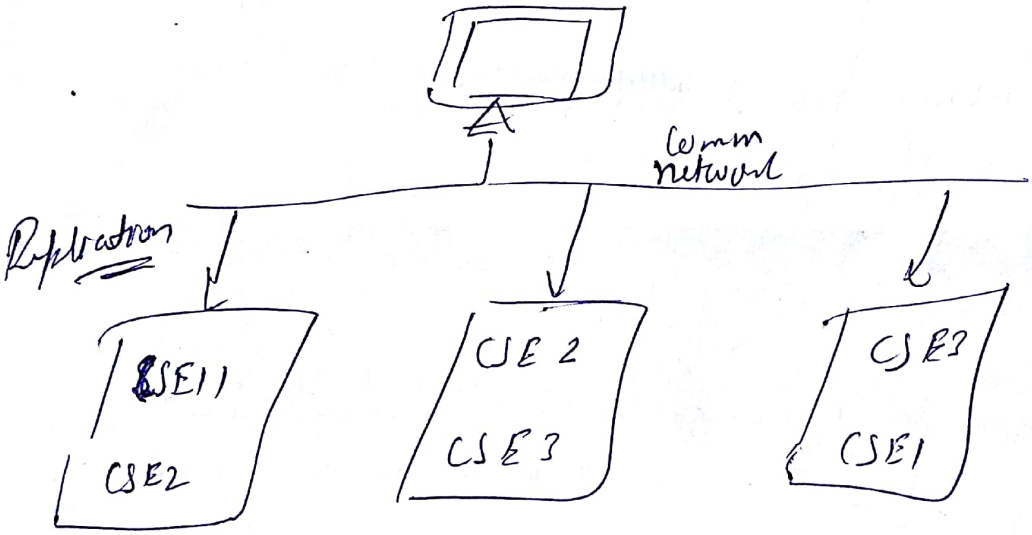
It refers to the fact that the command used to perform a task is independent of the location of data & the location of the nodes & the system on which an operation is carried out.

1.2] Naming Transparency:

It means that a unique name is provided for each object in the database. In the absence of naming transparency, users are required to embed the location name as part of the object.
is embedded as a part of object name

2] Replication Transparency:

The copies of the same data objects may be stored at multiple sites for better availability, performance and reliability. Replication transparency makes the user unaware of the existing existance of these copies.



3] Fragmentation Transparency:

Two types of fragmentation are possible. Horizontal & Vertical

Horizontal Fragmentation distributes a relation (table) into sub relations that are subsets of the tuples (rows) in the original relations.

Vertical Fragmentation:
distributes a relation into subrelations where each subrelation is defined by a subset of the columns of the original relation.

Fragmentation transparency makes the user ~~unknown~~ unaware of the existance of fragments.

4 Design Transparency & Execution Transparency:
Users have no idea how the distributed database is designed and how a transaction executes.

5 Autonomy
It determines the extent to which individual node or database in a ~~connected~~ DDB can operate independently.
A high degree of autonomy is desirable for increased flexibility customised maintainance of an individual node
Autonomy can be applied to design, communication & execution.

1) Design Autonomy: Refers to independence of data model usage & transaction management techniques among nodes
Pure, Network, Tree, Hierarchical, Object oriented

2) Communication Autonomy: It determines the extent to which each node can decide on sharing of information with other nodes.

3) Execution Autonomy: It refers to the independence of user to act as they please.

3) Availability & Reliability
(Trustworthy)

Reliability is defined as the probability that a system is running (not down) at a certain time point.
(performing) (means ki chez theek kam kar rahi hai.

Availability is a probability that the system is continously available during a time interval.
(Means sab kuch hai yah koi chez hai.

Reliability and Availability are related to database failures, faults & errors.
(Error ni h jo system available hai oh reliable v ho.)

Advantages of Distributed Database

(1) **Improved ease & flexibility of Applications development**

Developing and maintaining applications at geographically distributed sites of an organisation is facilated owing to transparency of data distribution & control.

(Data different sites pe distributed hone ke bad isse kaise transparency maintain hoti hai).

(2) **Increased Reliability & availability**

This is achieved by the isolation of faults to their site of origin without affecting the other databases connected to the network.

When the data & DDBMS software are distributed over several sites, one site may fail while other sites continue to operate.

Only the data and software that exist at the failed site cannot be accessed. This improves both reliability & availability.

Replication → User can recover all the data of failed site from other sites because, In distributed databases data is replicated on multiple nodes.

- In Central ×
- In Parallel ×

(3) **Improved Performance**

A distributed DBMS fragments the database by keeping the data closer to where it is needed most. (Data localization)

reduces the contention for CPU & I/O services.

Data locality

Easier Expansion :- (Scalability)

In distributed environment expansion of the system in term of adding more data increasing database sizes, or adding more processors is much easier. It is The most important feature of distributed database system. which is also known as scalability.

Functions of Distributed Database

1) Keeping track of data distribution

The ability to keep track of the data distribution, fragmentation, & replication by expanding the DDBMS catalog.

metadata file Information.

2) Distributed Query Processing :-

The ability to access remote sites and transmit queries & data among the various sites via a communication network

plan

3) Distributed Transaction Management :- The ability to devise execution strategies for queries and transactions that access data from more than one site and to synchronize the access to distributed data and maintain the integrity of the overall database.

transaction

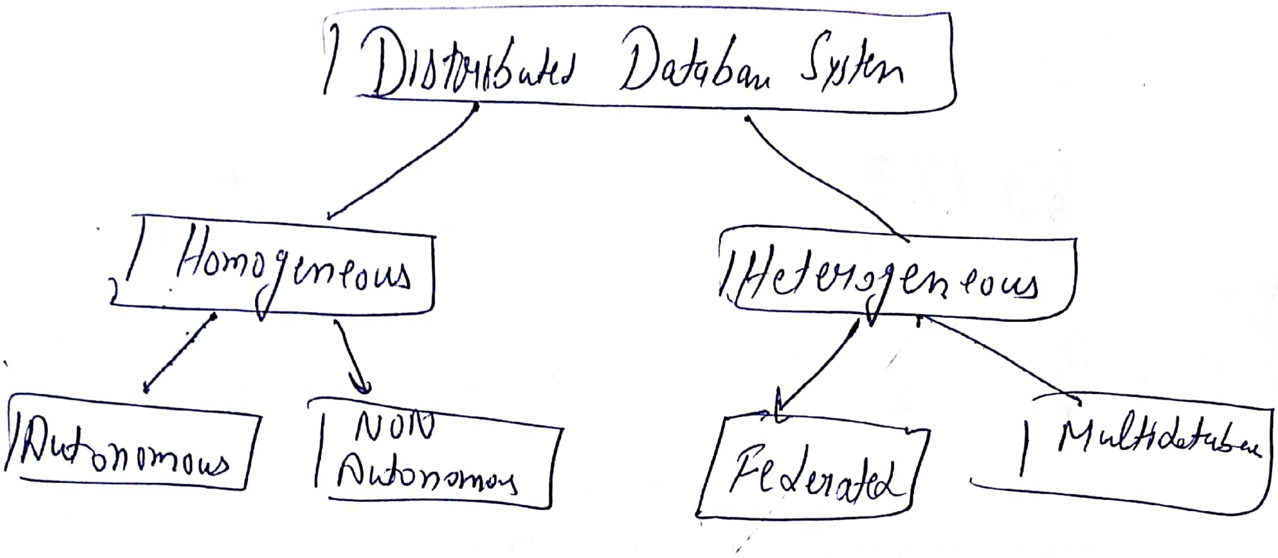
4 Replicated Data management :
The ability to decide which copy of a replicated data item to access and to maintain the consistency of copies of a replicated data item.
↳ [means har site te same copy honi chahiti hai]

5 Distributed Database Recovery :
The ability to recover from individual site crashes and from various types of failures.

6 Security :
Distributed transaction must be executed with the proper management of the security of the data and the authorization/access privileges of users.

7 Distributed directory (Catalog) mangmt :
A directory contain information (metadata) about the data in the database. The directory may be global for the entire DDB or local for each site.

Types of Distributed Database System



(1) Homogeneous Distributed System:

i) All the servers and client use identical software and operating system.

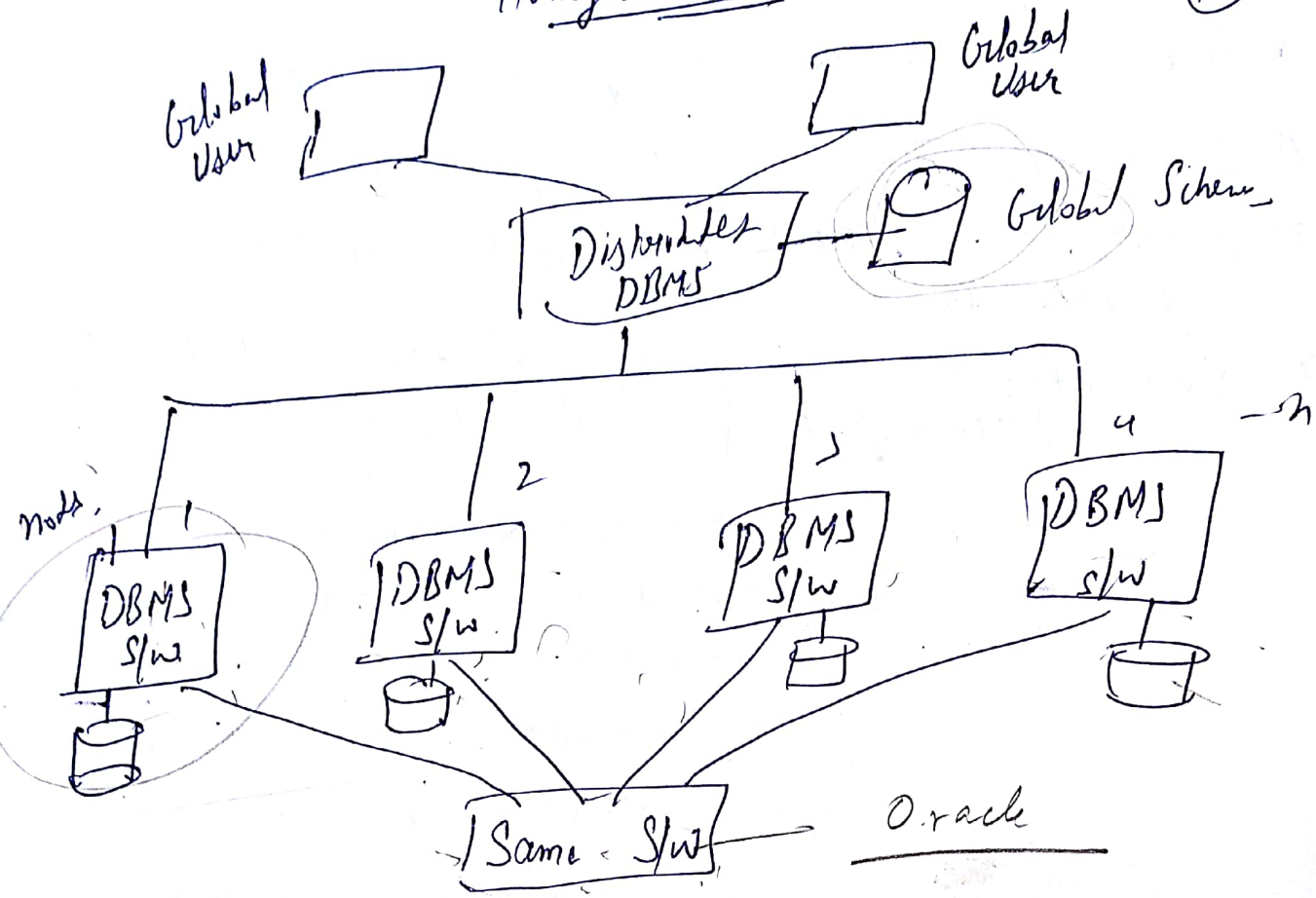
ii) Each site is aware of all other sites & cooperates with other sites to process user requests.

iii) The database is accessed through a single interface as if it is a single database
   i.e it appears to user as a single system.

iv) It is easy to design & manage.

v) Difficult for most organisations to force a homogeneous environment
[Har company different configuration ya OS use karti hai so ase bandation ni lga sakte k ik hi use krna] so ek difficult hai apply krna har company ko,

Homogenous Database



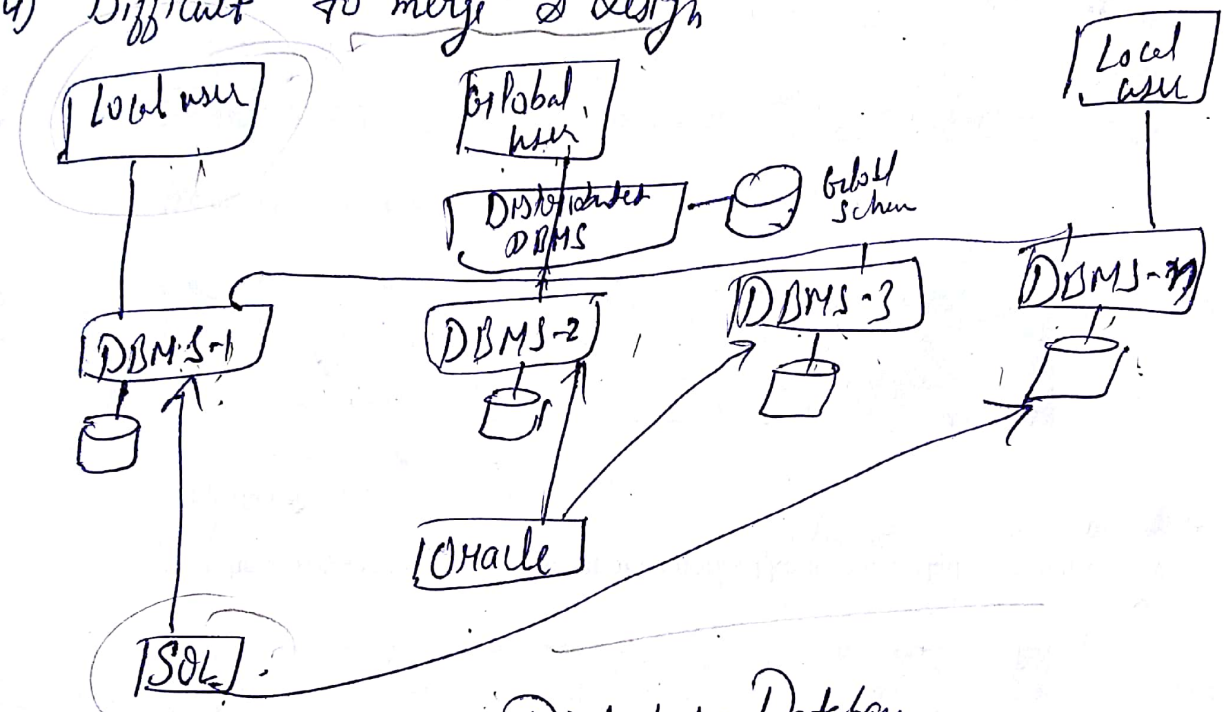
Types of Homogenous & distributed database

1. Autonomous : (All sites are identical)
   Each database is independent that functions on its own. They are integrated by a controlling application & ~~use~~ use message passing to share data updates.

2. Non-Autonomous : Data is distributed across the homogenous nodes and a master DBMS co-ordinates. Data updates across the sites.

Heterogeneous Distributed Database

1) Different sites have different operating systems, data models and dbms s/w.
2) Query processing and transaction processing is complex due to dissimilar ~~site~~ s/w.
3) A site may not be aware of other sites & so there is limited co-operation in processing user requests
4) Difficult to merge & design



Types of Heterogeneous Distributed Database

1) Federated:
The Heterogeneous database systems are independent in nature & integrated together so that they function as a single database system. ~~Global schema or view is shared by the applications.~~

2) Multidatabase System:
There is no global schema, but it constructs ~~one~~ may be present as needed by the applications.

~~Relational~~ Relational

Parallel DBMS	Distributed DBMS
1) Machines are physically close to each other, Ex same server room.	1) Machines can be far from each other ex:- in different continent.
2) Machines are connected with high-speed LANs & switches.	2) Machines can be connected using public network ex:- Internet.
3) Communication cost is small	3) Communication cost is high.
4) Follow shared memory, shared-disk or shared nothing architecture	4) Follows shares nothing architecture.
5) It is a software system where multiple processors are used to execute & run queries in parallel.	5) It is a software system that manages multiple logically interrelated databases distributed over a communication network.
~~6) Execution speed is fast~~	
6) Nodes should be homogenous.	6) Nodes may be homogenous or heterogeneous
7) Backup at on site only.	7) Backup at multiple sites.
8) Diagram shared Disk, Shared memory Shared nothing.	8) Diagram : distributed Database
9) Replication feature not exist. (when a node fails)	9) Replication feature exists. with the help of Replication factor we can recover data from any node.

Architectural Models of DDBMS

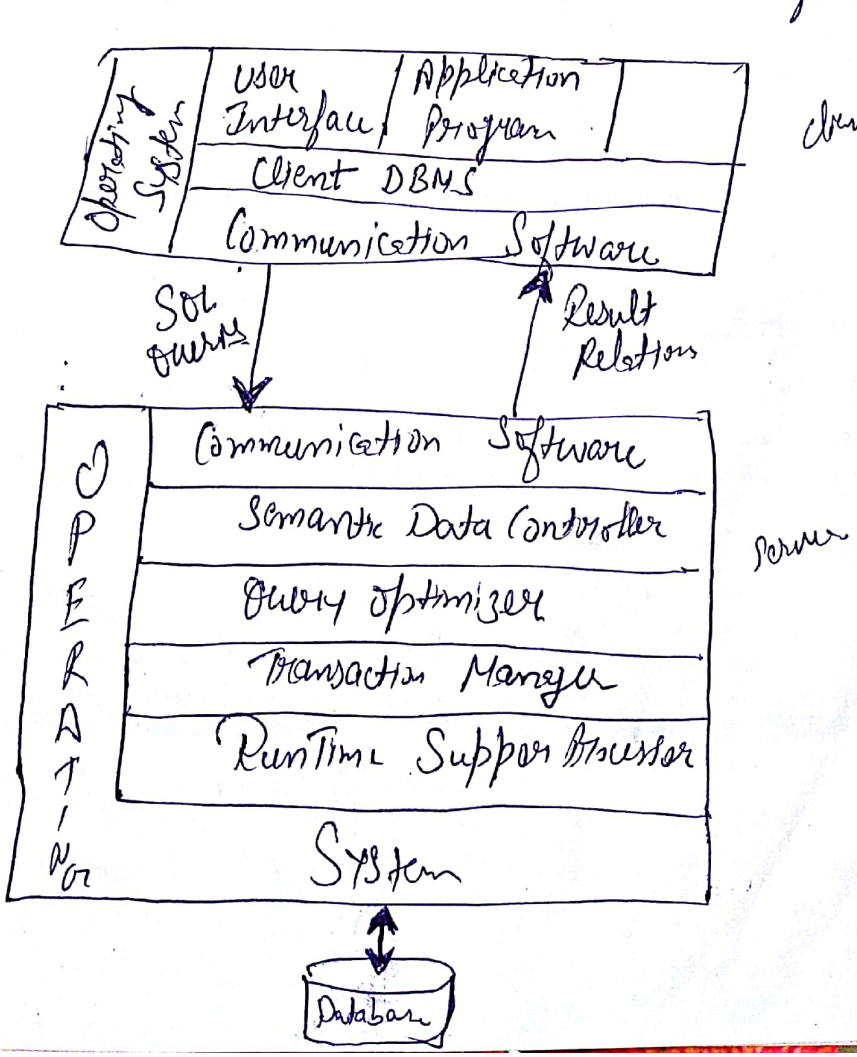
1) Client-Server Architecture.
2) Peer to Peer Architecture.
3) Multi DBMS Architecture.

1 Client-Server Architecture

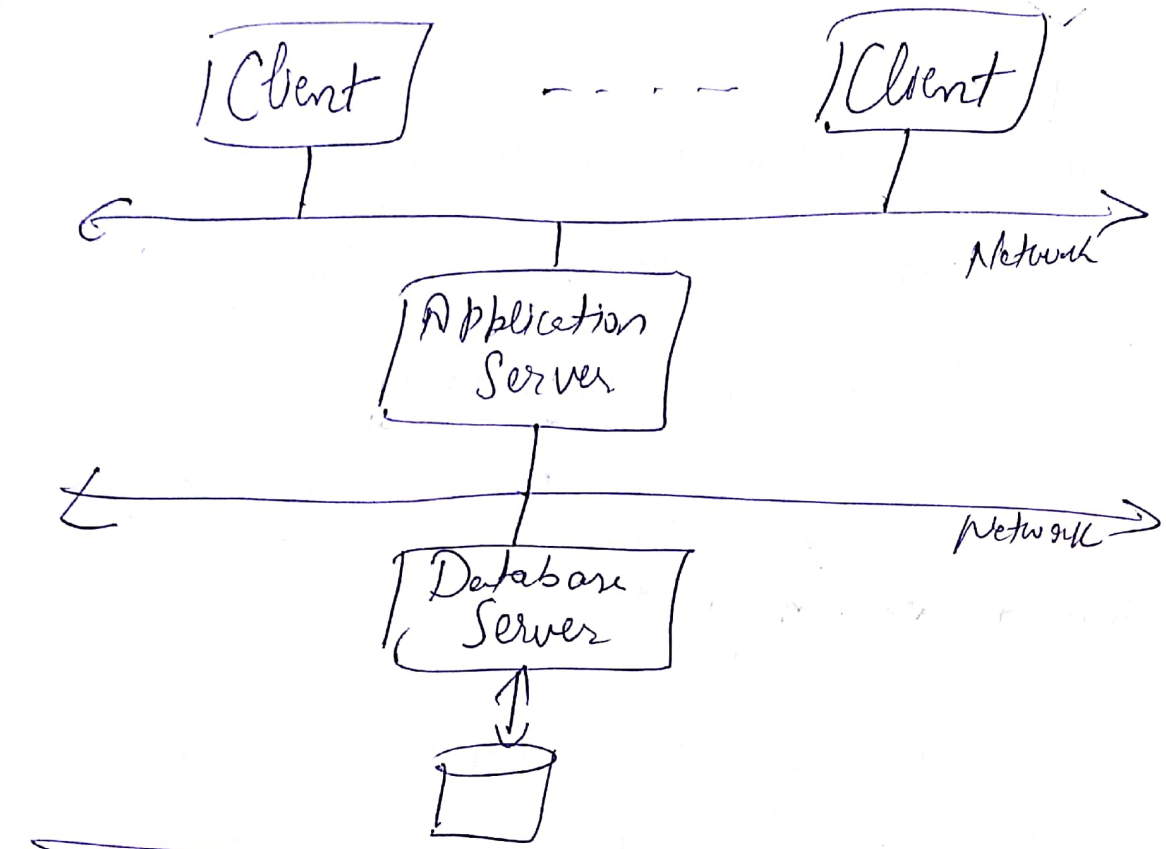
It is a 2 level architecture. Where functionality is divided into servers & clients.

- Server does most of the data management work. This means that all the query processing & optimization, transaction management & storage management is done at Server ~~Front~~ end.
- Client function include applications user interface

Client/Server Architecture →



① Single Server / Multiple Clients



② Multiple Server / Multiple Clients

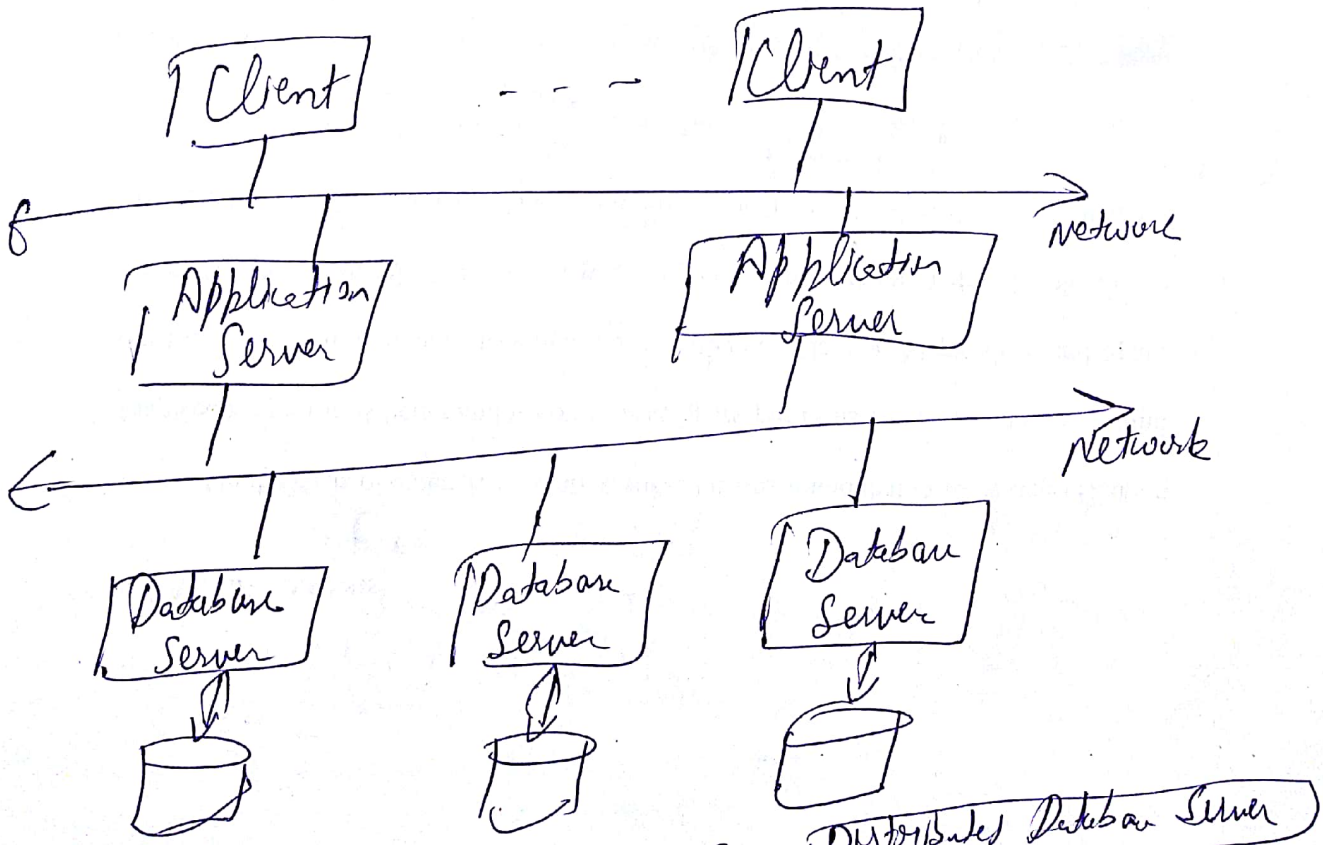


Figure: Distributed Database Server

⑧ Peer-to-Peer System

Every system acts a full DBMS & can communicate with others systems both as client & server to execute queries & transactions.

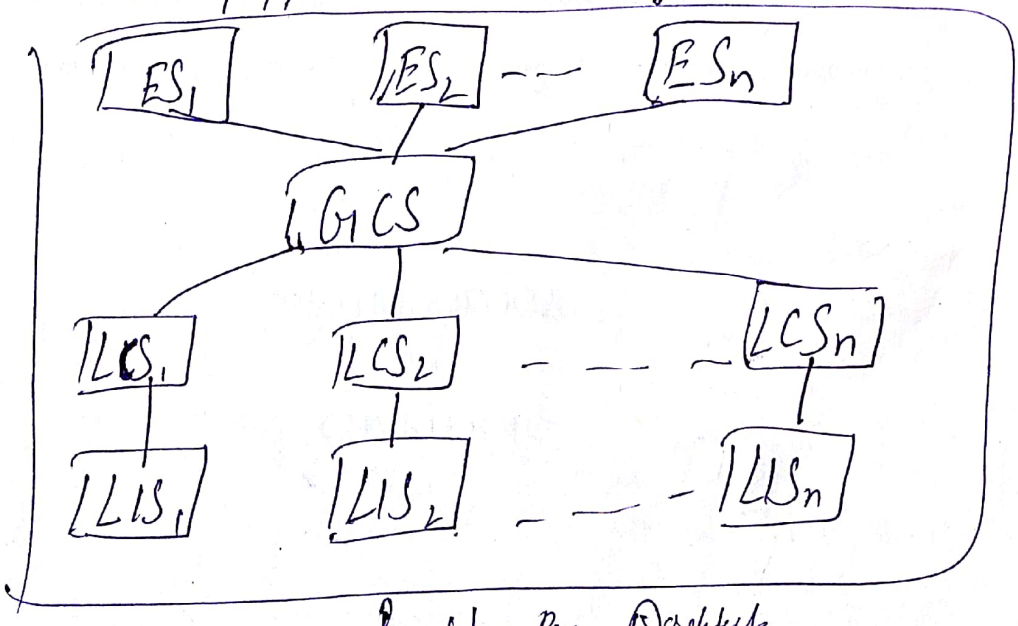
The peer-to-peer architecture has 4 levels of schema :-

1) Global Conceptual Schema (GCS):
   - global logical view of data
   - Union of logical conceptual schemas. (LCS)

2) LCS Logical Conceptual Schema :
   - Describe logical data organisation at each site. enterprise view of entire DBs.

3) Local ~~internal~~ Internal Schema - LIS
   - Local Physical data organisation at each site..

4) External Schema: (ES)
   - Describes the user/application view of on the data.



Peer-to-Peer Architecture

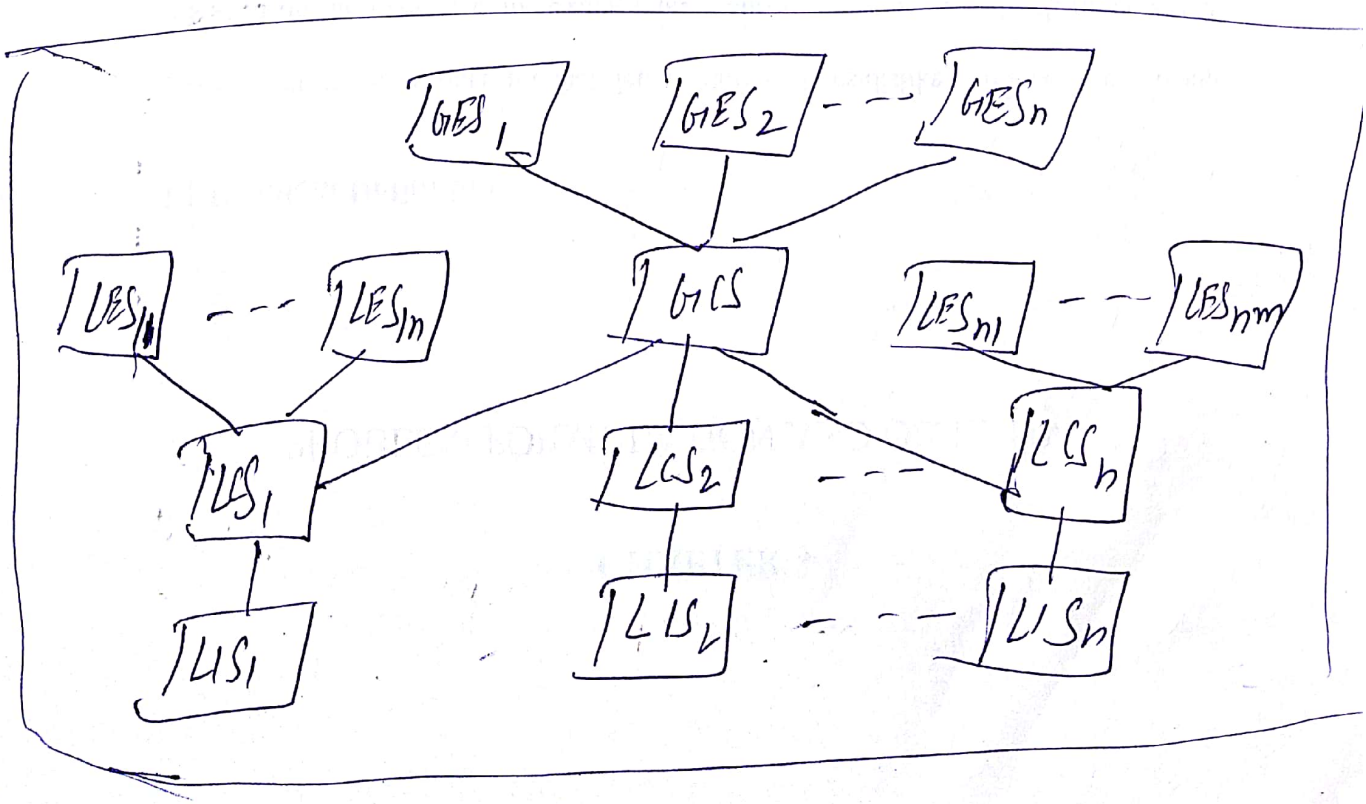
3) Multi-DBMS Architecture

Fundamental difference to peer-to-peer DBMS is the definition of the global conceptual Scheme (GCS).

- In multi-DBMS, the GCS represents only the collection of some of local databases that each local DBMS want to share.
- Data Distribution across different sites & multi DB to local data mapping.
- Two different architecture models:
  1) Models with a GCS.
  2) Models without GCS.

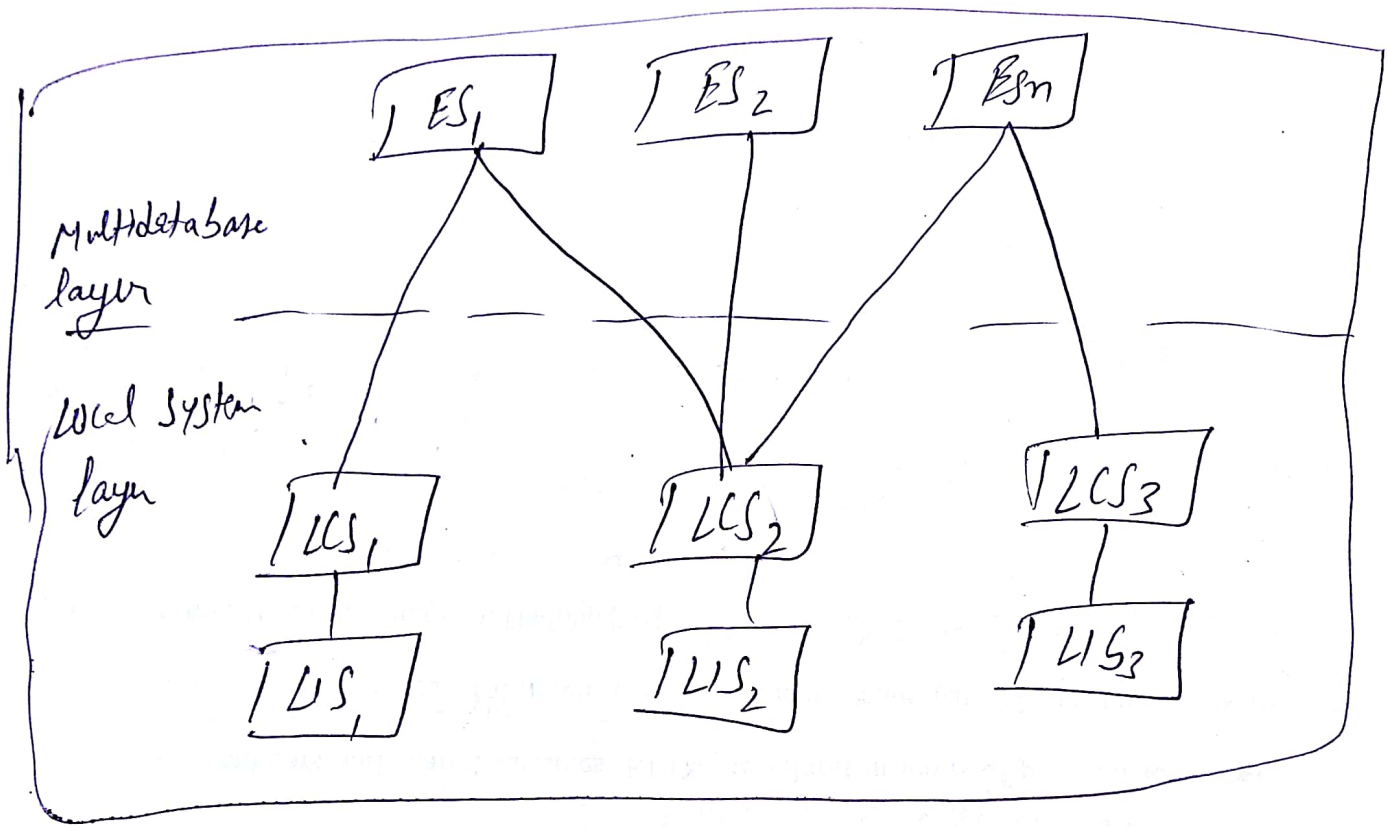
1) Model with a GCS

- GCS is a union of parts of the local DBMS.
- Define their own views on the local DB



2. Model without GCS:

- The local DBMSs present to the multi-database layer the part of their local DB they are willing to share.
- External views are defined on top of LCSs.



⇒ Data Fragmentation

Types
1) Horizontal
2) Vertical
3) Mixed (Vertical + Horizontal) (Hybrid)

Fragment of a relation is an appropriate unit of distribution.

This means :-
1) Application views are subsets of relations.
2) Locality of access of application is defines on subset of relations.
3) Number of transactions are executed concurrently by a transaction manager.
   Transaction manager maintains before and after database images, log of transactions & concurency control
4. Parallel execution of single query.

~~5) Semantic data control is more difficult.~~

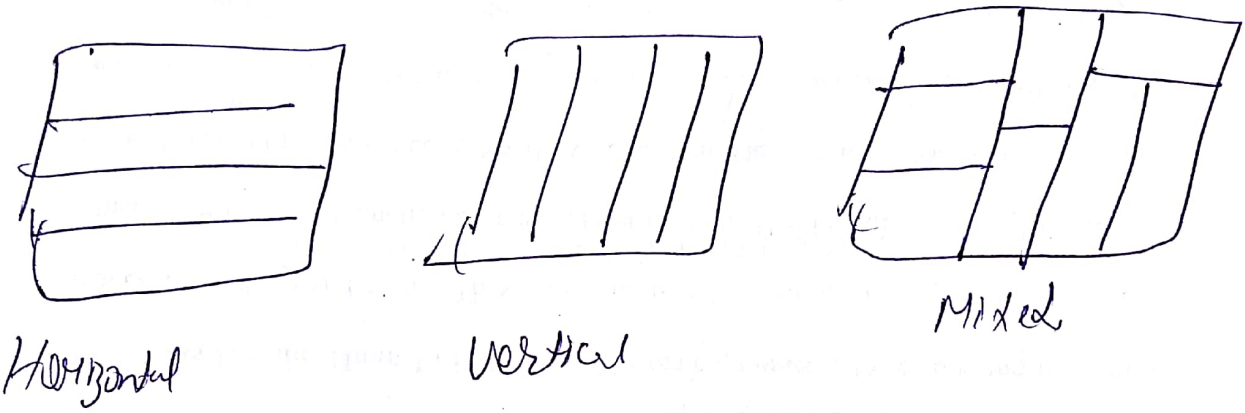
⇒ Fragmentation aims to improve :-
1) Relaibility
2) Performance
3) Balance storage capacity & costs.
4) Communication costs.
5) Security.

Types of Fragmentation

1) Horizontal Fragmentation:
Partitions a relation along its tuples (rows)

2) Vertical:
Partitions a relation along its attributes. (columns)

3) Mixed (Hybrid)
Combination of horizontal & vertical



Question PROJ

PNO	Pname	Budget	Loc
P_1	Inst	150000	Montreal
P_2	OD	135000	New York
P_3	CAD/CAM	250000	New York
P_4	Maint	310000	Paris
P_5	CAD/CAM	500000	Boston

1) Horizontal Fragmentation :-

1) PROJ1 : projects with budget less than 200000

2) PROJ2 : projects with budgets greater than or equal to 200000.

PROJ1 ← Budget < 200000 (PROJ)

PNO	PName	Budget	LOC
P1	Inst	150000	Montreal
P2	DD	135000	New York

PROJ2 ← Budget ≥ 200000 (PROJ)

PNO	PName	Budget	LOC
P3	CAD/CAM	250000	New York
P4	Maint	310000	Paris
P5	CAD/CAM	500000	Boston

2) Vertical Fragmentation

1) PROJ1 : Information about project budgets.

2) PROJ2 : Information about project names & locations

PROJ1 ← π PNO, Budget (PROJ)

PNO	Budget
P1	150000
P2	135000
P3	250000
P4	310000
P5	500000

PROJ2 ← π PNO, PNAME, LOC (PROJ)

PNO	PName	LOC
P1	Inst	Montrel
P2	DD	New York
P3	CAD/CAM	New York
P4	Maint	Paris
P5	CAD/CAM	Boston

(3) Hybrid Fragmentation :

PROJ : Project number, Project name & location with Budget > 200000

PROJ

PNo	Pname	Loc
P3	CAD/CAM	New York
P4	Maint	Paris
P5	CAD/CAM	Boston

⇒ $\Pi_{Pno, Pname, Loc}(\sigma_{budget > 200000})$

OR $\Pi_{Pno, Pname, Loc}(\sigma_{budget > 200000})$;

Most Imp ⭐⭐

⇒ Correctness Rules of Fragmentation

1) Completeness :
Decomposition of relation R into fragments $R_1, R_2, \ldots R_n$ is complete if each data item in R can be also be found in some R_i

2) Reconstruction : If a relation R is decomposed into fragments $R_1, R_2, R_3 \ldots R_n$. then fragments R. there should exist some relation operator reconstructs R from its fragments.

$$R = R_1 \nabla \ldots \nabla R_n$$

- Union to combine horizontal fragments.
- Join to combine vertical fragments.

3) Disjointness : (Data item in one fragment should not present in other fg)
If a relation R is decomposed into fragments $R_1, R_2 \ldots R_n$, & data item d_i appears in fragment R_j,

then d_i should not appear in any other fragment R_K, $K \neq J$;

- For horizontal fragmentation, data item is tuple.
- For vertical, data item is an attribute.

Imp

Replication

It is useful in improving the availability of data by copying data at multiple sites.

- A relation or a fragment can be replicated at one or more sites.
- Fully redundant databases are those in which every site contains a copy of the entire database.

There are 3 types of Replication

1) Full Replication
2) No Replication
3) Partial Replication

1) Full Replication

Whole database is copied at every site. This improves availability because the system can continue to operate as long as atleast one site is up.

2) Partial Replication

Some fragments of the database may be replicated others may not.

⇒ Data may be replicated row by row, table by table, database by database.

Allocation Techs :-

- Centralized
- Partitioned.