1) CAP Theorem

→ also known as brewer's theorem

→ fundamental principle in Distributed system

C – Consistency

A – Availability

P – Partition tolerance

Consistency (correctness)

In Distributed System multiple accessing (read/write) on same data may lead to inconsistent state will produce an error

Availability

every request should receive a response

Availability means system remain ~~operational~~ operational even incase of failures

Page No.
Date

partition tolerance

This system continue to operate despite communication breakdowns that may cause messages to be delayed or lost between nodes.

2) Distributed Database.

→ Distributed database is a database system that stores data in multiple location instead of one location through network.

Advantage

→ fast processing
→ Reliability / availability
→ reduced operating cost
→ easy to expand

Types

Homogeneous database
Heterogeneous database.

Homogeneous

→ All site have same kind of database system

Heterogeneous database

→ All sites have different kind of database

Architecture

→ each sites have own database system

→ All sites interconnected through communication network

Data storage

Data Replication

Data fragmentation

Definition of distributed databases :

- A distributed database system consists of loosely coupled sites (computer) that share no physical components and each site is associated a database system.
- The software that maintains and manages the working of distributed databases is called distributed database management system.
- The database system that runs on each site is independent of each other. Refer Fig. 5.1.1.

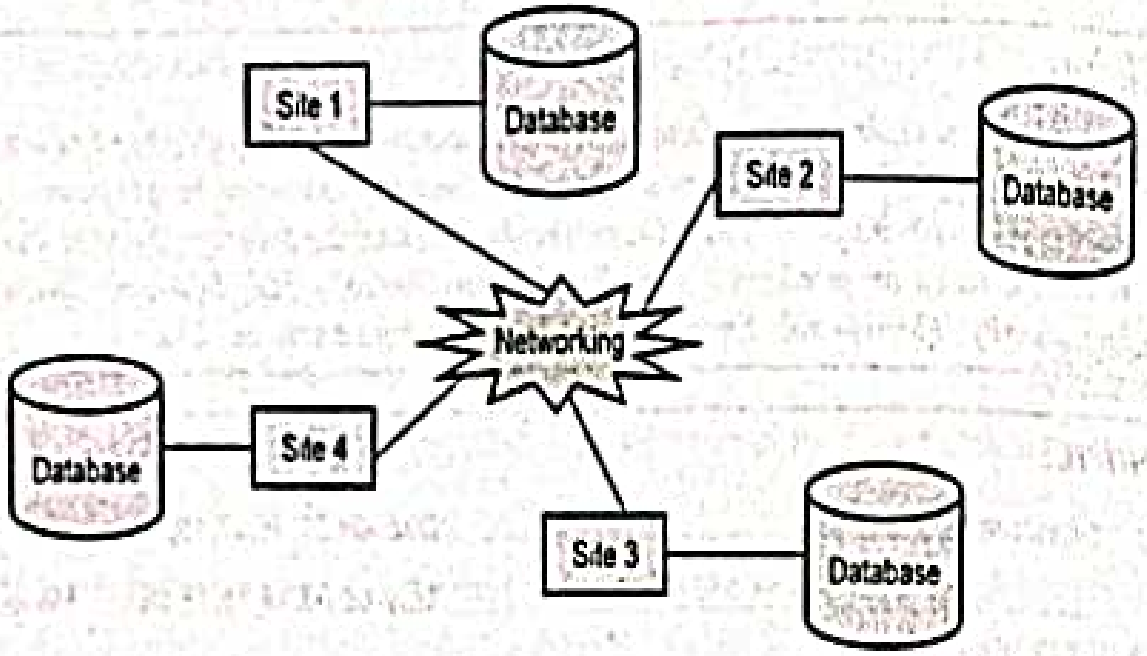


Fig. 5.1.1 : Distributed database system

The transactions can access data at one or more sites.

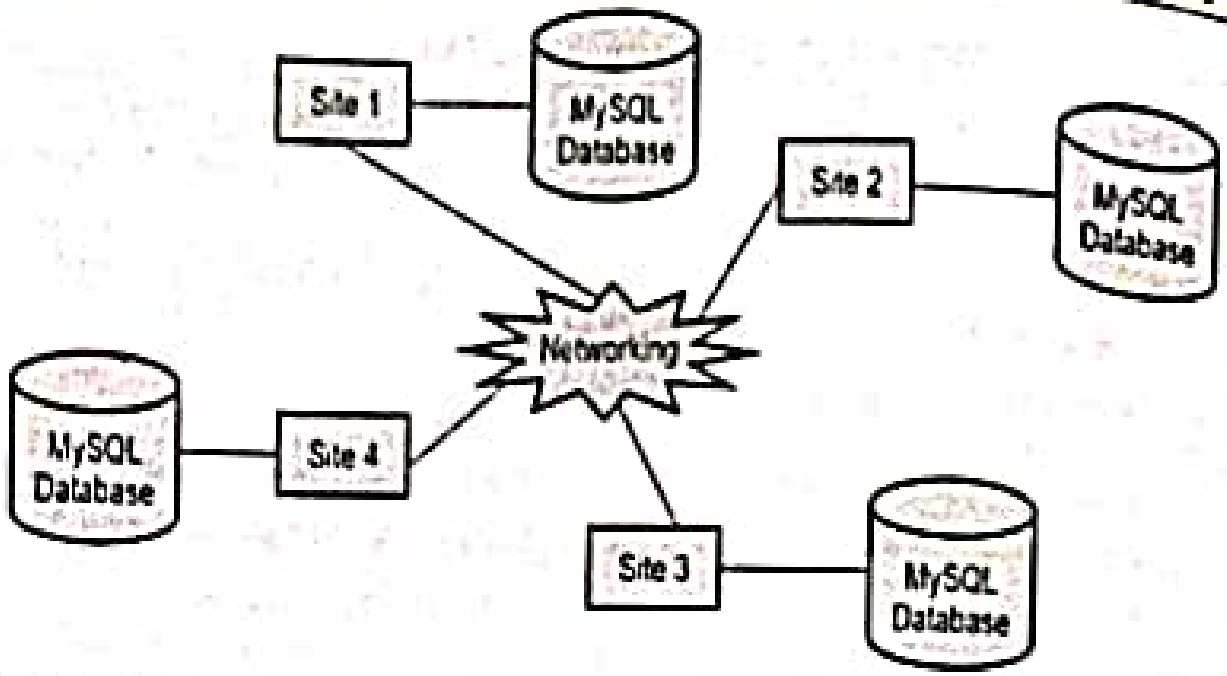
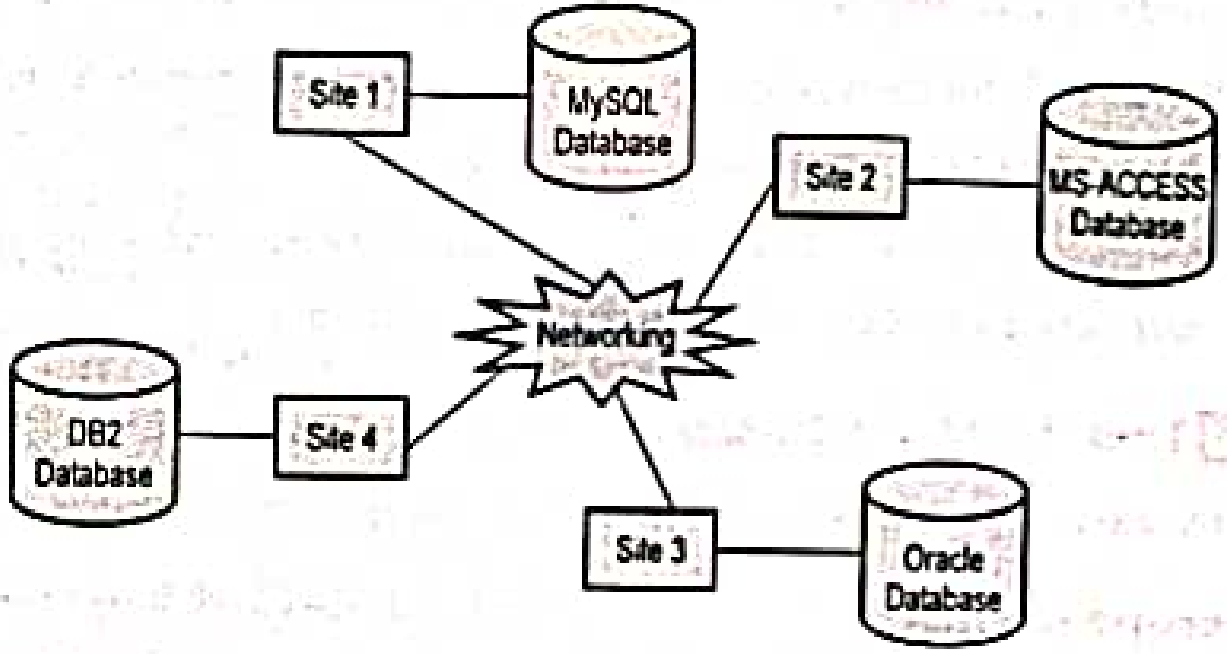


Fig. 5.1.2 : Homogeneous databases

- In this system, all the sites are aware of the other sites present in the system and they all cooperate in processing user's request.
- Each site present in the system, surrenders part of its autonomy in terms of right to change schemas or software.
- The homogeneous database system appears as a single system to the user.

(2) Heterogeneous databases

- The heterogeneous databases are kind of database systems in which different sites have different schema or software. Refer Fig. 5.1.3.



5.1.2 Architecture

- Following is an architecture of distributed databases. In this architecture the local database is maintained by each site.
- Each site is interconnected by communication network.

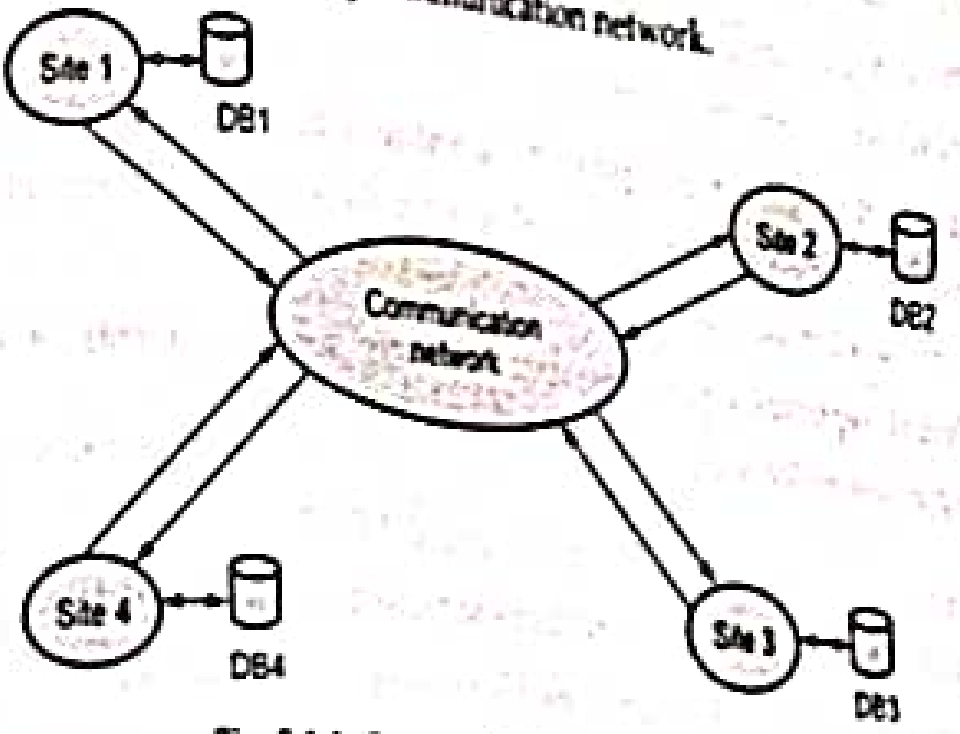
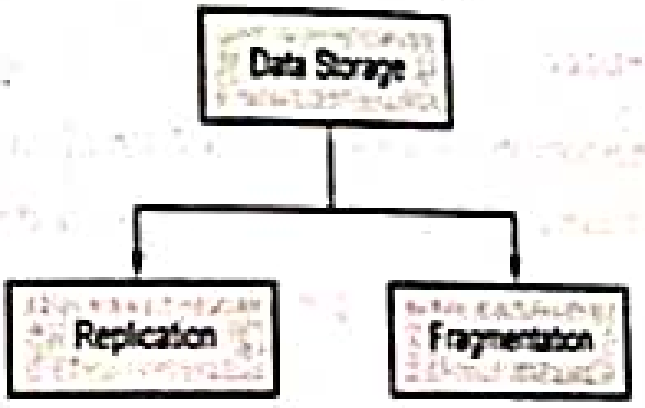


Fig. 5.1.4 : Distributed database architecture

When user makes a request for particular data at site Si then it is first searched at the local database. If the data is not present in the local database then the request for that data is passed to all the other sites via communication network. Each site then searches for that data at its local database. When data is found at particular site say Sj then it is transmitted to site Si via communication network.

5.1.3 Data Storage

There are two approaches of storing relation r in distributed database -



(1) Replication : System maintains multiple copies of data, stored in different sites, for

Data Replication

→ Storing copy of data in two or more sites

Page No.
Date

full replication:

entire relation is stored in all sites

partial replication

Some fragments of relation are stored in other sites

Advantages

availability

Fast accessing

Data fragmentation

→ fragmentation is a devision of relation r into fragments $r_1, r_2, r_3, \ldots r_m$

2types

Horizontal fragmentation

vertical fragmentation

Horizontal fragmentation

each tuple of "r" is fragmented (divided)

relation R fragmented to $r_1, r_2, \ldots$

$[R = r_1 \cup r_2 \cup r_3]$

Vertical fragmentation

Column in relation R are fragmented.

$R \rightarrow r_1, r_2, r_3 \dots r_m$

$[R = r1 \bowtie r2]$ $\bowtie$ join

2) ~~NO SQL Injection~~ // extract user data by injecting web page input as Statement through SQL commands'.

A) SQL Injection:

→ SQL injection is a type of code injection technique that might destroy the databases

→ this is a kind of malicious code which attack database

Sql injection prevention

→ only away to prevent SQL injection is to validate every input field.

→ another method parameterized query
↳ called as prepared statement

by this way apln code never use the input directly

The intruder retrieve all the user data present in the database, such as user details, credit card information .... etc

No SQL Database

Stands For

→ Not only SQL

→ non tabular database System that Stores non tabular data.

Types

→ Key-value store

→ Document store

→ Graph based

→ Wide colum store

Need

used for storing modeling structured, semi structured and unstructured data

Scalability

cost effective

~~Dege~~ Does't need schema or relaxed schema

RDBMS	NO SQL
It uses Sql quere	uses unstructured query language
→ Predefined Schema	Doesn't have Schema or relaxed schema
→ Ex: MySQL, Oracle, Postgre Sql	MongoDB, BigTable, Redis

Key-value Store

→ Simplest type

→ key-value
key - is - unique
value is can be JSON, String or binary

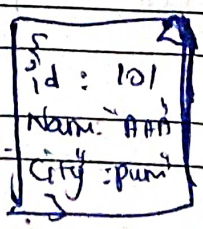
ex

```
{customer
    {
    { "id" :1 "name": "Ankita"}
    {"id": 2, "name": "Kavitha"}
    }
}
```

Document based System (ex: mongoDB, couchDB)

→ The document is stored in the form of XML and JSON

→ This type uses key-value pair type to store and retrieve data



... popular document oriented NOSQL database.

5.4.3 Column Based Systems

- The column store model is similar to traditional relational database. In this mode the columns are created for each row rather than having predefined by the tabl structure.
- In this model number of columns are not fixed for each record.
- Columns databases can quickly aggregate the value of a given column.
- For example -

Row ID	Columns...		
1	Name	City	
	Ankita	Pune	
2	Name	City	email
	Kavita	Mumbai	kavita123@gmail.com

The column store databases are widely used to manage data warehouses, business intelligence, HBase, Cassandra are examples of column based databases.

Graph Database

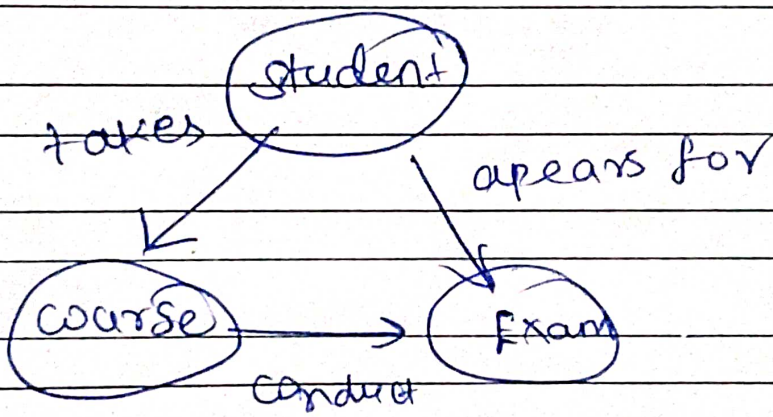
→ Graph Database is used where relationship among the element exist.

Components.

Node : entity

edge : relationship among node

Link : connect 2 entity



Database security

→ technique provide protection

→ restrict unathorized user to access data.

(3m) Types of cryptography

1. Symmetric key encryption
2. Asymmetric key encryption

(2m) Digital signature

→ Is a mathamatical scheme for authentication

→ If the recipient gets a message with digital signature then he belives that the message was created by known sender

Query processing and optimization in database management systems (DBMS) are crucial aspects that determine the efficiency and performance of database operations. Here's a detailed explanation of each component:

of each component.

Query Processing:

1. Parsing:

- **Lexical Analysis:** The query is broken down into tokens (keywords, identifiers, operators).
- **Syntax Analysis:** Tokens are checked against the grammar rules of the query language (e.g., SQL) to form a parse tree.

2. Semantic Analysis:

- The query's validity is checked against the schema of the database (tables, columns, relationships).

3. Optimization:

3. Optimization:

- **Query Rewrite:** The optimizer may rewrite the query to improve efficiency (e.g., using equivalent transformations).
- **Cost Estimation:** The optimizer estimates the cost of different query execution plans based on statistics (like table sizes, index selectivity).
- **Plan Selection:** The optimizer chooses the best execution plan based on cost estimates.

4. Query Execution Plan Generation:

- The selected execution plan defines how data will be retrieved and processed.
- It includes operations like joins, selections (filters), aggregations, and sorting.

5. Execution:

- The DBMS executes the query plan using algorithms like nested loop joins, hash joins, sort-merge joins, etc.
- Data is fetched from storage (disk or memory) and processed according to the query plan.

Query Optimization:

1. Goals of Optimization:

- **Minimize Response Time:** Reduce the time taken to execute the query.
- **Minimize Resource Utilization:** Optimize the use of CPU, memory, and disk I/O.
- **Maximize Throughput:** Handle multiple queries efficiently.

2. Techniques Used:

- **Cost-Based Optimization:** Estimates the cost of various execution plans and selects the one with the lowest estimated cost.
    - **Statistics:** Information about data distribution (like table sizes, index selectivity) used to estimate costs.
    - **Query Rewriting:** Transformations to simplify or improve the query.