UNIT V: NoSQL Databases

Introduction to Distributed Database System:

A DDBMS is a collection of databases distributed accross multiple location connected via network.

key features:

Data distribution: Data is distributed accross various locations (horizontally or vertically)

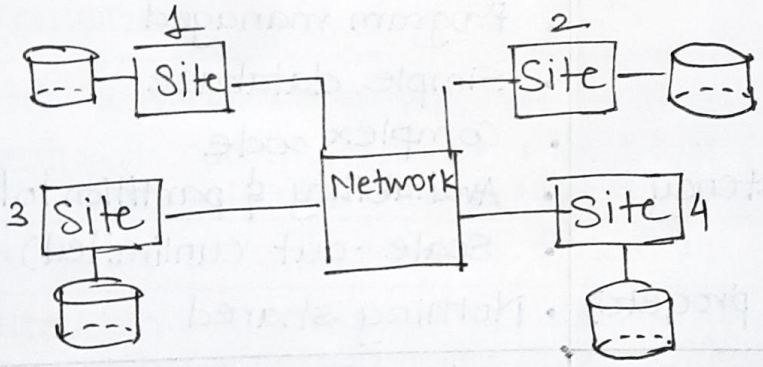
Autonomy: Can act/operate independantly

Transparency Users need not know the storing location

Scalability Easily expandable by adding more nodes

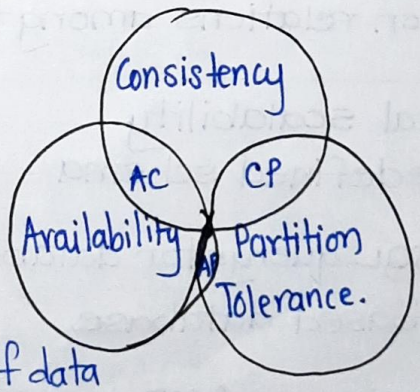
Reliability Data is accessible even during site or network failur

Concurrency control: Ensures consistency accross multiple users. simultaneously.



CAP THEOREM.

It states that all three of the qualities cannot be concurrently guarenteed in any distributed system



C - All nodes in system have same view of data

A - The system remains accessible even if some nodes are down

P - The system can operate even if some nodes are disconnected.

BASE PROPERTIES.

1. Basic Availability
   - DB should always be available to respond to user request
2. Softstate
   - DB can change over time without any user due to BG process
3. Eventual consistency.
   - DB might not be immediately consistent. but it will become consistent after updates reach other nodes.

ACID (relational)	BASE (NoSQL)
• Strong consistency	• Weak consistency
• Isolation	• Last write wins
• Transaction	• Program managed
• Robust database	• Simple database
• Simple code	• Complex code
• Availability & Consistency	• Availability & partition tolerance
• Scale up (limited)	• Scale-out (unlimited)
• Shared (disk, mem, proc, etc)	• Nothing shared

RDBMS	NO SQL
• based on relations among the tables	• non relational, can be used in distributed environment
• vertical scalability	• horizontal scalability
• has predefined schema	• either no schema or relaxed schema.
• uses SQL query for database	• uses unstructured query
• table based database	• document, graph, key-value based.
• emphasis on ACID property	• emphasis on BASE property and CAP theorem
• example: Oracle, mySQL, PostgreSQL	• example: MongoDB, BigTable, Redis.

MONGO DB

MongoDB is a document based database
It was developed and supported by a company named 10gen which is now known as MongoDB inc

features:

- Scheme-less document based DB
- doesnt follow any relational model
- either has no schema or relaxed schema
- It is both structured and semi structured DB
- It is highly scalable thus widely used.
- It is cost effective.
- Provides high performance.
- supports data in form of key value pair.
- provides horizontal scalability with the help of sharding (distributing data on multiple servers)

Operations:

Create : ~~db.collectionnam~~ db.createCollection('collection name')

Insert: db.collection name.insertOne({f1:v1, f2:v2});
db.collection name.insertMany({f1:v1},{f1:v2});

Find : db.collectionname.find();

Update: db.collectionname.updateOne(cond);
db.collection name.updateMany(cond);

Delete: db.collectionname.deleteOne();
db.collectionname.deleteMany();

Drop DB: dropDatabase();

Removing doc: db.collectionname.remove(cond);

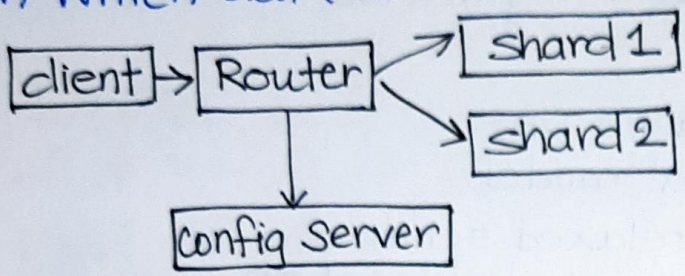
Drop collection: ~~db.drop~~ db.collectionname.drop();

Aggregation: It is used to group data from multiple location and perform some operations onto ~~for~~ it to produce some result. It is similar to count(*) and groupby clause in SQL

example: $match(), $group(), $sort(), $sum, $avg, $min, $max

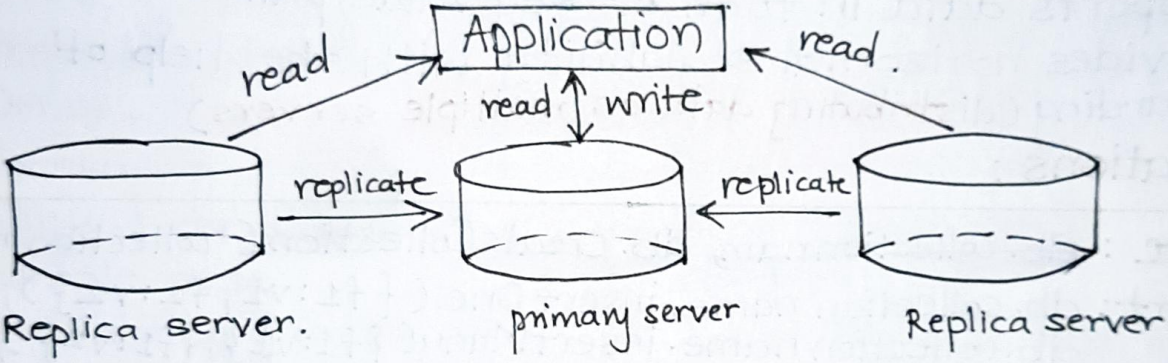
Sharding in MongoDB

Sharding is a concept in mongodb, which helps achieve distributed data into different locations. This method in which data is allocated accross multiple machine.



REPLICATION in MongoDB

Replication is the method of duplication of data across multiple servers in mongoDB



MAP REDUCE

Map reduce is a data processing programming model that helps to perform operations on large data sets and produce aggregated results.

In mongo this has two main functions

map function : `var map1 = function() {emit (key, value);};`

reduce function : `var reduce1 = function(key, value){ };`

then `db.collectionname.mapReduce ( map1, reduce1, {query:{ }, out : "collection name" } );`

Index

Create simple index : db.collectionname.createIndex({f1:1});
compound index db.collectionname.createIndex({f1:1,f2:1});

Get Index : db.collectionname.getIndex();

Index are used for uniquely identifying each data set
This makes operations like reading & searching more efficient.