4. MongoDB

1) Compare & Contrast MongoDB w traditional RDBMS.

RDBMS	MongoDB
⊛ Relational database	* non-Relational & document oriented.
* predefind Schema	* Dynamic Schema
* vertically Scalable	* Horizontally scalable
* not suitable for large data storage	* suitable for large data storage
* It's row baged	* It's document baged
* supports Complex joins	* no support for complex joins.
* follows ACID properties	* follows CAP theorm
* It's column baged	* It's field baged
* It support's SQL only	* It supprts JSON & also SQL

(2) Explain following

(a) Sharding :- It's a technique used in MongoDB to horizontally partition data accross multiple servers or nodes to improve scalability & performance

→ The components of sharding are :-

- Shards :- Shards are used to store data. provide high availability & data consistancy. In production environment each shard is a seperate replica set.

- Config Servers :- It stores cluster's metadata.
  * This data contains a mapping of cluster's data set to the shards.

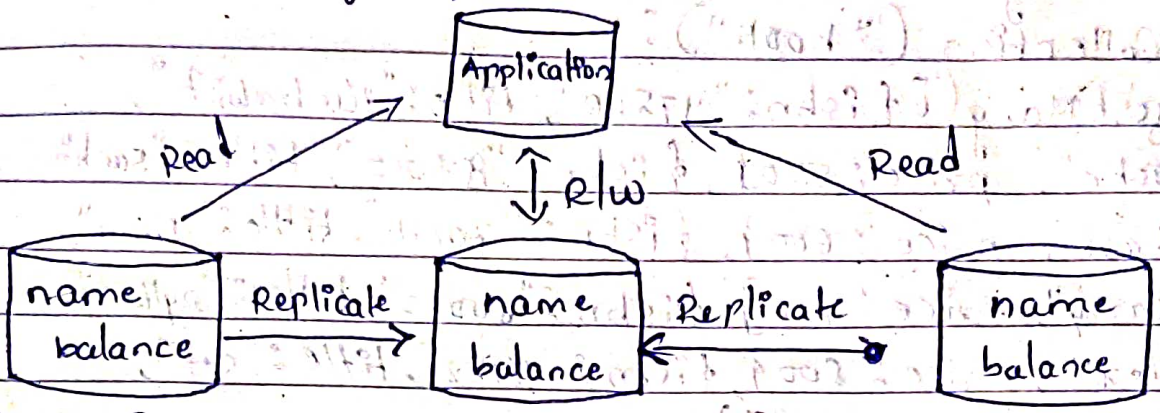
- Query Routers :- are responsible for routing client requests to the appropriate shard baged on shard key.
- Shard key :- It's field in each doct. i.e how data is distributed across shards.

Ex:- use <database name>

db.createCollection("collectionname");

sh.shardCollection("<db-name>.<collname>", {"<sharding-key>":1})

(b) Replication :- It involves creating copies of data & distributing them to multiple servers or nodes to ensure data availability, fault tolerance & data redundancy



* Consider above diagram, we have an applic^2 & it read & write data & says thy to server A which hay name & balance which will replicate to two other servers in 2-diff loca^2.
* MongoDB uses primary-secondary replic^2 model, where one node serves as primary source of data, while other replicate data from primary.
* If the primary node fails, one of the secondaries can be automatically promoted to primary role.

(c) CRUD opera^2 :-
→ Create :- It's used to insert new documents into coll^2 using insertone, insertMany or bulkwrite methods.
→ Read :- used to retrieve data from colle^2 using find, findone & aggregation opera^2.
→ Update :- used to modify existing documents in a coll^2. using Updateone, UpdateMany & replaceone.
→ Delete :- used to remove documents from colle^2 using Deleteone, DeleteMany.

(3) To create colle2 by name book having columns (isbn, title, author, price) & insert 5 records

→ Use myLibrary;

db.createCollection("book");

db.book.insertMany([{isbn:"97840", title:"The habit", author:"John", price:500}, {isbn:"87000", title:"sachin", author:"hari", price:600}, {isbn:"70011", title:"Java", author:"Carry", price:700}, {isbn:"8002r", title:"python", author:"alex", price:800}, {isbn:"7182", title:"c++", author:"warn", price:900}]);

→ query to search a book title is :-

db.book.find({title:"Java"});

4] What is map-reduce architecture? explain w. ex.

=> It's data processing technique used for aggregating & transforming data with in the database.

* It allow you to perform complex data analysis & aggregation operations on large data sets.
* MongoDB uses map-Reduce command for map-reduce operations.
* Generally used for large data sets.

parameters of Map-Reduce are:-

(a) Map function:- It takes documents from a mongoDB collec² as I/P & emits (produce) key value pairs as o/p.

* The emitted key-value pairs are typically used for grouping & sorting the data for further processing in the Reduce step.

Ex:- ~~let's say you have collec² of documents representing website access data.~~

```
{
    "page" : "/homepage"
    "age" : 25
```

(b) Reduce function:- used to process & aggregate data that shares the same key generated by the map func².

* It takes key & an array of values as I/P & produces a single o/p value for that key.

ex:-
```
var mapFunction = function()
{ emit(this.age, { count: 1, totalage: this.age });
};                    // Map func²

var reduceFunction = function (key, values)
{ var result = { count:0, totalage:0 };
  values.forEach (function(value)
  { result.count += value.count;
    result.totalage += value.totalage;
  });
  return result;
};
```

(5) (a) Select stdname, course, grade from student where course_id = MCA

=> db.student.find({course: 'MCA'}, {_id:0, stdname: 1, course: 1, grade: 1});

(b) select stdname, course, grade from student where grade <> 'F'.

=> db.student.find({grade: {$ne: 'F'}}, {_id:0, stdname: 1, course: 1, grade: 1});

(6) (a) Ans. 5 (a)

(b) Update student set grade = 'A' where id = 4

db.student.updateOne({id: 4}, {$set: {grade: 'A'}});