**Mongo DB Database**

**Window User :**

Open the terminal or command prompt inside **C:\Program Files\MongoDB\Server\7.0\bin**

**mongod : this command is use to run the mongo db server.**

Before running this command we need to create one directory or folder inside C drive.

C: Drive

**data** 🡪folder

**db** -🡪folder

Non window user

Open the terminal

sudo mkdir -p /data/db creating the folder.

sudo chown -R $(whoami) /data/db provide the permission

now run the command as mongod

**mongod** -🡪 to run the server

**mongo**  -🡪 to run client terminal old version

**mongosh** -🡪run the client terminal new version to run client terminal

show databases; it is use to display all databases present in mongodb

show dbs;

use databasename; it that database not present it will create and switch inside that database. If already present then it move inside that database.

show collections;

Or

show tables; it display all tables present in current db.

Creating collection in mongo db.

Mongo DB provided pre defined object it db. Which contains lot of function which help to do operation as create, view, delete, update etc.

db.createCollection(“CollectionName”)

inserting document inside a collection

db.collectionName.insertOne(**{key:value}**);

**db.Sample.insertOne({"name":"Steven"});**

db.Sample.insertMany([{name:"Ajay",city:"Bangalore"},{name:"Vikash",age:34},{fname:"Raj",lname:"Deep"}]);

**To view documents from a collections**

db.collectionName.find();

mongo db by default \_id attribute created for each document with unique value.

In mongo db \_id is like primary key. But in mongo db \_id is a pre-defined attribute. We can’t make any other column as PK. If you want to insert unique value you can pass the attribute name must \_id.

In Mongo DB we can insert the document in a collection without creating a collection.

db.Employees.insertMany(

[

{\_id:100,name:"Steven",age:25,salary:45000,deptid:1,city:"Bangalore"},

{\_id:101,name:"Raj",age:27,salary:42000,deptid:2,city:"Mumbai"},

{\_id:102,name:"Ajay",age:29,salary:41000,deptid:3,city:"Bangalore"},

{\_id:103,name:"Leena",age:35,salary:49000,deptid:1,city:"Delhi"},

{\_id:104,name:"John",age:38,salary:48000,deptid:1,city:"Bangalore"},

{\_id:105,name:"Root",age:21,salary:46000,deptid:2,city:"Mumbai"},

{\_id:106,name:"Butler",age:27,salary:45000,deptid:2,city:"Delhi"},

{\_id:107,name:"Reeta",age:22,salary:42000,deptid:3,city:"Bangalore"},

{\_id:108,name:"Meeta",age:29,salary:43000,deptid:3,city:"Delhi"},

{\_id:109,name:"Leeta",age:30,salary:42000,deptid:2,city:"Bangalore"},

{\_id:110,name:"Keeta",age:32,salary:40000,deptid:1,city:"Mumbai"}

]

);

**View control functions**

limit() function : this function is use to view top n document present in collection

db.Employees.findOne(); : it is use to retrieve top most document present in collection without condition.

skip() function : it is use to skip number of document from a collection.

Sort() function :

db.Employees.find().sort({salary:-1}); desc -1 descending order

db.Employees.find().sort({salary:1}); asc 1 ascending order

filter the document using conditions like where clause in MySQL Database.

db.Employee.find({filedName:value})

**db.Employees.find({\_id:100});**

**db.Employees.find({city:"Bangalore"});**

to apply relational operator condition mongo db provided lot of pre defined keyword or operator start with pre-fix $ followed by operator name.

db.Employee.find({salary:{$gt:45000}});

db.Employees.find({salary:{$gt:45000}});

db.Employees.find({salary:{$gte:45000}});

db.Employees.find({salary:{$lt:45000}});

db.Employees.find({salary:{$lte:45000}});

db.Employees.find({salary:{$eq:45000}});

**or**

db.Employees.find({salary:45000});

db.Employees.find({salary:{$ne:45000}});

if we want to check more than one condition ie and/or operator so mongo db provided

$and / $or

**db.Employees.find({$and:[{\_id:100},{salary:45000}]});**

**db.Employees.find({$and:[{\_id:100},{salary:{$gte:45000}}]});**

**db.Employees.find({$or:[{\_id:101},{salary:45000}]});**

view specific fields value from a document rather than all fields.

**select \* from employees; db.Employees.find();**

**select \* from employee where id=100**

**db.Employees.find({\_id:100})**

**select id,name from employees;**

**db.Employees.find({},{name:1})**

**select id,name from employees where salary >45000**

**db.Employees.find({salary:{$gt:45000}},{name:1})**

db.Emloyees.find({condition},{fieldname:1})

db.Employees.find({},{name:1,age:1}); display name, age and \_id field

db.Employees.find({},{name:1}); display name and \_id field

db.Employees.find({},{name:1,age:1,\_id:0}); display name and age field

**update document field value**

**if we are going to update with \_id field condition then use updateOne else use updateMany**

db.Employees.updateOne({condition},{$set:{field1:value1,field2:value2}})

db.Employees.updateMany({condition},{$set:{field1:value1,field2:value2}})

db.Employees.updateOne({\_id:100},{$set:{salary:48000}});

db.Employees.updateMany({deptid:1},{$set:{salary:50000}});

**remove document from a collections**

db.Employees.deleteOne({\_id:106})

db.Employees.deleteMany({deptid:3});

In mongo DB each document allow us store value in the form of JSON. So key must be unique and value can be number, string, Boolean, array type, object type or array of object etc.

**Student**

\_id, sname, age skillset : 0 or 1 or many

db.StudentInfo.insertMany([

{\_id:1,sname:"Steven",age:21},

{\_id:2,sname:"John",age:24,skillSet:["Java"]},

{\_id:3,sname:"Reeta",age:23,skillSet:["Python","AI","Ml"]},

{\_id:4,sname:"Neena",age:22,skillSet:["HTML","CSS","JS","Angular"]},

{\_id:5,sname:"Raj",age:27,skillSet:["C","C++"]}

]);

If we want to find one or many document particular field is missing.

db.StudentInfo.find({skillSet:{$exists:false}}); skillset field is missing

db.StudentInfo.find({skillSet:{$exists:true}}); skillset fields must be present.

Apply condition for array field

**db.StudentInfo.find({skillSet:"Angular"});**

**db.StudentInfo.find({skillSet:"Java"});**

**Relationship :**

One to many 🡪 Trainer handling more than one students

One to one -🡪 person has only one passport

Many to many 🡪 Many student know more than one technologies.

In Mongo DB we make relationship on document level rather than collection. But in MySQL we do relationship on table level not on record level.

In mongo DB we can achieve relationship concept using

1. Embedded style : storing all information in only one collection.
2. Linking style : storing all information in more than one collection.

In MySQL

One to many relationship in mysql

**TechnologyTrainerRelationship**

**PK FK FK**

**Srno TecId TId**

A 1111 100

B 2222 100

C 2222 101

**Technology**

**PK**

**TechId TName**

1111 Java

2222 Python

**Trainer**

**PK**

**Tid TName**

100 John

101 Steven

**Student**

**PK** **FK**

Sid SName age TId

1 Seeta 21 100

2 Meeta 22 100

3 Veeta 23 101