**MongoDB / Mysql**

For Node.js we want to use database so you can use **Mongo db or Mysql** , But **most of use Mongo db**

MongoDB is an **open-source** document database and leading **NoSQL** database. **MongoDB is written in C++.**

**What is mongodb**

* No sql database / **means we can not write select & query**
* Data store in **collection**
* Collection **don’t have row and column**
* Data **stored in form of object**

Database

**Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.**

Collection

**Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.**

**Custormer\_collection**

**[**

**{**

**Id:”1”,**

**Name:”nagar”,**

**Mobile:”1234567891”**

**},**

**{**

**Id:”1”,**

**Name:”nagar”,**

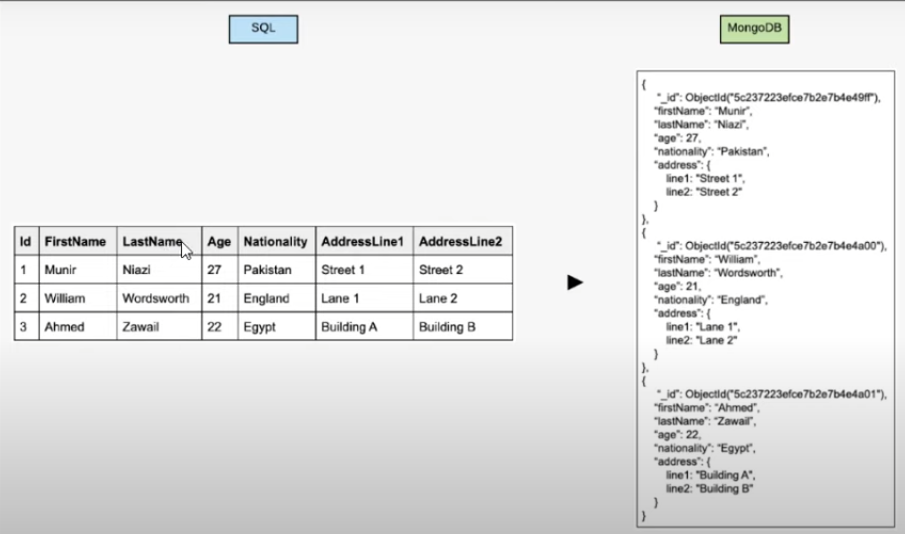
**Mobile:”1234567891”**

**},**

**]**

Document

**A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.**

****

**The following table shows the relationship of RDBMS terminology with MongoDB.**

|  |  |
| --- | --- |
| **RDBMS** | **MongoDB** |
| **Database** | **Database** |
| **Table** | **Collection** |
| **Tuple/Row** | **Document/obj** |
| **column** | **Field** |
| **Table Join** | **Embedded Documents** |
| **Primary Key** | **Primary Key (Default key \_id provided by MongoDB itself)** |
| **Database Server and Client** | |
| **mysqld/Oracle** | **mongod** |
| **mysql/sqlplus** | **mongo** |

**Download  & Install**

[**https://www.mongodb.com/try/download/community-edition**](https://www.mongodb.com/try/download/community-edition)

Next -> next : Installed

C:\Program Files\MongoDB\Server\6.0 in this folder

**Data & bin** is the most important folder in it.

**Note : MongoDB AFTER 6 VERSION mongo command not work**

Check installed or not:

cmd : **mongodb —-version**

**Use Mongosh in Mongodb Compass**

**Manage Database by command in mongosh**

**========================================**

Note : Below use in mongodb compass

Cmd :**show dbs**  // show collections //show all db or collections

Cmd : **use databasename  // Go in Database**

**Cmd : db.dropDatabase()**

Cmd : **db.createCollection(‘users’)**  // create collection /table

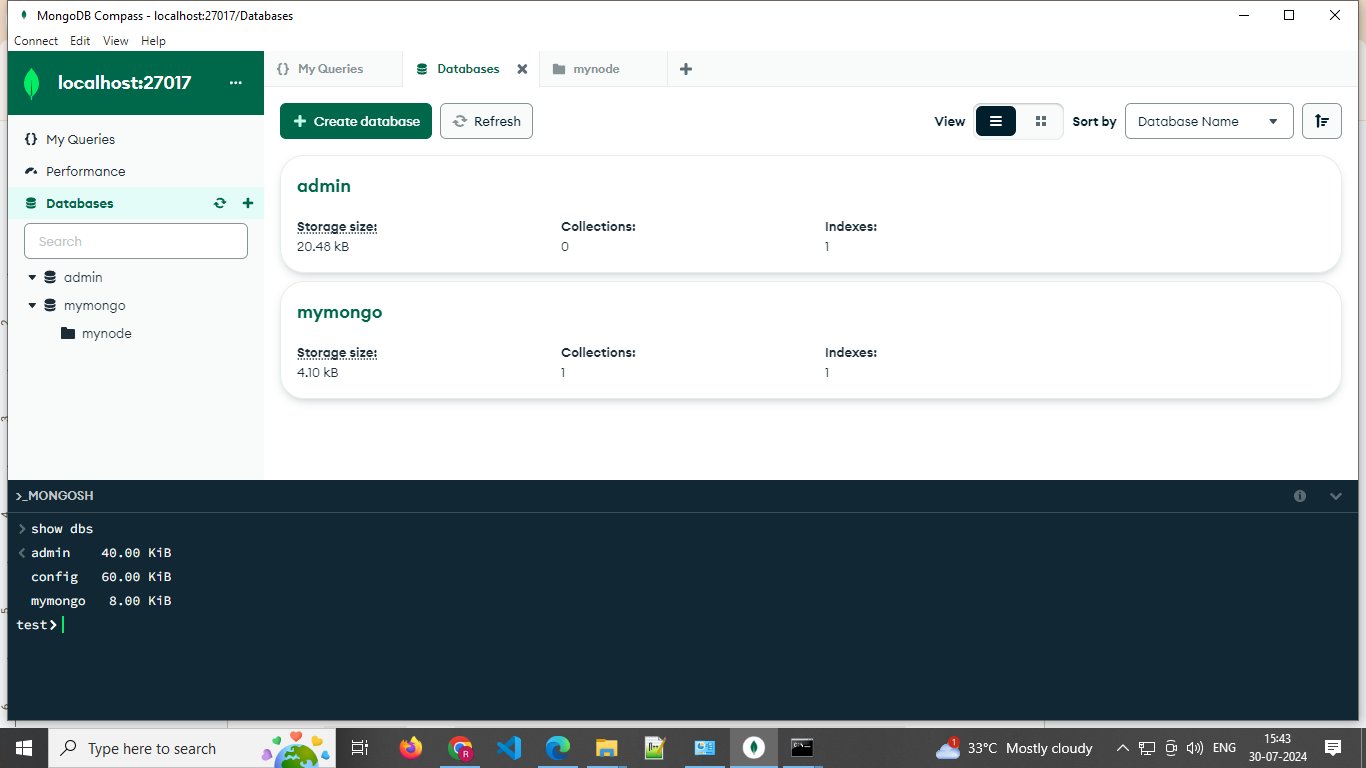
Cmd : **show collections**  // show all collection /table

Cmd : **db.users.drop()**  // delete collection /table

Cmd : **db.collections**  // show collection /table List

Cmd : **db.users.insert({"name":"tutorials point"})** // add column

db.customer.insertMany([{"name":"rajesh nagar","mobile":"9722041171","email":"rajeshnagar@gmail.com","password":"1234"}])



Note: If some time not install proper due to path not set then

Copy bin folder path & search env=> advance=> env variable => path => add new path

Mongo Compass : GUI TOOLS LIKE MYSQL

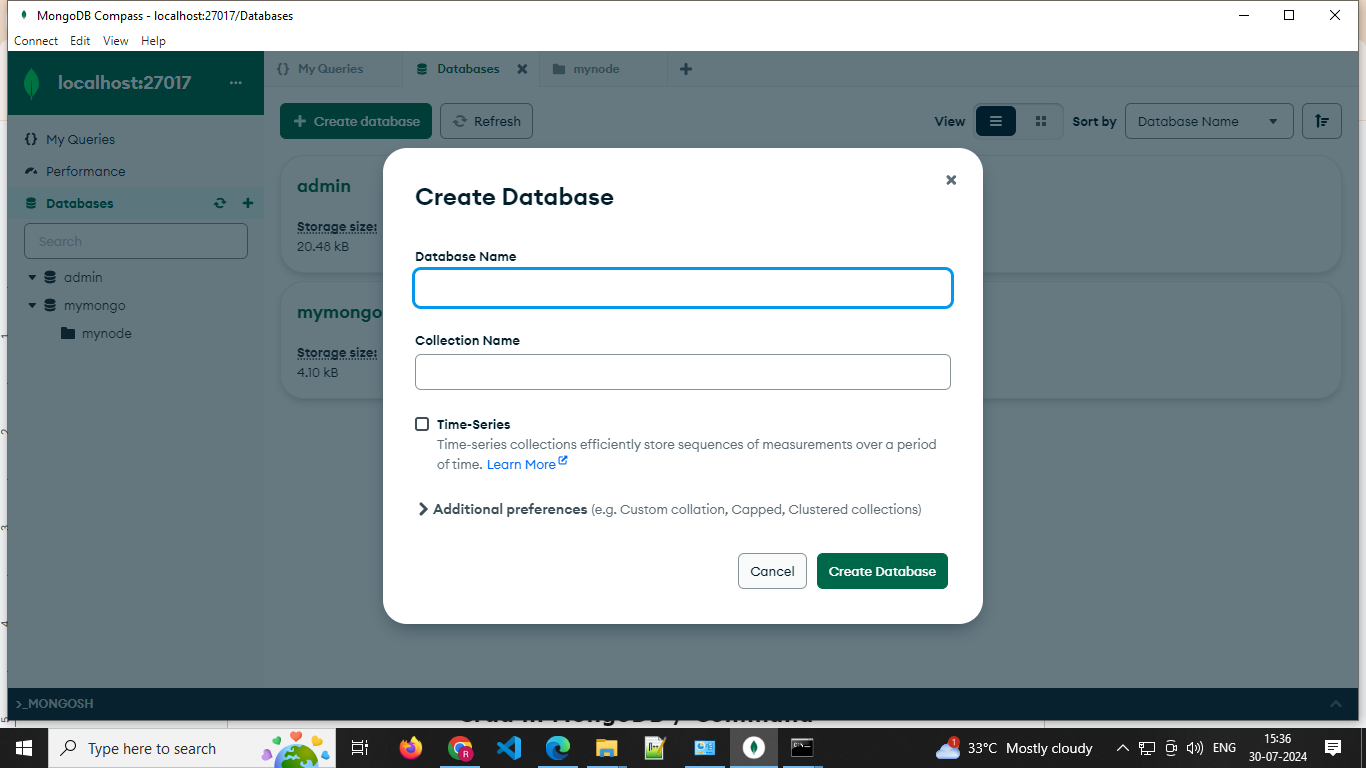
CREATE : DATABSE

CREATE : COLLECTION // LIKE SQL TABLE

Cmd : mongod --version  // 6.01 version  //old only mongo

**Database Create**

Create Database > Database Name > collection name



Que : Diffe between mongo & mysql

Ans 👍

Que : Mongo Path env setup

Ans:

Que Mongo gui tools

Ans: Mongodb Compass

**Crud in MongoDB / Command**

**create a new database :** use DATABASE\_NAME

**Show all db :** show dbs

**Your created database (mydb) is not present in list. To display database, you need to insert at least one document into it.**

**>db.movie.insert({"name":"tutorials point"})**

**db.dropDatabase()**

**Basic syntax of createCollection() command is as follows −**

**db.createCollection(name, options)**

**>db.createCollection("mycollection")**

**{ "ok" : 1 }**

**=================================================================**

**>show collections**

**In MongoDB, you don't need to create a collection. MongoDB creates a collection automatically, when you insert some document.**

**>db.tutorialspoint.insert({"name" : "tutorialspoint"}),**

**WriteResult({ "nInserted" : 1 })**

**Basic syntax of drop() command is as follows −**

**db.COLLECTION\_NAME.drop()**

===================================================================

**Example : Insert**

**db.createCollection("post")**

**> db.post.insert([**

**{**

**title: "MongoDB Overview",**

**description: "MongoDB is no SQL database",**

**by: "tutorials point",**

**url: "http://www.tutorialspoint.com",**

**tags: ["mongodb", "database", "NoSQL"],**

**likes: 100**

**},**

**{**

**title: "NoSQL Database",**

**description: "NoSQL database doesn't have tables",**

**by: "tutorials point",**

**url: "http://www.tutorialspoint.com",**

**tags: ["mongodb", "database", "NoSQL"],**

**likes: 20,**

**comments: [**

**{**

**user:"user1",**

**message: "My first comment",**

**dateCreated: new Date(2013,11,10,2,35),**

**like: 0**

**}**

**]**

**}**

**])**

The insertOne() method

**db.empDetails.insertOne(**

**{**

**First\_Name: "Radhika",**

**Last\_Name: "Sharma",**

**Date\_Of\_Birth: "1995-09-26",**

**e\_mail: "radhika\_sharma.123@gmail.com",**

**phone: "9848022338"**

**})**

The insertMany() method

**db.empDetails.insertMany(**

**[**

**{**

**First\_Name: "Radhika",**

**Last\_Name: "Sharma",**

**Date\_Of\_Birth: "1995-09-26",**

**e\_mail: "radhika\_sharma.123@gmail.com",**

**phone: "9000012345"**

**},**

**{**

**First\_Name: "Rachel",**

**Last\_Name: "Christopher",**

**Date\_Of\_Birth: "1990-02-16",**

**e\_mail: "Rachel\_Christopher.123@gmail.com",**

**phone: "9000054321"**

**},**

**{**

**First\_Name: "Fathima",**

**Last\_Name: "Sheik",**

**Date\_Of\_Birth: "1990-02-16",**

**e\_mail: "Fathima\_Sheik.123@gmail.com",**

**phone: "9000054321"**

**}**

**]**

**)**

===================================================================

MongoDB Query Operators

**There are many query operators that can be used to compare and reference document fields.**

Comparison

**The following operators can be used in queries to compare values:**

* **$eq: Values are equal**
* **$ne: Values are not equal**
* **$gt: Value is greater than another value**
* **$gte: Value is greater than or equal to another value**
* **$lt: Value is less than another value**
* **$lte: Value is less than or equal to another value**
* **$in: Value is matched within an array**

Logical

**The following operators can logically compare multiple queries.**

* **$and: Returns documents where both queries match**
* **$or: Returns documents where either query matches**
* **$nor: Returns documents where both queries fail to match**
* **$not: Returns documents where the query does not match**

Evaluation

**The following operators assist in evaluating documents.**

* **$regex: Allows the use of regular expressions when evaluating field values**
* **$text: Performs a text search**
* **$where: Uses a JavaScript expression to match documents**

MongoDB Update Operators

**There are many update operators that can be used during document updates.**

Fields

**The following operators can be used to update fields:**

* **$currentDate: Sets the field value to the current date**
* **$inc: Increments the field value**
* **$rename: Renames the field**
* **$set: Sets the value of a field**
* **$unset: Removes the field from the document**

Array

**The following operators assist with updating arrays.**

* **$addToSet: Adds distinct elements to an array**
* **$pop: Removes the first or last element of an array**
* **$pull: Removes all elements from an array that match the query**
* **$push: Adds an element to an array**

=============================================

The find() Method

**To query data from MongoDB collection, you need to use MongoDB's find() method.**

**>db.COLLECTION\_NAME.find()  // sele \* from table**

The pretty() Method

**To display the results in a formatted way, you can use pretty() method.**

Syntax

**>db.COLLECTION\_NAME.find().pretty()**

**Find By column**

**db.mycol.find({<key1>:<value1>}})**

AND in MongoDB

**>db.mycol.find({ $and: [ {<key1>:<value1>}, { <key2>:<value2>} ] })**

**> db.mycol.find({$and:[{"by":"tutorials point"},{"title": "MongoDB Overview"}]}).pretty()**

OR in MongoDB

**>db.mycol.find({$or:[{"by":"tutorials point"},{"title": "MongoDB Overview"}]}).pretty()**

Using AND and OR Together

**db.mycol.find({"likes": {$gt:10}, $or: [{"by": "tutorials point"},**

**{"title": "MongoDB Overview"}]}).pretty()**

NOR in MongoDB

**db.empDetails.find(**

**{**

**$nor:[**

**40**

**{"First\_Name": "Radhika"},**

**{"Last\_Name": "Christopher"}**

**]**

**}**

**).pretty()**

NOT in MongoDB

**db.empDetails.find( { "Age": { $not: { $gt: "25" } } } )**

**{**

**"\_id" : ObjectId("5dd6636870fb13eec3963bf7"),**

**"First\_Name" : "Fathima",**

**"Last\_Name" : "Sheik",**

**"Age" : "24",**

**"e\_mail" : "Fathima\_Sheik.123@gmail.com",**

**"phone" : "9000054321"**

**}**

MongoDB Update() Method

Syntax

**The basic syntax of update() method is as follows −**

**>db.COLLECTION\_NAME.update(SELECTION\_CRITERIA, UPDATED\_DATA)**

**>db.mycol.update({'title':'MongoDB Overview'},{$set:{'title':'New MongoDB Tutorial'}})**

**>db.mycol.updateOne({'title':'MongoDB Overview'},{$set:{'title':'New MongoDB Tutorial'}})**

**==============================================**

The remove() Method

**MongoDB's remove() method is used to remove a document from the collection. remove() method accepts two parameters. One is deletion criteria and second is justOne flag.**

* **deletion criteria − (Optional) deletion criteria according to documents will be removed.**
* **justOne − (Optional) if set to true or 1, then remove only one document.**

**Basic syntax of remove() method is as follows −**

**>db.COLLECTION\_NAME.remove(DELLETION\_CRITTERIA)**

**>db.mycol.remove({title:'MongoDB Overview'})**

**Or**

**>db.mycol.deleteOne({title:'MongoDB Overview'})**

Remove Only One

**If there are multiple records and you want to delete only the first record, then set justOne parameter in remove() method.**

**>db.COLLECTION\_NAME.remove(DELETION\_CRITERIA,1)**

Remove All Documents

**> db.mycol.remove({})**

=====================================================================================