At First After Installation Start The MongoDB Service From Services.msc

Then Type The Following Command

C:\Program Files\MongoDB\Server\5.0\bin>mongo

show dbs

use <database\_name>

use mydatabase

db.users.insert({ name: "John Doe", email: "john.doe@example.com" })

show collections

db.users.find()

db.users.find({ name: "John Doe" })

db.posts.insertOne({

title: "Post Title 1",

body: "Body of post.",

category: "News",

likes: 1,

tags: ["news", "events"],

date: Date()

})

db.posts.insertMany([

{

title: "Post Title 2",

body: "Body of post.",

category: "Event",

likes: 2,

tags: ["news", "events"],

date: Date()

},

{

title: "Post Title 3",

body: "Body of post.",

category: "Technology",

likes: 3,

tags: ["news", "events"],

date: Date()

},

{

title: "Post Title 4",

body: "Body of post.",

category: "Event",

likes: 4,

tags: ["news", "events"],

date: Date()

}

])

db.posts.find()

db.posts.findOne()

db.posts.find( {category: "News"} )

this time, let's exclude the \_id field.

db.posts.find({}, {\_id: 0, title: 1, date: 1})

db.posts.find( { title: "Post Title 1" } )

db.posts.updateOne( { title: "Post Title 1" }, { $set: { likes: 2 } } )

Update the document, but if not found insert it:

db.posts.updateOne(

{ title: "Post Title 5" },

{

$set:

{

title: "Post Title 5",

body: "Body of post.",

category: "Event",

likes: 5,

tags: ["news", "events"],

date: Date()

}

},

{ upsert: true }

)

The updateMany() method will update all documents that match the provided query.

Update(Increase) The likes on all documents by 1. For this we will use the $inc (increment) operator:

db.posts.updateMany({}, { $inc: { likes: 1 } })

db.posts.deleteOne({ title: "Post Title 5" })

db.posts.deleteMany({ category: "Technology" })

> db.emp.find({},{\_id:0,empNo:1,empName:1,age:1})

{ "empNo" : 1, "empName" : "Subhabrata Sinha", "age" : 46 }

{ "empNo" : 2, "empName" : "Deb Bhowmick", "age" : 36 }

{ "empNo" : 3, "empName" : "Partha Chatterjee", "age" : 26 }

> db.emp.find({},{\_id:0,empNo:1,age:1})

{ "empNo" : 1, "age" : 46 }

{ "empNo" : 2, "age" : 36 }

{ "empNo" : 3, "age" : 26 }

> show dbs

admin 0.000GB

config 0.000GB

local 0.000GB

mydatabase 0.000GB

> show collections

emp

> db.emp.find({age:{$gt:30}})

{ "\_id" : ObjectId("66c855b346698d830d7fcaeb"), "empNo" : 1, "empName" : "Subhabrata Sinha", "age" : 46 }

{ "\_id" : ObjectId("66c855cf46698d830d7fcaec"), "empNo" : 2, "empName" : "Deb Bhowmick", "age" : 36 }

> db.emp.find({age:{$gt:20}})

{ "\_id" : ObjectId("66c855b346698d830d7fcaeb"), "empNo" : 1, "empName" : "Subhabrata Sinha", "age" : 46 }

{ "\_id" : ObjectId("66c855cf46698d830d7fcaec"), "empNo" : 2, "empName" : "Deb Bhowmick", "age" : 36 }

{ "\_id" : ObjectId("66c855e846698d830d7fcaed"), "empNo" : 3, "empName" : "Partha Chatterjee", "age" : 26 }

> db.emp.find({},{\_id:0,empNo:1,empName:1,age:1},{age:{$gt:20}})

{ "empNo" : 1, "empName" : "Subhabrata Sinha", "age" : 46 }

{ "empNo" : 2, "empName" : "Deb Bhowmick", "age" : 36 }

{ "empNo" : 3, "empName" : "Partha Chatterjee", "age" : 26 }

>

Dropping A table

db.mycollection1.drop()

Droppinmg A Database

use databaseName

db.dropDatabase()

In MongoDB, you can perform **read** and **write operations** directly using MongoDB shell commands or through drivers in various programming languages. Below are examples of common **read operations** and **write operations** using **MongoDB shell commands**.

**1. Read Operations in MongoDB**

**Common Commands:**

1. **find()**: Retrieves multiple documents.
2. **findOne()**: Retrieves a single document.
3. **countDocuments()**: Counts documents matching a query.
4. **Projection**: Specifies the fields to include or exclude in the output.

**Examples:**

bash

CopyEdit

# Switch to the database

use myDatabase;

# 1. Retrieve all documents

db.myCollection.find();

# 2. Retrieve documents with a filter

db.myCollection.find({ age: { $gt: 25 } });

# 3. Retrieve a single document

db.myCollection.findOne({ name: "Alice" });

# 4. Retrieve documents with a projection (include only name and age)

db.myCollection.find({ age: { $gt: 25 } }, {\_id:0, name: 1, age: 1 });

# 5. Count the number of documents matching a query

db.myCollection.countDocuments({ age: { $gt: 25 } });

# 6. Sort documents (e.g., by age in descending order)

db.myCollection.find().sort({ age: -1 });

db.myCollection.find({},{\_id:0,name:1,age:1}).sort({ age: -1 });

# 7. Limit the number of documents returned

db.myCollection.find().limit(5);

# 8. Skip a number of documents (useful for pagination)

db.myCollection.find().skip(10).limit(5);

**2. Write Operations in MongoDB**

**Common Commands:**

1. **insertOne()**: Inserts a single document.
2. **insertMany()**: Inserts multiple documents.
3. **updateOne()**: Updates a single document.
4. **updateMany()**: Updates multiple documents.
5. **deleteOne()**: Deletes a single document.
6. **deleteMany()**: Deletes multiple documents.

**Examples:**

bash

CopyEdit

# Switch to the database

use myDatabase;

# 1. Insert a single document

db.myCollection.insertOne({ name: "Alice", age: 30, city: "New York" });

# 2. Insert multiple documents

db.myCollection.insertMany([

{ name: "Bob", age: 25, city: "Los Angeles" },

{ name: "Charlie", age: 35, city: "Chicago" }

]);

# 3. Update a single document (update Alice's age to 31)

db.myCollection.updateOne(

{ name: "Alice" }, # Filter

{ $set: { age: 31 } } # Update

);

# 4. Update multiple documents (set status for all users under age 30)

db.myCollection.updateMany(

{ age: { $lt: 30 } }, # Filter

{ $set: { status: "young" } } # Update

);

# 5. Delete a single document (delete Bob)

db.myCollection.deleteOne({ name: "Bob" });

# 6. Delete multiple documents (delete all documents with age > 40)

db.myCollection.deleteMany({ age: { $gt: 40 } });