MSC CS – I

Name: Ruchita Chipkar

Roll No: 34

Business Intelligence and Big Data Analytics

Mini Project

(Implementation of NOSQL database – MognoDB**)**

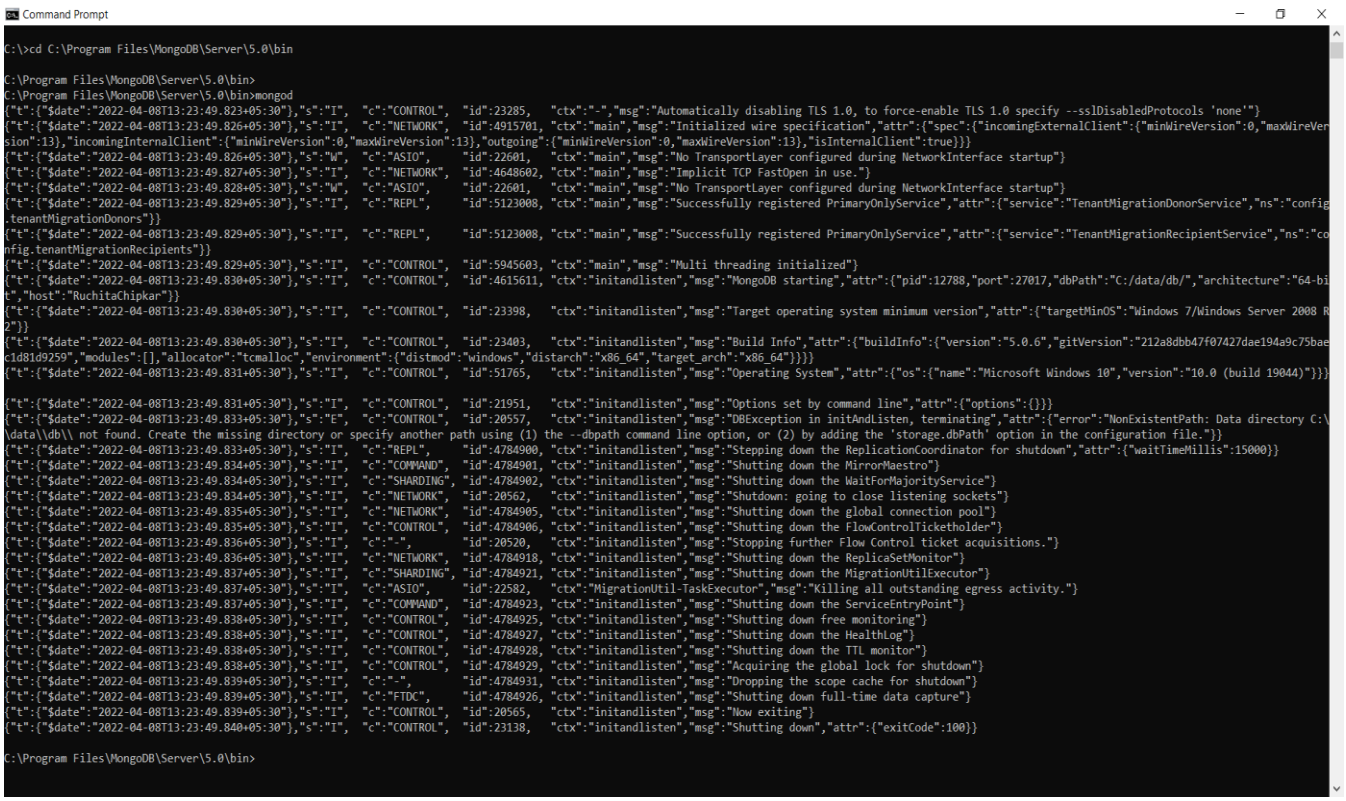
Aim: Executing CRUD operations in MongoDB shell.

Steps:

1. Open the command prompt and go to the folder location where MongoDB is installed.

2. Execute the mongod command.

   *mongod command: The main purpose of mongod is to **manage all the MongoDB server tasks**. For instance, accepting requests, responding to client, and memory management. mongo is a command line shell that can interact with the client (for example, system administrators and developers).*



3. Execute mongo command.
   (To work in MongoDB, we use mongo command)

```
Command Prompt - mongo
C:\Program Files\MongoDB\Server\5.0\bin>mongo
MongoDB shell version v5.0.6
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("0acb540d-f27a-400c-bc30-7ab414a36c57") }
MongoDB server version: 5.0.6
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
        https://community.mongodb.com
---
The server generated these startup warnings when booting:
        2022-04-07T20:49:31.600+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricte
d
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
>
```

4. Check for any existing databases using ***Show dbs*** command.

```
Select Command Prompt - mongo
> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

5. So, we do not have our own existing database, hence we'll <u>create a new database</u>.

```
Command Prompt - mongo
>
> use emp
switched to db emp
> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

6. We've created a database named emp here, but it is not displayed because it's empty, so we need to create a collection first inside this database. To insert a document into the collection json format is followed.

```
Command Prompt - mongo
> show collections
> db.employee.insertOne(Name: "Ruchita", RollNo: 34)
uncaught exception: SyntaxError: missing ) after argument list :
@(shell):1:26
>
>
> db.employee.insertOne({Name: "Ruchita", RollNo: 34})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("624fee951c6a47edd8954c86")
}
>
```

7. Here, we've created a collection in the emp database named employee and added a document of one employee. So now if we check the databases on the system, we can see the emp database.

```
Select Command Prompt - mongo
>
> show dbs;
admin   0.000GB
config  0.000GB
emp     0.000GB
local   0.000GB
>
```

8. Now, to check if the document is added in the collection we run:

```
Command Prompt - mongo
>
> show collections
employee
> db.employee.find()
{ "_id" : ObjectId("624fee951c6a47edd8954c86"), "Name" : "Ruchita", "RollNo" : 34 }
>
```

9. So, the document we inserted earlier is shown here. If we want it in a more readable format, we can use the pretty() function.

```
Command Prompt - mongo
>
> db.employee.find().pretty()
{
        "_id" : ObjectId("624fee951c6a47edd8954c86"),
        "Name" : "Ruchita",
        "RollNo" : 34
}
>
```

10. We know how to create a database. Now let's see how to delete/drop a database.

    Here, I've already created another sample database "demodb" with a document in it.

```
Command Prompt - mongo
> use demodb
switched to db demodb
> db.test.insertOne({Name: "abc"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("62500dbd1c6a47edd8954c87")
}
>
>
>
```

```
Command Prompt - mongo
>
> show dbs;
admin   0.000GB
config  0.000GB
demodb  0.000GB
emp     0.000GB
local   0.000GB
>
```

To drop a single collection, you can do as follows:

```
Command Prompt - mongo
>
> db.test.drop()
true
>
```

To drop the whole database, you can do as follows:

```
Command Prompt - mongo
> use demodb
switched to db demodb
> db.dropDatabase()
{ "ok" : 1 }
>
>
> show dbs;
admin   0.000GB
config  0.000GB
emp     0.000GB
local   0.000GB
>
```

11. The basic CRUD operations include Create, Read, Update & Delete.
12. The Create commands are of two types “insertOne(data, options)” & “insertMany([data], options)”.
13. The Read commands are of two types “find(filter, options)” & “findOne(filter, options)”.
14. The Update command are of three types “updateOne(filter, data, options)”; “updateMany(filter, data, options)” & “replaceOne(filter, data, options)”.
15. The Delete command are of two types “deleteOne(filter, options)” & “deleteMany(filter, options)”.
16. Executing the insertOne and insertMany commands:

insertOne:

```
Command Prompt - mongo
>
> use school
switched to db school
> db.student.insertOne({Name: "Ruchita", RollNo: 34, Address: "Mumbai"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("6250156b1c6a47edd8954c88")
}
>
>
```

insertMany:

```
Command Prompt - mongo
>
>
> db.student.insertMany([{Name: "Siddhant", RollNo: 06, Address: "Pune"},{Name: "Jay", RollNo: 15,
Address: "Mumbai"}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("62501c071c6a47edd8954c89"),
                ObjectId("62501c071c6a47edd8954c8a")
        ]
}
>
```

17. Here check the records/document we have updated in the collection student.

```
Command Prompt - mongo
>
> use school
switched to db school
> show collections
student
>
> db.student.find().pretty()
{
        "_id" : ObjectId("6250156b1c6a47edd8954c88"),
        "Name" : "Ruchita",
        "RollNo" : 34,
        "Address" : "Mumbai"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c89"),
        "Name" : "Siddhant",
        "RollNo" : 6,
        "Address" : "Pune"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c8a"),
        "Name" : "Jay",
        "RollNo" : 15,
        "Address" : "Mumbai"
}
>
```

Here, we've successfully executed the insertOne and insertMany commands and also Read the data in the Document.

18. Now let's try updating the RollNo of Siddhant to 10 in the document and check the record is updated or not.

```
Command Prompt - mongo
>
> db.student.updateOne({Name: "Siddhant"}, {$set:{RollNo: 10}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
>
> db.student.find().pretty()
{
        "_id" : ObjectId("6250156b1c6a47edd8954c88"),
        "Name" : "Ruchita",
        "RollNo" : 34,
        "Address" : "Mumbai"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c89"),
        "Name" : "Siddhant",
        "RollNo" : 10,
        "Address" : "Pune"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c8a"),
        "Name" : "Jay",
        "RollNo" : 15,
        "Address" : "Mumbai"
}
>
```

19. Now let's try updateMany command.

Select Command Prompt - mongo

```
>
>
> db.student.updateMany({},{$set: {BloodGroup: "unknown"}})
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 3 }
>
```

Keeping the first parameter blank means updating all the entries.

Command Prompt - mongo

```
>
> db.student.find().pretty()
{
        "_id" : ObjectId("6250156b1c6a47edd8954c88"),
        "Name" : "Ruchita",
        "RollNo" : 34,
        "Address" : "Mumbai",
        "BloodGroup" : "unknown"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c89"),
        "Name" : "Siddhant",
        "RollNo" : 10,
        "Address" : "Pune",
        "BloodGroup" : "unknown"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c8a"),
        "Name" : "Jay",
        "RollNo" : 15,
        "Address" : "Mumbai",
        "BloodGroup" : "unknown"
}
>
```

20. Now let's change the Blood Group of one student.

Command Prompt - mongo

```
>
> db.student.updateOne({Name: "Ruchita"}, {$set: {"BloodGroup": "O+"}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
>
> db.student.find().pretty()
{
        "_id" : ObjectId("6250156b1c6a47edd8954c88"),
        "Name" : "Ruchita",
        "RollNo" : 34,
        "Address" : "Mumbai",
        "BloodGroup" : "O+"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c89"),
        "Name" : "Siddhant",
        "RollNo" : 10,
        "Address" : "Pune",
        "BloodGroup" : "unknown"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c8a"),
        "Name" : "Jay",
        "RollNo" : 15,
        "Address" : "Mumbai",
        "BloodGroup" : "unknown"
}
>
```

21. Now using the Find command to find an entry with a particular tag.

```
Command Prompt - mongo
>
> db.student.find({RollNo: 10}).pretty()
{
        "_id" : ObjectId("62501c071c6a47edd8954c89"),
        "Name" : "Siddhant",
        "RollNo" : 10,
        "Address" : "Pune",
        "BloodGroup" : "unknown"
}
>
```

22. So now let's delete an entry from the student using deleteOne() where BloodGroup is O+.

```
Command Prompt - mongo
>
> db.student.deleteOne({BloodGroup: "O+"})
{ "acknowledged" : true, "deletedCount" : 1 }
>
> db.student.find().pretty()
{
        "_id" : ObjectId("62501c071c6a47edd8954c89"),
        "Name" : "Siddhant",
        "RollNo" : 10,
        "Address" : "Pune",
        "BloodGroup" : "unknown"
}
{
        "_id" : ObjectId("62501c071c6a47edd8954c8a"),
        "Name" : "Jay",
        "RollNo" : 15,
        "Address" : "Mumbai",
        "BloodGroup" : "unknown"
}
>
```

23. Now deleting users with deleteMany() operations where BloodGroup is unknown.

```
Command Prompt - mongo
>
> db.student.deleteMany({BloodGroup: "unknown"})
{ "acknowledged" : true, "deletedCount" : 2 }
>
> db.student.find().pretty()
>
>
```

All records are deleted and hence we now have an empty collection.

This is all about the CRUD operations in MongoDB.