

Name : Yash Kesharwani

Roll No : 3

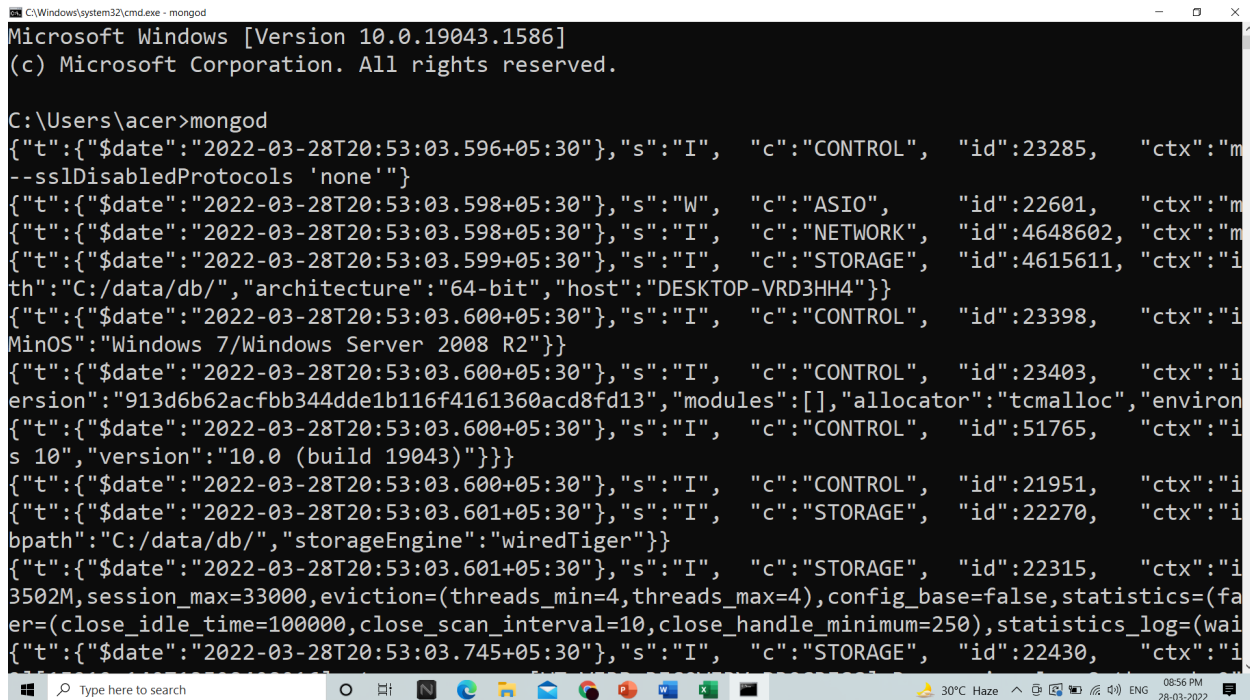
Class : MSC CS part 1

# BIBD MINI PROJECT

**Aim: Executing CRUD operations in MongoDB shell.**

**Steps :-**

**Start The MongoDB server**



```
C:\Windows\system32\cmd.exe - mongod
Microsoft Windows [Version 10.0.19043.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\acer>mongod
{"t":{"$date":"2022-03-28T20:53:03.596+05:30"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx":"m
--sslDisabledProtocols 'none'"}
{"t":{"$date":"2022-03-28T20:53:03.598+05:30"},"s":"W",  "c":"ASIO",    "id":22601,   "ctx":"m
{"t":{"$date":"2022-03-28T20:53:03.598+05:30"},"s":"I",  "c":"NETWORK",  "id":4648602, "ctx":"m
{"t":{"$date":"2022-03-28T20:53:03.599+05:30"},"s":"I",  "c":"STORAGE",  "id":4615611, "ctx":"i
th":"C:/data/db/", "architecture":"64-bit", "host":"DESKTOP-VRD3HH4"}}
{"t":{"$date":"2022-03-28T20:53:03.600+05:30"},"s":"I",  "c":"CONTROL",  "id":23398,   "ctx":"i
MinOS:"Windows 7/Windows Server 2008 R2"}}
{"t":{"$date":"2022-03-28T20:53:03.600+05:30"},"s":"I",  "c":"CONTROL",  "id":23403,   "ctx":"i
ersion":"913d6b62acfb344dde1b116f4161360acd8fd13", "modules":[], "allocator":"tcmalloc", "environ
{"t":{"$date":"2022-03-28T20:53:03.600+05:30"},"s":"I",  "c":"CONTROL",  "id":51765,   "ctx":"i
s 10", "version":"10.0 (build 19043)"}
{"t":{"$date":"2022-03-28T20:53:03.600+05:30"},"s":"I",  "c":"CONTROL",  "id":21951,   "ctx":"i
{"t":{"$date":"2022-03-28T20:53:03.601+05:30"},"s":"I",  "c":"STORAGE",  "id":22270,   "ctx":"i
bpath":"C:/data/db/", "storageEngine":"wiredTiger"}}
{"t":{"$date":"2022-03-28T20:53:03.601+05:30"},"s":"I",  "c":"STORAGE",  "id":22315,   "ctx":"i
3502M, session_max=33000, eviction=(threads_min=4, threads_max=4), config_base=false, statistics=(fa
er=(close_idle_time=100000, close_scan_interval=10, close_handle_minimum=250), statistics_log=(wai
{"t":{"$date":"2022-03-28T20:53:03.745+05:30"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"i
```

**Start The MongoDB shell.**

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

```
Command Prompt - mongo
Microsoft Windows [Version 10.0.19043.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\acer>mongo
MongoDB shell version v4.4.3
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("ae41d2cd-6740-44f3-87e0-0599076a9c61") }
MongoDB server version: 4.4.3
---
The server generated these startup warnings when booting:
  2022-03-20T20:31:01.643+05:30: Access control is not enabled for the database. Read and write access to data and configuration is not allowed.
---
  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()
---
>
```

**Check for any existing databases.**

```
---
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
oldSudents 0.000GB
test       0.000GB
yash       0.000GB
>
```

**We have created a new database**

```
> use customer
switched to db customer
> db
customer
>
```

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

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

```
> db.customer.insert({Name:'yash kesharwani',orderId : 100, productName : 'mobile', Quantity : 5})
WriteResult({ "nInserted" : 1 })
>
```

**Here, we've created a collection in the school database named students and added a document of one customer. So now if we check the databases on the system we can see the customer database.**

```
> show dbs
admin          0.000GB
config         0.000GB
customer       0.000GB
local          0.000GB
oldSudents    0.000GB
test           0.000GB
yash           0.000GB
>
```

**Now, to check if the document is added in the customer collection**

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

```
switched to db customer  
> db.customer.find().pretty()  
{  
  "_id" : ObjectId("6241d836eb7e5dd1ba5a66d4"),  
  "Name" : "yash kesharwani",  
  "orderId" : 100,  
  "productName" : "mobile",  
  "Quantity" : 5  
}  
>
```

**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 "test" with document in it.**

```
> show dbs
admin      0.000GB
config     0.000GB
customer   0.000GB
local      0.000GB
oldSudents 0.000GB
test       0.000GB
> use oldStudents
switched to db oldStudents
> db.dropDatabase()
{ "ok" : 1 }
> use test
switched to db test
> db.customer.insert({Name:'test Document'})
WriteResult({ "nInserted" : 1 })
> show dbs
admin      0.000GB
config     0.000GB
customer   0.000GB
local      0.000GB
oldSudents 0.000GB
test       0.000GB
> db.dropDatabase()
{ "dropped" : "test", "ok" : 1 }
> show dbs
admin      0.000GB
config     0.000GB
customer   0.000GB
local      0.000GB
oldSudents 0.000GB
>
```

**The basic CRUD operations include Create, Read, Update & Delete.**

**1. The Create commands are of two types “insertOne(data, options)” & “insertMany([data], options)”.**

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

**2. The Read command are of two types “find(filter, options)” & “findOne(filter, options)”.**

**3. The Update command are of three types “updateOne(filter, data, options)” ; “updateMany(filter, data, options)” & “replaceOne(filter, data, options)”.**

**4. The Delete command are of two types “deleteOne(filter, options)” & “deleteMany(filter, options)”.**

**Executing the insertOne and insertMany commands:**

```
> use employee
switched to db employee
> db.employee.insertOne({name : 'yash kesharwani', jobRole : 'web developer', address : 'ghatkopar', languages : ['python','java']})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("6241dfbdeb7e5dd1ba5a66d9")
}
> db.employee.insertMany([({name : 'yash walke', jobRole : 'developer', address : 'ghatkopar', languages : ['python','java']},{name:'anurag rai',jobRole : 'web dev',address:
}))
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("6241dfc7eb7e5dd1ba5a66da"),
    ObjectId("6241dfc7eb7e5dd1ba5a66db")
  ]
}
```

**Let us now check the database.**

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

```
> show dbs
admin      0.000GB
config     0.000GB
customer   0.000GB
employee    0.000GB
local      0.000GB
oldSudents 0.000GB
>
```

**Here check the records/document we have updated in the collection employee**

```
> db.employee.find().pretty()
{
  "_id" : ObjectId("6241dfbdeb7e5dd1ba5a66d9"),
  "name" : "yash kesharwani",
  "jobRole" : "web developer",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ]
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66da"),
  "name" : "yash walke",
  "jobRole" : "devloper",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ]
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66db"),
  "name" : "anurag rai",
  "jobRole" : "web dev",
  "address" : "bhiwandi",
  "languages" : [
    "python",
    "java"
  ]
}
>
```

- Here, we've successfully executed the insertOne and insertMany commands and also Read the data in the Document.
- Now let's try updating the location of anurag to Ghatkopar in the document



Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

```
> db.employee.updateOne({_id : ObjectId("6241dfc7eb7e5dd1ba5a66db")}, {$set: {"address": "london"}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
>
```

**Check if the value is updated**

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

```
{ acknowledged : true, matchedCount : 1, modifiedCount : 0 }
> db.employee.find().pretty()
{
  "_id" : ObjectId("6241dfbdeb7e5dd1ba5a66d9"),
  "name" : "yash kesharwani",
  "jobRole" : "web devloper",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ]
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66da"),
  "name" : "yash walke",
  "jobRole" : "devloper",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ]
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66db"),
  "name" : "anurag rai",
  "jobRole" : "web dev",
  "address" : "london",
  "languages" : [
    "python",
    "java"
  ]
}
```

**Now lets try updateMany command**

```
> db.employee.updateMany({}, {$set: {"experience-in-year": 1}})
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 3 }
> db.employee.find().pretty()
```

**Keeping the first parameter blank means updating all the entries.**

```
> db.employee.updateMany({}, {$set: {"experience-in-year": 1}})
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 3 }
> db.employee.find().pretty()
{
  "_id" : ObjectId("6241dfbdeb7e5dd1ba5a66d9"),
  "name" : "yash kesharwani",
  "jobRole" : "web developer",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ],
  "experience-in-year" : 1
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66da"),
  "name" : "yash walke",
  "jobRole" : "developer",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ],
  "experience-in-year" : 1
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66db"),
  "name" : "anurag rai",
  "jobRole" : "web dev",
  "address" : "london",
  "languages" : [
    "python",
    "java"
  ],
  "experience-in-year" : 1
}
```

**Now using the Find command to find an entry of a particular developer.**

```
> db.employee.find({jobRole : "web dev"}).pretty()
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66db"),
  "name" : "anurag rai",
  "jobRole" : "web dev",
  "address" : "london",
  "languages" : [
    "python",
    "java"
  ],
  "experience-in-year" : 1
}
```

**.So now let's delete an entry from employee using deleteOne() where name is yash kesharwani.**

```
> db.employee.deleteOne({name:"yash kesharwani"})
{ "acknowledged" : true, "deletedCount" : 1 }
```

**Checking database**

Name : Yash Kesharwani

Roll No : 3

Class : MSC CS part 1

```
> db.employee.find().pretty()
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66da"),
  "name" : "yash walke",
  "jobRole" : "devloper",
  "address" : "ghatkopar",
  "languages" : [
    "python",
    "java"
  ],
  "experience-in-year" : 1
}
{
  "_id" : ObjectId("6241dfc7eb7e5dd1ba5a66db"),
  "name" : "anurag rai",
  "jobRole" : "web dev",
  "address" : "london",
  "languages" : [
    "python",
    "java"
  ],
  "experience-in-year" : 1
}
>
```