

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Subject: Big Data Technology

Practical_6: Handling Unstructured

Open the command prompt and run the cd command to change directory and run the command "mongod" to start the server and run the command "mongo" to work mongoDB.

```
Command Prompt - mongod
ating System", "attr": {"os": {"name": "Microsoft Windows 10", "version": "10.0 (build 19044)"}}
{"t":{"$date":"2022-02-25T08:13:30.392+05:30"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options
set by command line", "attr":{"options":{}}}
{"t":{"$date":"2022-02-25T08:13:30.409+05:30"},"s":"E", "c":"CONTROL", "id":20557, "ctx":"initandlisten","msg":"DBEx
ception in initAndListen, terminating", "attr":{"error":"NonExistentPath: Data directory C:\\data\\db\\ not found. Create
the missing directory or specify another path using (1) the --dbpath command line option, or (2) by adding the 'storage
.dbPath' option in the configuration file."}}
{"t":{"$date":"2022-02-25T08:13:30.409+05:30"},"s":"I", "c":"REPL", "id":4784900, "ctx":"initandlisten","msg":"Step
ping down the ReplicationCoordinator for shutdown", "attr":{"waitTimeMillis":15000}}
{"t":{"$date":"2022-02-25T08:13:30.417+05:30"},"s":"I", "c":"COMMAND", "id":4784901, "ctx":"initandlisten","msg":"Shut
ting down the MirrorMaestro"}
{"t":{"$date":"2022-02-25T08:13:30.417+05:30"},"s":"I", "c":"SHARDING", "id":4784902, "ctx":"initandlisten","msg":"Shut
ting down the WaitForMajorityService"}
{"t":{"$date":"2022-02-25T08:13:30.418+05:30"},"s":"I", "c":"NETWORK", "id":20562, "ctx":"initandlisten","msg":"Shut
down: going to close listening sockets"}
{"t":{"$date":"2022-02-25T08:13:30.418+05:30"},"s":"I", "c":"NETWORK", "id":4784905, "ctx":"initandlisten","msg":"Shut
ting down the global connection pool"}
{"t":{"$date":"2022-02-25T08:13:30.419+05:30"},"s":"I", "c":"CONTROL", "id":4784906, "ctx":"initandlisten","msg":"Shut
ting down the FlowControlTicketHolder"}
{"t":{"$date":"2022-02-25T08:13:30.419+05:30"},"s":"I", "c":"-", "id":20520, "ctx":"initandlisten","msg":"Stop
ping further Flow Control ticket acquisitions."}
{"t":{"$date":"2022-02-25T08:13:30.423+05:30"},"s":"I", "c":"NETWORK", "id":4784918, "ctx":"initandlisten","msg":"Shut
ting down the ReplicaSetMonitor"}
{"t":{"$date":"2022-02-25T08:13:30.423+05:30"},"s":"I", "c":"SHARDING", "id":4784921, "ctx":"initandlisten","msg":"Shut
ting down the MigrationUtilExecutor"}
{"t":{"$date":"2022-02-25T08:13:30.424+05:30"},"s":"I", "c":"ASIO", "id":22582, "ctx":"MigrationUtil-TaskExecutor
", "msg":"Killing all outstanding egress activity."}
{"t":{"$date":"2022-02-25T08:13:30.432+05:30"},"s":"I", "c":"COMMAND", "id":4784923, "ctx":"initandlisten","msg":"Shut
ting down the ServiceEntryPoint"}
{"t":{"$date":"2022-02-25T08:13:30.433+05:30"},"s":"I", "c":"CONTROL", "id":4784925, "ctx":"initandlisten","msg":"Shut
ting down"}
Command Prompt - mongo
Exiting"}
{"t":{"$date":"2022-02-25T08:13:30.435+05:30"},"s":"I", "c":"CONTROL", "id":23138, "ctx":"initandlisten","msg":"Shut
ting down", "attr":{"exitCode":100}}

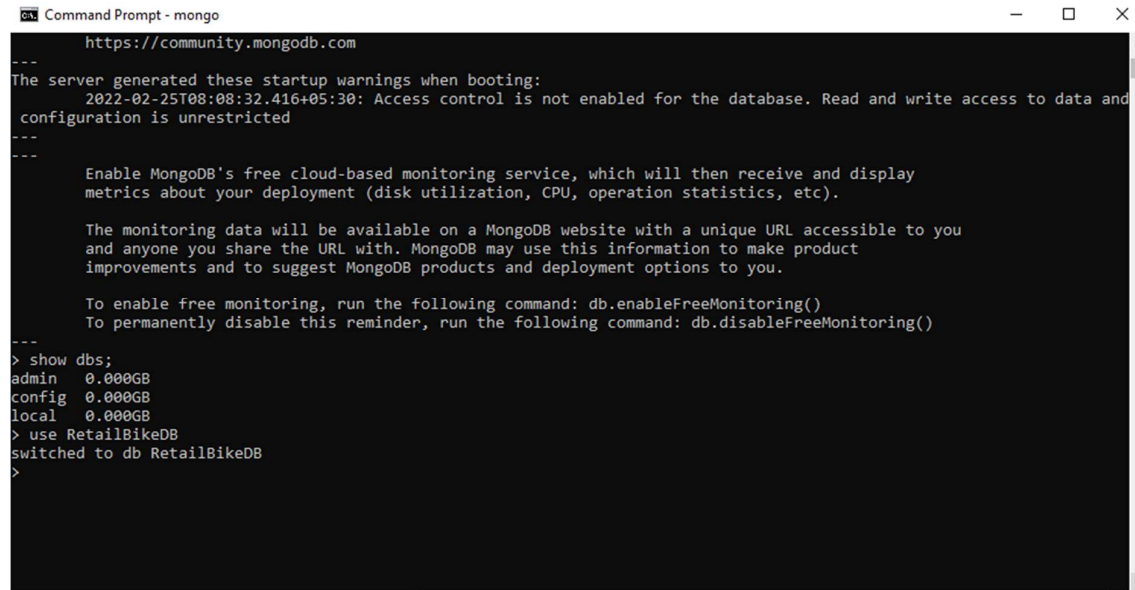
C:\Program Files\MongoDB\Server\5.0\bin>mongo
MongoDB shell version v5.0.6
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
implicit session: session { "id" : UUID("67bc152b-82ea-4650-a603-d4f6d24358e2") }
MongoDB server version: 5.0.6
=====
Warning: the "mongo" 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:
  2022-02-25T08:08:32.416+05:30: Access control is not enabled for the database. Read and write access to data and
configuration is unrestricted
---
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Show list of databases

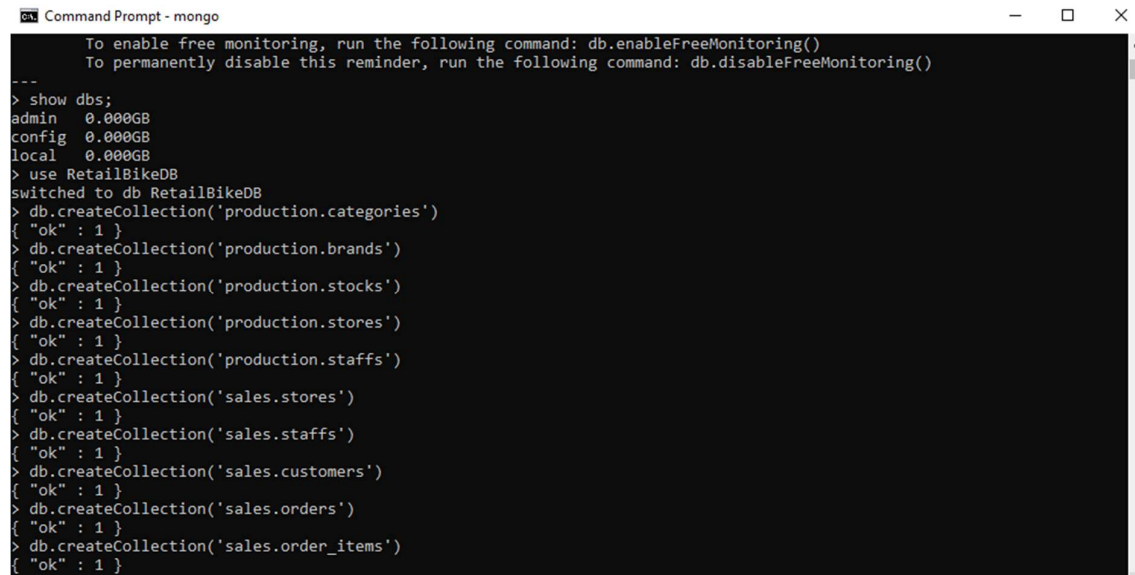


```
Command Prompt - mongo
https://community.mongodb.com
---
The server generated these startup warnings when booting:
  2022-02-25T08:08:32.416+05:30: Access control is not enabled for the database. Read and write access to data and
configuration is unrestricted
---
  Enable MongoDB's free cloud-based 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()
---
> show dbs;
admin      0.000GB
config     0.000GB
local      0.000GB
> use RetailBikeDB
switched to db RetailBikeDB
>
```

Creation of collection can be done using db.createCollection(name)



```
Command Prompt - mongo
  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs;
admin      0.000GB
config     0.000GB
local      0.000GB
> use RetailBikeDB
switched to db RetailBikeDB
> db.createCollection('production.categories')
{ "ok" : 1 }
> db.createCollection('production.brands')
{ "ok" : 1 }
> db.createCollection('production.stocks')
{ "ok" : 1 }
> db.createCollection('production.stores')
{ "ok" : 1 }
> db.createCollection('production.staffs')
{ "ok" : 1 }
> db.createCollection('sales.stores')
{ "ok" : 1 }
> db.createCollection('sales.staffs')
{ "ok" : 1 }
> db.createCollection('sales.customers')
{ "ok" : 1 }
> db.createCollection('sales.orders')
{ "ok" : 1 }
> db.createCollection('sales.order_items')
{ "ok" : 1 }
```

We can show list of collection using “Show collections” commands in MongoDB.

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

```
> show collections
production.brands
production.categories
production.staffs
production.stocks
production.stores
sales.customers
sales.order_items
sales.orders
sales.staffs
sales.stores
>
```

1. production.categories

Select Command Prompt - mongo

```
> db.production.categories.insertMany(
... [
... {
...   'category_id':1,
...   "category_name": "Road Bike"
... },
... {
...   "category_id": 2,
...   "category_name": "Mountain Bike"
... },
... {
...   "category_id": 3,
...   "category_name": "Hybrid Bike"
... },
... {
...   "category_id": 4,
...   "category_name": "Folding Bike"
... },
... {
...   "category_id": 5,
...   "category_name": "Touring Bike"
... },
... {
...   "category_id": 6,
...   "category_name": "Cruiser Bike"
... },
... {
...   "category_id": 7,
...   "category_name": "Women Bike"
... }
... ]
... )
```

```
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184a406080c37f46a3ad74"),
    ObjectId("62184a406080c37f46a3ad75"),
    ObjectId("62184a406080c37f46a3ad76"),
    ObjectId("62184a406080c37f46a3ad77"),
    ObjectId("62184a406080c37f46a3ad78"),
    ObjectId("62184a406080c37f46a3ad79"),
    ObjectId("62184a406080c37f46a3ad7a")
  ]
}
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

2. production.products

```
Command Prompt - mongo
> db.production.products.insertMany(
... [
... {
...   "product_id": 1,
...   "product_name": "Honda Superfast",
...   "brand_id": 1,
...   "category_id": 1,
...   "model_year": 1994,
...   "list_price": 25000
... },
... {
...   "product_id": 2,
...   "product_name": "6KU Bikes",
...   "brand_id": 2,
...   "category_id": 4,
...   "model_year": 2000,
...   "list_price": 30000
... },
... {
...   "product_id": 3,
...   "product_name": "Bianchi",
...   "brand_id": 3,
...   "category_id": 2,
...   "model_year": 2002,
...   "list_price": 30000
... },
... {
...   "product_id": 4,
...   "product_name": "BMC Hybrid Bike",
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Command Prompt - mongo

```
... {
...   "product_id": 4,
...   "product_name": "BMC Hybrid Bike",
...   "brand_id": 4,
...   "category_id": 3,
...   "model_year": 2009,
...
...   "list_price": 45000
... },
... {
...   "product_id": 5,
...   "product_name": "Huffy Women Bike",
...   "brand_id": 5,
...   "category_id": 7,
...   "model_year": 2019,
...   "list_price": 50000
... }
... ]
... )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184ab76080c37f46a3ad7b"),
    ObjectId("62184ab76080c37f46a3ad7c"),
    ObjectId("62184ab76080c37f46a3ad7d"),
    ObjectId("62184ab76080c37f46a3ad7e"),
    ObjectId("62184ab76080c37f46a3ad7f")
  ]
}
```


Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

3. production.stocks

Command Prompt - mongo

```
> db.production.stocks.insertMany(
... [
... {
...   "store_id": 1,
...   "product_id": 1,
...   "quantity": 15
... },
... {
...   "store_id": 2,
...   "product_id": 4,
...   "quantity": 20
... },
... {
...   "store_id": 3,
...   "product_id": 5,
...   "quantity": 30
... },
... {
...   "store_id": 1,
...   "product_id": 5,
...   "quantity": 100
... },
... {
...   "store_id": 2,
...   "product_id": 2,
...   "quantity": 4
... },
... {
...   "store_id": 3,
...   "product_id": 3,
```

Command Prompt - mongo

```
... {
...   "store_id": 3,
...   "product_id": 3,
...   "quantity": 9
... }
... ]
... )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184b446080c37f46a3ad80"),
    ObjectId("62184b446080c37f46a3ad81"),
    ObjectId("62184b446080c37f46a3ad82"),
    ObjectId("62184b446080c37f46a3ad83"),
    ObjectId("62184b446080c37f46a3ad84"),
    ObjectId("62184b446080c37f46a3ad85")
  ]
}
```

4. sales.customers

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Command Prompt - mongo

```
}  
> db.sales.customers.insertMany(  
... [  
... {  
...   "customer_id": "Cus001",  
...   "first_name": "Jay",  
...   "last_name": "Mehta",  
...   "phone": 1234567890,  
...   "email": "jay@gmail.com",  
...   "street": "T.P Road",  
...   "city": "Mumbai",  
...   "state": "Maharashtra",  
...   "zip_code": 400072  
... },  
... {  
...   "customer_id": "Cus002",  
...   "first_name": "Ruhi",  
...   "last_name": "Singh",  
...   "phone": 7894561230,  
...   "email": "ruhi@yahoo.com",  
...   "street": "M.G Chauk",  
...   "city": "Mumbai",  
...   "state": "Maharashtra",  
...   "zip_code": 400072  
... },  
... {  
...   "customer_id": "Cus003",  
...   "first_name": "Aria",  
...   "last_name": "Josh",  
...   "phone": 1245789630,  
... }
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Command Prompt - mongo

```
... "email": "aria.josh@gmail.com",
... "street": "JVM",
... "city": "Gandhi Nagar",
...
... "state": "Gujrat",
... "zip_code": 401235
... },
... {
... "customer_id": "Cus004",
... "first_name": "Mahi",
... "last_name": "Kaur",
... "phone": 4567890123,
... "email": "kaur.mahi@hotmail.com",
... "street": "J.V.L.R",
... "city": "Mumbai",
... "state": "Maharashtra",
... "zip_code": 400072
... },
... {
... "customer_id": "Cus005",
... "first_name": "Aditya",
... "last_name": "Yadav",
... "phone": 9638527410,
... "email": "aditya@gmail.com",
... "street": "Koliwada",
... "city": "Pune",
... "state": "Maharashtra",
... "zip_code": 300075
... }
... ]
```

```
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184c366080c37f46a3ad8c"),
    ObjectId("62184c366080c37f46a3ad8d"),
    ObjectId("62184c366080c37f46a3ad8e"),
    ObjectId("62184c366080c37f46a3ad8f"),
    ObjectId("62184c366080c37f46a3ad90")
  ]
}
```


Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

5. sales.order_items

CA Command Prompt - mongo

```
    ]
  }
> db.sales.order_items.insertMany(
... [
... {
...   "order_id": "ORD001",
...   "product_id": 1,
...   "quantity": 2,
...   "list_price": 50000
... },
... {
...   "order_id": "ORD002",
...   "product_id": 2,
...   "quantity": 3,
...   "list_price": 90000
... },
... {
...   "order_id": "ORD003",
...   "product_id": 3,
...   "quantity": 1,
...   "list_price": 30000
... },
... {
...   "order_id": "ORD004",
...   "product_id": 4,
...   "quantity": 8,
...   "list_price": 360000
... },
... {
...   "order_id": "ORD005",
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Command Prompt - mongo

```
... },
... {
...   "order_id": "ORD004",
...   "product_id": 4,
...   "quantity": 8,
...   "list_price": 360000
... },
... {
...   "order_id": "ORD005",
...   "product_id": 5,
...   "quantity": 2,
...   "list_price": 100000
... }
... ]
... )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184d186080c37f46a3ad91"),
    ObjectId("62184d186080c37f46a3ad92"),
    ObjectId("62184d186080c37f46a3ad93"),
    ObjectId("62184d186080c37f46a3ad94"),
    ObjectId("62184d186080c37f46a3ad95")
  ]
}
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

6. Sales.orders

Command Prompt - mongo

```
    ]
  }
> db.sales.orders.insertMany(
... [
... {
...   "order_id": "ORD001",
...   "customer_id": "Cus001",
...   "order_status": "Completed",
...   "order_date": 43992,
...   "shipped_date": 43994,
...   "store_id": 1,
...   "staff_id": 1
... },
... {
...   "order_id": "ORD002",
...   "customer_id": "Cus002",
...   "order_status": "Completed",
...   "order_date": 44221,
...   "shipped_date": 44227,
...   "store_id": 2,
...   "staff_id": 2
... },
... {
...   "order_id": "ORD003",
...   "customer_id": "Cus003",
...   "order_status": "Completed",
...   "order_date": 44306,
...   "shipped_date": 44314,
...   "store_id": 2,
...   "staff_id": 2
... }
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

Command Prompt - mongo

```
... "shipped_date": 44314,
... "store_id": 2,
... "staff_id": 2
...
... },
... {
... "order_id": "ORD004",
... "customer_id": "Cus004",
... "order_status": "Pending",
... "order_date": 44367,
... "shipped_date": 44377,
... "store_id": 3,
... "staff_id": 3
... },
... {
... "order_id": "ORD005",
... "customer_id": "Cus005",
... "order_status": "Pending",
... "order_date": 44367,
... "shipped_date": 44377,
... "store_id": 1,
... "staff_id": 1
... }
... ]
... )
```

```
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184d6a6080c37f46a3ad96"),
    ObjectId("62184d6a6080c37f46a3ad97"),
    ObjectId("62184d6a6080c37f46a3ad98"),
    ObjectId("62184d6a6080c37f46a3ad99"),
    ObjectId("62184d6a6080c37f46a3ad9a")
  ]
}
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

7. sales.staffs

```
Command Prompt - mongo
> db.sales.staffs.insertMany(
... [
... {
...   "staff_id": 1,
...   "first_name": "Pushpa",
...   "last_name": "Yadav",
...   "email": "pushpa@gmail.com",
...   "phone": 9999999999,
...   "active": "Yes",
...   "store_id": 1,
...   "manager_id": 1
... },
... {
...   "staff_id": 2,
...   "first_name": "Sadiksha",
...   "last_name": "Singh",
...   "email": "sadiksha@gmail.com",
...   "phone": 8888888888,
...   "active": "Yes",
...   "store_id": 2,
...   "manager_id": 1
... },
... {
...   "staff_id": 3,
...   "first_name": "Priya",
...   "last_name": "Nadar",
...   "email": "priya@gmail.com",
...   "phone": 7777777777,
...   "active": "Yes",
...   "store_id": 3,
...   "manager_id": 1
... }
... ]
... )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184e3f6080c37f46a3ad9b"),
    ObjectId("62184e3f6080c37f46a3ad9c"),
    ObjectId("62184e3f6080c37f46a3ad9d")
  ]
}
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

8. sales.stores

```
Command Prompt - mongo
> db.sales.stores.insertMany(
... [
... {
...   "store_id": 1,
...   "store_name": "Ambika Showroom",
...   "phone": 123456,
...   "email": "ambika@gmail.com",
...   "street": "GP",
...   "city": "Mumbai",
...   "state": "Maharashtra",
...   "zip_code": 400072
... },
... {
...   "store_id": 2,
...   "store_name": "Yash Bikes",
...   "phone": 789456,
...   "email": "yash.bikes@yahoo.com",
...   "street": "H.G",
...   "city": "Pune",
...   "state": "Maharashtra",
...   "zip_code": 300075
... },
... {
...   "store_id": 3,
...   "store_name": "Josh Automobiles",
...   "phone": 456983,
...   "email": "josh@gmail.com",
...   "street": "M.G",
...   "city": "Gandhi Nagar",
...   "state": "Gujrat",
...   "zip_code": 401235
... }
... ]
... )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184e886080c37f46a3ad9e"),
    ObjectId("62184e886080c37f46a3ad9f"),
    ObjectId("62184e886080c37f46a3ada0")
  ]
}
```


Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

9. Production.brands

```
CA: Command Prompt - mongo
> db.production.brands.insertMany(
... [
... {
...   "brand_id": 1,
...   "brand_name": "Honda"
... },
... {
...   "brand_id": 2,
...   "brand_name": "6KU Bikes"
... },
... {
...   "brand_id": 3,
...   "brand_name": "Bianchi"
... },
... {
...   "brand_id": 4,
...   "brand_name": "BMC"
... },
... {
...   "brand_id": 5,
...   "brand_name": "Huffy"
... }
... ]
... )
... )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("62184ecc6080c37f46a3ada1"),
    ObjectId("62184ecc6080c37f46a3ada2"),
    ObjectId("62184ecc6080c37f46a3ada3"),
    ObjectId("62184ecc6080c37f46a3ada4"),
    ObjectId("62184ecc6080c37f46a3ada5")
  ]
}
```

If you want to get more specific with a read operation and find a desired subsection of the records, you can use the previously mentioned filtering criteria to choose what results should be returned.

One of the most common ways of filtering the results is to search by value.

Example - `db.production.brands.find({"brand_name": "Honda"})`

```
CA: Command Prompt - mongo
}
> show dbs;
RetailBikeDB  0.000GB
admin         0.000GB
config        0.000GB
local         0.000GB
```

findOne()

In order to get one document that satisfies the search criteria, we can simply use the `findOne()` method on our chosen collection. If multiple documents satisfy the query, this method returns the first document according to the natural order which reflects the order of documents on the disk. If

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

no documents satisfy the search criteria, the function returns null. The function takes the following form of syntax.

Syntax- db.{collection}.findOne({query}, {projection})

Example – db.sales.order_items.findOne({"quantity": 2})

```
> db.production.brands.find({'brand_name': 'Honda'})
{ "_id" : ObjectId("62184ecc6080c37f46a3ada1"), "brand_id" : 1, "brand_name" : "Honda" }
> db.sales.order_items.findOne({"quantity": 2})
{
  "_id" : ObjectId("62184d186080c37f46a3ad91"),
  "order_id" : "ORD001",
  "product_id" : 1,
  "quantity" : 2,
  "list_price" : 50000
}
```

a) Specify Equality Condition

> db.production.products.find() - to show all the documents

> db.production.products.find({"list_price": 30000}) - to show only documents which have "list_price" as 30000

```
Command Prompt - mongo
> db.production.products.find()
{ "_id" : ObjectId("62184ab76080c37f46a3ad7b"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7c"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7d"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7e"), "product_id" : 4, "product_name" : "BMC Hybrid Bike", "brand_id" : 4, "category_id" : 3, "model_year" : 2009, "list_price" : 45000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7f"), "product_id" : 5, "product_name" : "Huffy Women Bike", "brand_id" : 5, "category_id" : 7, "model_year" : 2019, "list_price" : 50000 }
> db.production.products.find({"list_price": 30000})
{ "_id" : ObjectId("62184ab76080c37f46a3ad7c"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7d"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
> db.sales.customers.find()
{ "_id" : ObjectId("62184c366080c37f46a3ad8c"), "customer_id" : "Cus001", "first_name" : "Jay", "last_name" : "Mehta", "phone" : 1234567890, "email" : "jay@gmail.com", "street" : "T.P Road", "city" : "Mumbai", "state" : "Maharashtra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c366080c37f46a3ad8d"), "customer_id" : "Cus002", "first_name" : "Ruhi", "last_name" : "Singh", "phone" : 7894561230, "email" : "ruhi@yahoo.com", "street" : "M.G Chauk", "city" : "Mumbai", "state" : "Maharashtra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c366080c37f46a3ad8e"), "customer_id" : "Cus003", "first_name" : "Aria", "last_name" : "Josh", "phone" : 124567890, "email" : "aria.josh@gmail.com", "street" : "JVM", "city" : "Gandhi Nagar", "state" : "Gujrat", "zip_code" : 401235 }
{ "_id" : ObjectId("62184c366080c37f46a3ad8f"), "customer_id" : "Cus004", "first_name" : "Mahi", "last_name" : "Kaur", "phone" : 4567890123, "email" : "kaur.mahi@hotmail.com", "street" : "J.V.L.R", "city" : "Mumbai", "state" : "Maharashtra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c366080c37f46a3ad90"), "customer_id" : "Cus005", "first_name" : "Aditya", "last_name" : "Yadav", "phone" : 9638527410, "email" : "aditya@gmail.com", "street" : "Koliwada", "city" : "Pune", "state" : "Maharashtra", "zip_code" : 300075 }
```

b) Specify Conditions Using Query Operators

> db.sales.customers.find() - to show all the documents

> db.sales.customers.find({city: { \$in: ["Mumbai", "Pune"] } }) - to show all the documents where "city" is either "Mumbai" or "Pune"

```
> db.sales.customers.find({city: { $in: [ "Mumbai", "Pune" ] } })
{ "_id" : ObjectId("62184c366080c37f46a3ad8c"), "customer_id" : "Cus001", "first_name" : "Jay", "last_name" : "Mehta", "phone" : 1234567890, "email" : "jay@gmail.com", "street" : "T.P Road", "city" : "Mumbai", "state" : "Maharashtra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c366080c37f46a3ad8d"), "customer_id" : "Cus002", "first_name" : "Ruhi", "last_name" : "Singh", "phone" : 7894561230, "email" : "ruhi@yahoo.com", "street" : "M.G Chauk", "city" : "Mumbai", "state" : "Maharashtra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c366080c37f46a3ad8f"), "customer_id" : "Cus004", "first_name" : "Mahi", "last_name" : "Kaur", "phone" : 4567890123, "email" : "kaur.mahi@hotmail.com", "street" : "J.V.L.R", "city" : "Mumbai", "state" : "Maharashtra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c366080c37f46a3ad90"), "customer_id" : "Cus005", "first_name" : "Aditya", "last_name" : "Yadav", "phone" : 9638527410, "email" : "aditya@gmail.com", "street" : "Koliwada", "city" : "Pune", "state" : "Maharashtra", "zip_code" : 300075 }
```

c) Specify AND Conditions

> db.sales.order_items.find() - to show all the documents

> db.sales.order_items.find({quantity: 2, list_price: { \$gt: 70000}}) - Here we are trying to find how

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

sales order item documents for which quantity is 2 and list price is greater than 70000

```
> db.sales.order_items.find()
{ "_id" : ObjectId("62184d186080c37f46a3ad91"), "order_id" : "ORD001", "product_id" : 1, "quantity" : 2, "list_price" : 50000 }
{ "_id" : ObjectId("62184d186080c37f46a3ad92"), "order_id" : "ORD002", "product_id" : 2, "quantity" : 3, "list_price" : 90000 }
{ "_id" : ObjectId("62184d186080c37f46a3ad93"), "order_id" : "ORD003", "product_id" : 3, "quantity" : 1, "list_price" : 30000 }
{ "_id" : ObjectId("62184d186080c37f46a3ad94"), "order_id" : "ORD004", "product_id" : 4, "quantity" : 8, "list_price" : 360000 }
{ "_id" : ObjectId("62184d186080c37f46a3ad95"), "order_id" : "ORD005", "product_id" : 5, "quantity" : 2, "list_price" : 100000 }

> db.sales.order_items.find({quantity: 2, list_price: { $gt: 70000}})
{ "_id" : ObjectId("62184d186080c37f46a3ad95"), "order_id" : "ORD005", "product_id" : 5, "quantity" : 2, "list_price" : 100000 }
```

d)Specify OR Conditions

> db.production.products.find() - to show all the documents

> db.production.products.find({ \$or: [{ product_name: "Honda Superfast" }, { model_year : { \$lt: 2003 } }] }) - to show all the document which is either "product name" as "Honda Superfast" or "model year" is less than year 2003

```
> db.production.products.find()
{ "_id" : ObjectId("62184ab76080c37f46a3ad7b"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7c"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7d"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7e"), "product_id" : 4, "product_name" : "BMC Hybrid Bike", "brand_id" : 4, "category_id" : 3, "model_year" : 2009, "list_price" : 45000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7f"), "product_id" : 5, "product_name" : "Huffy Women Bike", "brand_id" : 5, "category_id" : 7, "model_year" : 2019, "list_price" : 50000 }

> db.production.products.find({ $or: [ { product_name: "Honda Superfast" }, { model_year : { $lt: 2003 } } ] })
{ "_id" : ObjectId("62184ab76080c37f46a3ad7b"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7c"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184ab76080c37f46a3ad7d"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
```

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

updateOne()

We can update a currently existing record and change a single document with an update operation.

To do this, we use the updateOne() method on a chosen collection. To update a document, we provide the method with two arguments: an update filter and an update action.

The update filter defines which items we want to update, and the update action defines how to update those items. We first pass in the update filter. Then, we use the "\$set" key and provide the fields we want to update as a value. This method will update the first record that matches the provided filter.

Example –

> db.sales.staffs.find() - to show current document in the system

> db.sales.staffs.updateOne({first_name: "Pushpa"}, {\$set:{phone: 999999988}}) - update mobile number from 9999999999 to 999999988 for document name where name is " first_name " in collection

```
> db.sales.staffs.find()
{ "_id" : ObjectId("62184e3f6080c37f46a3ad9b"), "staff_id" : 1, "first_name" : "Pushpa", "last_name" : "Yadav", "email" : "pushpa@gmail.com", "phone" : 9999999999, "active" : "Yes", "store_id" : 1, "manager_id" : 1 }
{ "_id" : ObjectId("62184e3f6080c37f46a3ad9c"), "staff_id" : 2, "first_name" : "Sadiksha", "last_name" : "Singh", "email" : "sadiksha@gmail.com", "phone" : 8888888888, "active" : "Yes", "store_id" : 2, "manager_id" : 1 }
{ "_id" : ObjectId("62184e3f6080c37f46a3ad9d"), "staff_id" : 3, "first_name" : "Priya", "last_name" : "Nadar", "email" : "priya@gmail.com", "phone" : 7777777777, "active" : "Yes", "store_id" : 3, "manager_id" : 1 }
>
>
>
>
> db.sales.staffs.updateOne({first_name: "Pushpa"}, {$set:{phone: 999999988}}) - update mobile
uncaught exception: SyntaxError: unexpected token: identifier :
@ (shell):1:86
> number from 9999999999 to 999999988 for document name where name is " first_name " in
uncaught exception: SyntaxError: unexpected token: 'from' :
@ (shell):1:7
>
>
>
> db.sales.staffs.updateOne({first_name: "Pushpa"}, {$set:{phone: 999999988}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

updateMany() allows us to update multiple items by passing in a list of items, just as we did when inserting multiple items. This update operation uses the same syntax for updating a single document.

Example –

db.sales.orders.find() - to show current document in the system

db.sales.orders.updateMany({shipped_date:44377}, {\$set: {shipped_date: "1-July-2021"}}) – with

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

the help of this command we are updating "shipped_date" to "1-July-2021" where shipped_date is 44377

```
> db.sales.orders.find()
{ "_id" : ObjectId("62184d6a6080c37f46a3ad96"), "order_id" : "ORD001", "customer_id" : "Cus001", "order_status" : "Completed", "order_date" : 43992, "shipped_date" : 43994, "store_id" : 1, "staff_id" : 1 }
{ "_id" : ObjectId("62184d6a6080c37f46a3ad97"), "order_id" : "ORD002", "customer_id" : "Cus002", "order_status" : "Completed", "order_date" : 44221, "shipped_date" : 44227, "store_id" : 2, "staff_id" : 2 }
{ "_id" : ObjectId("62184d6a6080c37f46a3ad98"), "order_id" : "ORD003", "customer_id" : "Cus003", "order_status" : "Completed", "order_date" : 44306, "shipped_date" : 44314, "store_id" : 2, "staff_id" : 2 }
{ "_id" : ObjectId("62184d6a6080c37f46a3ad99"), "order_id" : "ORD004", "customer_id" : "Cus004", "order_status" : "Pending", "order_date" : 44367, "shipped_date" : 44377, "store_id" : 3, "staff_id" : 3 }
{ "_id" : ObjectId("62184d6a6080c37f46a3ad9a"), "order_id" : "ORD005", "customer_id" : "Cus005", "order_status" : "Pending", "order_date" : 44367, "shipped_date" : 44377, "store_id" : 1, "staff_id" : 1 }
>
>
> db.sales.orders.updateMany({shipped_date:44377}, {$set: {shipped_date: "1-July-2021"}})
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
```

deleteOne()

deleteOne() is used to remove a document from a specified collection on the MongoDB server. A filter criteria is used to specify the item to delete. It deletes the first record that matches the provided filter.

Example –

> db.production.brands.find() - to show current documents in collection

> db.production.brands.deleteOne({brand_id:6}) - delete the document where "brand_id" is 6

```
> db.production.brands.find()
{ "_id" : ObjectId("62184ecc6080c37f46a3ada1"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada2"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada3"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada4"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada5"), "brand_id" : 5, "brand_name" : "Huffy" }
>
>
> db.production.brands.deleteOne({brand_id:6})
{ "acknowledged" : true, "deletedCount" : 0 }
```

deleteMany()

deleteMany() is a method used to delete multiple documents from a desired collection with a single delete operation. A list is passed into the method and the individual items are defined with filter criteria as in deleteOne().

Example –

> db.production.brands.find() - to show current documents in collection

Name: Vishal Mane

Roll No: 36

MSc Part-I Sem-2

> db.production.brands.deleteMany({brand_id: {\$gt: 5}}) - delete documents for which brand id is greater than 5

```
>
> db.production.brands.find()
{ "_id" : ObjectId("62184ecc6080c37f46a3ada1"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada2"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada3"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada4"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184ecc6080c37f46a3ada5"), "brand_id" : 5, "brand_name" : "Huffy" }
>
>
> db.production.brands.deleteMany({brand_id: {$gt: 5}})
{ "acknowledged" : true, "deletedCount" : 0 }
```