Experiment 5 - Performing CRUD Operations for unstructured data

For storing data in a MongoDB, you need to create a database first. It will allow you to systematically organize your data so that it can be retrieved as per requirement. If you wish to delete a database, MongoDB also allows you to delete that.

What is CRUD in MongoDB?

CRUD operations describe the conventions of a user-interface that let users view, search, and modify parts of the database.

MongoDB documents are modified by connecting to a server, querying the proper documents, and then changing the setting properties before sending the data back to the database to be updated. CRUD is data-oriented, and it's standardized according to HTTP action verbs.

When it comes to the individual CRUD operations:

1. The Create operation is used to insert new documents in the MongoDB database.
2. The Read operation is used to query a document in the database.
3. The Update operation is used to modify existing documents in the database.
4. The Delete operation is used to remove documents in the database.

Creating a Database in MongoDB

MongoDB has no "create" command for creating a database. Also, it is essential to note that MongoDB does not provide any specific command for creating a database. This might seem a bit agitated if you are new to this subject and database tool or in case you have used that conventional SQL as your database where you are required to create a new database, which will contain table and, you will then have to use the INSERT INTO TABLE to insert values manually within your table.

In this MongoDB tool, you need not have to produce, or it is optional to create a database manually. This is because MongoDB has the feature of automatically creating it for the first time for you, once you save your value in that collection. So, explicitly, you do not need to mention or put a command to create a database; instead it will be created automatically once the collection is filled with values.

The "use" Command_for_Creating_Database_in_MongoDB

You can make use of the "use" command followed by the database_name for creating a database.

This command will tell the MongoDB client to create a database by this name if there is no database exists by this name. Otherwise, this command will return the existing database that has the name.

Show list of databases > show dbs; admin 0.000GB config 0.000GB local 0.000GB

myNewDB   0.000GB **> use**

RetailBikeDB switched to

dbRetailBikeDB

Creating and showing collections in MongoDB

Collections are like that of tables of RDBMS and are capable enough to store documents of diverse or dissimilar types. Creation and removal of collections in MongoDB can be done in specific ways. In this chapter, you will learn about the creation of collections in a database created using MongoDB.

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

> db.createCollection("production.categories")

{ "ok" : 1 }

> db.createCollection("production.brands")

{ "ok" : 1 }

> db.createCollection("production.products")

{ "ok" : 1 }

> db.createCollection("production.stocks")

{ "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.

> show collections

production.brands

production.categories

production.products

production.stocks

sales.customers

sales.order_items

sales.orders sales.staffs

sales.stores

CRUD operation

CRUD operation is one of the essential concepts of a database system. Inserting data in the database comes under one of the CRUD operations. If you do not insert data in your database, you will not be able to continue with other activities within your document.

1) Create Operations

For MongoDB CRUD, if the specified collection doesn't exist, the create operation will create the collection when it's executed. Create operations in MongoDB target a single collection, not multiple collections. Insert operations in MongoDB are atomic on a single document level.

MongoDB provides two different create operations that you can use to insert documents into a collection:

1. db.collection.insertOne()
2. db.collection.insertMany()

The insertOne() method

As the namesake, insertOne() allows you to insert one document into the collection. Example -

db.movie.insertOne({ _id: 2, writername: "Stan Lee", name: "Aquaman" })

The insertMany() method

It's possible to insert multiple items at one time by calling the insertMany() method on the desired collection. In this case, we pass multiple items into our chosen collection (RetailDB) and separate them by commas. Within the parentheses, we use brackets to indicate that we are passing in a list of multiple entries. This is commonly referred to as a nested method.

I have taken example of Retail Bike store database to demonstrate insert crud operations.

1. production.categories

```
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"
    }
]
)
```

```
...
> 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("62184aee7d48a1997e6d64bf"),
                ObjectId("62184aee7d48a1997e6d64c0"),
                ObjectId("62184aee7d48a1997e6d64c1"),
                ObjectId("62184aee7d48a1997e6d64c2"),
                ObjectId("62184aee7d48a1997e6d64c3"),
                ObjectId("62184aee7d48a1997e6d64c4"),
                ObjectId("62184aee7d48a1997e6d64c5")
        ]
}
> db.production.categories.find()
{ "_id" : ObjectId("62184aee7d48a1997e6d64bf"), "category_id" : 1, "category_name" : "Road Bike" }
{ "_id" : ObjectId("62184aee7d48a1997e6d64c0"), "category_id" : 2, "category_name" : "Mountain Bike" }
{ "_id" : ObjectId("62184aee7d48a1997e6d64c1"), "category_id" : 3, "category_name" : "Hybrid Bike" }
{ "_id" : ObjectId("62184aee7d48a1997e6d64c2"), "category_id" : 4, "category_name" : "Folding Bike" }
{ "_id" : ObjectId("62184aee7d48a1997e6d64c3"), "category_id" : 5, "category_name" : "Touring Bike" }
{ "_id" : ObjectId("62184aee7d48a1997e6d64c4"), "category_id" : 6, "category_name" : "Cruiser Bike" }
{ "_id" : ObjectId("62184aee7d48a1997e6d64c5"), "category_id" : 7, "category_name" : "Women Bike" }
>
```

2. production.products

```
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","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
        }
    ]
)
```

```
> 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",
... "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("62184be27d48a1997e6d64c6"),
                ObjectId("62184be27d48a1997e6d64c7"),
                ObjectId("62184be27d48a1997e6d64c8"),
                ObjectId("62184be27d48a1997e6d64c9"),
                ObjectId("62184be27d48a1997e6d64ca")
        ]
}
> db.production.products.find()
{ "_id" : ObjectId("62184be27d48a1997e6d64c6"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c7"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c8"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c9"), "product_id" : 4, "product_name" : "BMC Hybrid Bike", "brand_id" : 4, "category_id" : 3, "model_year" : 2009, "list_price" : 45000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64ca"), "product_id" : 5, "product_name" : "Huffy Women Bike", "brand_id" : 5, "category_id" : 7, "model_year" : 2019, "list_price" : 50000 }
```

3. production.stocks

```
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,
        "quantity": 9
    }
]
)
```

```
> 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,
... "quantity": 9
... }
... ]
... )
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("62184c1a7d48a1997e6d64cb"),
                ObjectId("62184c1a7d48a1997e6d64cc"),
                ObjectId("62184c1a7d48a1997e6d64cd"),
                ObjectId("62184c1a7d48a1997e6d64ce"),
                ObjectId("62184c1a7d48a1997e6d64cf"),
                ObjectId("62184c1a7d48a1997e6d64d0")
        ]
}
> db.production.stocks.find()
{ "_id" : ObjectId("62184c1a7d48a1997e6d64cb"), "store_id" : 1, "product_id" : 1, "quantity" : 15 }
{ "_id" : ObjectId("62184c1a7d48a1997e6d64cc"), "store_id" : 2, "product_id" : 4, "quantity" : 20 }
{ "_id" : ObjectId("62184c1a7d48a1997e6d64cd"), "store_id" : 3, "product_id" : 5, "quantity" : 30 }
{ "_id" : ObjectId("62184c1a7d48a1997e6d64ce"), "store_id" : 1, "product_id" : 5, "quantity" : 100 }
{ "_id" : ObjectId("62184c1a7d48a1997e6d64cf"), "store_id" : 2, "product_id" : 2, "quantity" : 4 }
{ "_id" : ObjectId("62184c1a7d48a1997e6d64d0"), "store_id" : 3, "product_id" : 3, "quantity" : 9 }
>
```

4. sales.customers

```
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,
        "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
    }
]
)
```

```
> 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,
... "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("62184c887d48a1997e6d64d1"),
        ObjectId("62184c887d48a1997e6d64d2"),
        ObjectId("62184c887d48a1997e6d64d3"),
        ObjectId("62184c887d48a1997e6d64d4"),
        ObjectId("62184c887d48a1997e6d64d5")
    ]
}
> db.sales.customers.find()
{ "_id" : ObjectId("62184c887d48a1997e6d64d1"), "customer_id" : "Cus001", "first_name" : "Jay", "last_name" : "Mehta", "phone" : 1234567890, "email" : "jay@gmail.com", "street" : "T.P Road", "city" : "Mumbai", "state" : "Maharashtra", "z
ip_code" : 400072 }
{ "_id" : ObjectId("62184c887d48a1997e6d64d2"), "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("62184c887d48a1997e6d64d3"), "customer_id" : "Cus003", "first_name" : "Aria", "last_name" : "Josh", "phone" : 1245789630, "email" : "aria.josh@gmail.com", "street" : "JVM", "city" : "Gandhi Nagar", "state" : "Gujrat",
"zip_code" : 401235 }
{ "_id" : ObjectId("62184c887d48a1997e6d64d4"), "customer_id" : "Cus004", "first_name" : "Mahi", "last_name" : "Kaur", "phone" : 4567890123, "email" : "kaur.mahi@hotmail.com", "street" : "J.V.L.R", "city" : "Mumbai", "state" : "Maharasht
ra", "zip_code" : 400072 }
{ "_id" : ObjectId("62184c887d48a1997e6d64d5"), "customer_id" : "Cus005", "first_name" : "Aditya", "last_name" : "Yadav", "phone" : 9638527410, "email" : "aditya@gmail.com", "street" : "Koliwada", "city" : "Pune", "state" : "Maharashtra"
, "zip_code" : 300075 }
```

5. sales.order_items

```
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",
        "product_id": 5,
        "quantity": 2,
        "list_price": 100000
    }
]
)
```

```
> 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",
... "product_id": 5,
... "quantity": 2,
... "list_price": 100000
... }
...
... ]
... )
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("62184cd37d48a1997e6d64d6"),
                ObjectId("62184cd37d48a1997e6d64d7"),
                ObjectId("62184cd37d48a1997e6d64d8"),
                ObjectId("62184cd37d48a1997e6d64d9"),
                ObjectId("62184cd37d48a1997e6d64da")
        ]
}
>
> db.sales.order_items.find()
{ "_id" : ObjectId("62184cd37d48a1997e6d64d6"), "order_id" : "ORD001", "product_id" : 1, "quantity" : 2, "list_price" : 50000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d7"), "order_id" : "ORD002", "product_id" : 2, "quantity" : 3, "list_price" : 90000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d8"), "order_id" : "ORD003", "product_id" : 3, "quantity" : 1, "list_price" : 30000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d9"), "order_id" : "ORD004", "product_id" : 4, "quantity" : 8, "list_price" : 360000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64da"), "order_id" : "ORD005", "product_id" : 5, "quantity" : 2, "list_price" : 100000 }
>
```

6. sales.orders

```
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
},
{
        "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
}
]
)
```

```
> 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
...
... },
... {
... "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("62184d1d7d48a1997e6d64db"),
                ObjectId("62184d1d7d48a1997e6d64dc"),
                ObjectId("62184d1d7d48a1997e6d64dd"),
                ObjectId("62184d1d7d48a1997e6d64de"),
                ObjectId("62184d1d7d48a1997e6d64df")
        ]
}
```

```
> db.sales.order.find()
> db.sales.orders.find()
{ "_id" : ObjectId("62184d1d7d48a1997e6d64db"), "order_id" : "ORD001", "customer_id" : "Cus001", "order_status" : "Completed", "order_date" : 43992, "shipped_date" : 43994, "store_id" : 1, "staff_id" : 1 }
{ "_id" : ObjectId("62184d1d7d48a1997e6d64dc"), "order_id" : "ORD002", "customer_id" : "Cus002", "order_status" : "Completed", "order_date" : 44221, "shipped_date" : 44227, "store_id" : 2, "staff_id" : 2 }
{ "_id" : ObjectId("62184d1d7d48a1997e6d64dd"), "order_id" : "ORD003", "customer_id" : "Cus003", "order_status" : "Completed", "order_date" : 44306, "shipped_date" : 44314, "store_id" : 2, "staff_id" : 2 }
{ "_id" : ObjectId("62184d1d7d48a1997e6d64de"), "order_id" : "ORD004", "customer_id" : "Cus004", "order_status" : "Pending", "order_date" : 44367, "shipped_date" : 44377, "store_id" : 3, "staff_id" : 3 }
{ "_id" : ObjectId("62184d1d7d48a1997e6d64df"), "order_id" : "ORD005", "customer_id" : "Cus005", "order_status" : "Pending", "order_date" : 44367, "shipped_date" : 44377, "store_id" : 1, "staff_id" : 1 }
```

7. sales.staffs

```
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
    }
]
)
```

```
> 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("62184d7e7d48a1997e6d64e0"),
        ObjectId("62184d7e7d48a1997e6d64e1"),
        ObjectId("62184d7e7d48a1997e6d64e2")
    ]
}
> db.sales.staffs.find()
{ "_id" : ObjectId("62184d7e7d48a1997e6d64e0"), "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("62184d7e7d48a1997e6d64e1"), "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("62184d7e7d48a1997e6d64e2"), "staff_id" : 3, "first_name" : "Priya", "last_name" : "Nadar", "email" : "priya@gmail.com", "phone" : 7777777777, "active" : "Yes", "store_id" : 3, "manager_id" : 1 }
```

8. sales.stores

```
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
    }
]
```

)

9. production.brands

```
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"
    }
]
)
```

```
>
> 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("62184de57d48a1997e6d64e6"),
                ObjectId("62184de57d48a1997e6d64e7"),
                ObjectId("62184de57d48a1997e6d64e8"),
                ObjectId("62184de57d48a1997e6d64e9"),
                ObjectId("62184de57d48a1997e6d64ea")
        ]
}
>
>
> db.production.brands.find()
{ "_id" : ObjectId("62184de57d48a1997e6d64e6"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e7"), "brand_id" : 2, "brand_name" : "6KU Bikes"
{ "_id" : ObjectId("62184de57d48a1997e6d64e8"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e9"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184de57d48a1997e6d64ea"), "brand_id" : 5, "brand_name" : "Huffy" }
>
```

2) Read Operations

The read operations allow you to supply special query filters and criteria that let you specify which documents you want. The MongoDB documentation contains more information on the available query filters. Query modifiers may also be used to change how many results are returned.

MongoDB has two methods of reading documents from a collection:

- db.collection.find()
- db.collection.findOne()

find()
In order to get all the documents from a collection, we can simply use the find() method on our chosen collection. Executing just the find() method with no arguments will return all records currently in the collection.

Example - db.production.brands.find()

Here we can see that every record has an assigned "ObjectId" mapped to the "_id" key.

```
> show dbs;
RetailBikeDB  0.000GB
admin         0.000GB
config        0.000GB
local         0.000GB
myNewDB       0.000GB
> use RetailBikeDB
switched to db RetailBikeDB
> show collections;
production.brands
production.categories
production.products
production.stocks
sales.customers
sales.order_items
sales.orders
sales.staffs
sales.stores
> db.production.brands.find()
{ "_id" : ObjectId("60cf7f9b63a1711313736c8d"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("60cf7f9b63a1711313736c8e"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("60cf7f9b63a1711313736c8f"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("60cf7f9b63a1711313736c90"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("60cf7f9b63a1711313736c91"), "brand_id" : 5, "brand_name" : "Huffy" }
>
```

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"})

```
> show dbs;
RetailBikeDB  0.000GB
admin         0.000GB
config        0.000GB
local         0.000GB
> use RetailBikeDB;
switched to db RetailBikeDB
> db.production.brands.find({"brand_name" : "Honda"})
{ "_id" : ObjectId("62184de57d48a1997e6d64e6"), "brand_id" : 1, "brand_name" : "Honda" }
```

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 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.sales.order_items.find()
{ "_id" : ObjectId("62184cd37d48a1997e6d64d6"), "order_id" : "ORD001", "product_id" : 1, "quantity" : 2, "list_price" : 50000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d7"), "order_id" : "ORD002", "product_id" : 2, "quantity" : 3, "list_price" : 90000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d8"), "order_id" : "ORD003", "product_id" : 3, "quantity" : 1, "list_price" : 30000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d9"), "order_id" : "ORD004", "product_id" : 4, "quantity" : 8, "list_price" : 360000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64da"), "order_id" : "ORD005", "product_id" : 5, "quantity" : 2, "list_price" : 100000 }
> db.sales.order_items.findOne({"quantity": 2})
{
        "_id" : ObjectId("62184cd37d48a1997e6d64d6"),
        "order_id" : "ORD001",
        "product_id" : 1,
        "quantity" : 2,
        "list_price" : 50000
}
>
```

Query Documents

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

```
> db.production.products.find()
{ "_id" : ObjectId("62184be27d48a1997e6d64c6"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c7"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c8"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c9"), "product_id" : 4, "product_name" : "BMC Hybrid Bike", "brand_id" : 4, "category_id" : 3, "model_year" : 2009, "list_price" : 45000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64ca"), "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("62184be27d48a1997e6d64c7"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c8"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
>
```

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()
{ "_id" : ObjectId("62184c887d48a1997e6d64d1"), "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("62184c887d48a1997e6d64d2"), "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("62184c887d48a1997e6d64d3"), "customer_id" : "Cus003", "first_name" : "Aria", "last_name" : "Josh", "phone" : 1245789630, "email" : "aria.josh@gmail.com", "street" : "JVM", "city" : "Gandhi Nagar", "state" : "Gujrat", "zip_code" : 401235 }
{ "_id" : ObjectId("62184c887d48a1997e6d64d4"), "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("62184c887d48a1997e6d64d5"), "customer_id" : "Cus005", "first_name" : "Aditya", "last_name" : "Yadav", "phone" : 9638527410, "email" : "aditya@gmail.com", "street" : "Koliwada", "city" : "Pune", "state" : "Maharashtra", "zip_code" : 300075 }
> db.sales.customers.find({city: { $in: [ "Mumbai", "Pune" ] } })
{ "_id" : ObjectId("62184c887d48a1997e6d64d1"), "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("62184c887d48a1997e6d64d2"), "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("62184c887d48a1997e6d64d4"), "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("62184c887d48a1997e6d64d5"), "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 sales order item documents for which quantity is 2 and list price is greater than 70000

```
> db.sales.order_items.find()
{ "_id" : ObjectId("62184cd37d48a1997e6d64d6"), "order_id" : "ORD001", "product_id" : 1, "quantity" : 2, "list_price" : 50000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d7"), "order_id" : "ORD002", "product_id" : 2, "quantity" : 3, "list_price" : 90000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d8"), "order_id" : "ORD003", "product_id" : 3, "quantity" : 1, "list_price" : 30000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64d9"), "order_id" : "ORD004", "product_id" : 4, "quantity" : 8, "list_price" : 360000 }
{ "_id" : ObjectId("62184cd37d48a1997e6d64da"), "order_id" : "ORD005", "product_id" : 5, "quantity" : 2, "list_price" : 100000 }
> db.sales.order_items.find({quantity: 2, list_price : { $gt: 70000}})
{ "_id" : ObjectId("62184cd37d48a1997e6d64da"), "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("62184be27d48a1997e6d64c6"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c7"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c8"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c9"), "product_id" : 4, "product_name" : "BMC Hybrid Bike", "brand_id" : 4, "category_id" : 3, "model_year" : 2009, "list_price" : 45000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64ca"), "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("62184be27d48a1997e6d64c6"), "product_id" : 1, "product_name" : "Honda Superfast", "brand_id" : 1, "category_id" : 1, "model_year" : 1994, "list_price" : 25000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c7"), "product_id" : 2, "product_name" : "6KU Bikes", "brand_id" : 2, "category_id" : 4, "model_year" : 2000, "list_price" : 30000 }
{ "_id" : ObjectId("62184be27d48a1997e6d64c8"), "product_id" : 3, "product_name" : "Bianchi", "brand_id" : 3, "category_id" : 2, "model_year" : 2002, "list_price" : 30000 }
```

3) Update Operations

Like create operations, update operations operate on a single collection, and they are atomic at a single document level. An update operation takes filters and criteria to select the documents you want to update.

You should be careful when updating documents, as updates are permanent and can't be rolled back. This applies to delete operations as well.

For MongoDB CRUD, there are three different methods of updating documents:

- db.collection.updateOne()
- db.collection.updateMany()
- db.collection.replaceOne()

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 currect 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

updateMany()

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

replaceOne()

The replaceOne() method is used to replace a single document in the specified collection. replaceOne() replaces the entire document, meaning fields in the old document not contained in the new will be lost.

4)Delete Operations

Delete operations operate on a single collection, like update and create operations. Delete operations are also atomic for a single document. You can provide delete operations with filters andcriteria in order to specify which documents you would like to delete from a collection. The filter options rely on the same syntax that read operations utilize.

MongoDB has two different methods of deleting records from a collection:

- db.collection.deleteOne()
- db.collection.deleteMany()

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("62184de57d48a1997e6d64e6"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e7"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e8"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e9"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184de57d48a1997e6d64ea"), "brand_id" : 5, "brand_name" : "Huffy" }
> db.production.brands.deleteOne({brand_id:6})
{ "acknowledged" : true, "deletedCount" : 0 }
> db.production.brands.find()
{ "_id" : ObjectId("62184de57d48a1997e6d64e6"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e7"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e8"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e9"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184de57d48a1997e6d64ea"), "brand_id" : 5, "brand_name" : "Huffy" }
>
```

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

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

```
> db.production.brands.find()
{ "_id" : ObjectId("62184de57d48a1997e6d64e6"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e7"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e8"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e9"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184de57d48a1997e6d64ea"), "brand_id" : 5, "brand_name" : "Huffy" }
> db.production.brands.deleteMany({brand_id: {$gt: 5}})
{ "acknowledged" : true, "deletedCount" : 0 }
> db.production.brands.find()
{ "_id" : ObjectId("62184de57d48a1997e6d64e6"), "brand_id" : 1, "brand_name" : "Honda" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e7"), "brand_id" : 2, "brand_name" : "6KU Bikes" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e8"), "brand_id" : 3, "brand_name" : "Bianchi" }
{ "_id" : ObjectId("62184de57d48a1997e6d64e9"), "brand_id" : 4, "brand_name" : "BMC" }
{ "_id" : ObjectId("62184de57d48a1997e6d64ea"), "brand_id" : 5, "brand_name" : "Huffy" }
>
```