**MongoDB Exercise 1**

**Aims:**

1. **Getting familiar with the MongoDB Compass interface**: viewing databases and collections, exploring other features of MongoDB Compass.
2. **Accessing MongoDB Shell**
3. **Working with Mongosh - The MongoDB shell**: performing basic CRUD operations and running queries.

**Note:** *Before starting with this document,*

1. *if you are using your laptop/PC please make sure you have successfully installed MongoDB or follow instructions for installing MongoDB provided on BB.*
2. *if you are using UWE machine, you must have MongoDB compass successfully loaded on your screen or follow instructions provided on BB for using MongoDB Compass on UWEmachine*
3. *All the instructions/commands in this file are based on MongoDB version 8.0.4 and MongoDB compass 1.45.2. For other versions, there could be a slight change in the interface and/or commands which may need some adjustments.*

**The MongoDB Compass Interface:**

1. Click on the plus sign near localhost in the left pane. You will see a dialog box where you can enter name of the database and collection.
2. Enter name of the database and collection. For e.g. as mentioned in the screenshot. The collection ItemSales will be created within the database called SalesSytem.

A screenshot of a computer

Description automatically generated

1. Click on Create Database and you will see the following screen. Alternatively, you can click on the plus sign near the SalesSystem to create collection - Please keep exploring the interface.

A screenshot of a computer

Description automatically generated

1. After creating collection, you can insert your data using the Add Data button. It gives two options either to insert data document by document or to import data from a csv or json file.
2. First, we will import data using the import data command button at the bottom of the screen. Import SalesSystem.json and you will see something like the screenshot below:

A screenshot of a computer

Description automatically generated

1. Let’s try the second option for inserting data - document by document.

A screenshot of a computer

Description automatically generated

1. Select **insert document** as seen in the above screenshot. It will ask you to enter data in the JSON format in a dialog box. Paste the below text in that dialog box. *Make sure you are entering a comma after the \_id{}, else it will give you an error.*

|  |
| --- |
| "item": {  "ItemNo": 13001,  "Description": "Earings 13 century",  "PriceGBP": 34000  },  "order": {  "OrderNumber": "PO1732",  "DateofPurchase": "22/12/2017",  "Delivered": "N"  },  "customer": {  "CustomerNumber": "CU211",  "CustomerName": "George Andrews",  "CustomerAddress": "27 Tree Road, Bristol, BS7 3TY",  "CustomerTelNo": 1173786935  },  "payment\_card": {  "PaymentCard": 3345470000000000,  "CardType": "Amex",  "ExpiryDate": "31/12/2024"  } |

1. Click on insert – If everything is well, you have inserted a document using the MongoDB Compass interface!

A screenshot of a computer

Description automatically generated

1. To edit a document, you can click on the edit document button as shown in the screenshot and edit the fields that you want to edit.

A screenshot of a computer

Description automatically generated

1. To delete a document, you can click on the delete document button as shown in the screenshot.

A screenshot of a computer

Description automatically generated

1. Find button can be used to display documents. You can give parameters in the space provided to add filters for the query and click on find. If you click on find without giving any filter, it will display all documents.

**Accessing MongoDB Shell**

In this part of the exercise, we will be accessing MongoDB shell to perform operations using command line which we did above using the GUI. You can access the embedded MongoDB Shell through the MongoDB Compass interface. We will use the embedded MongoDB shell for our exercises.

A screenshot of a computer

Description automatically generated

*Note: In the older version of MongoDB you will find the MongoDB shell at the bottom.*

**Using the MongoDB Shell for performing CRUD (Create, Read, Update, Delete) operations.**

A screenshot of a computer

Description automatically generated

**Create Database**

Type the following command to switch to another database if you are not on SalesSystem. If the database doesn’t exist, a new database will be created when the same command is executed. If you are already there, move to next step.

**>use SalesSystem**

1. Now you are on the **SalesSystem>** prompt. You can work with all the collections inside the database SalesSystem. All the commands that you execute will be executed on the SalesSystem database.

**Create Collection**

To create a new collection you can use the command

**db.createCollection(“collection\_name”)**

However, we have already created a collection called ItemSales and will be working with that collection for further exercise. However, please try out using the MongoDB shell to create a new collection for your practice.

**Insert**

Commands like **db.collection.insertOne()** or **db.collection.insertMany()** can be used to insert documents in the MongoDB.

|  |  |
| --- | --- |
| db.collection.insertOne[()](https://www.mongodb.com/docs/manual/reference/method/db.collection.insertOne/) | Inserts a single [document](https://www.mongodb.com/docs/manual/core/document/) into a collection. |
| db.collection.insertMany[()](https://www.mongodb.com/docs/manual/reference/method/db.collection.insertMany/) | Inserts multiple [documents](https://www.mongodb.com/docs/manual/core/document/) into a collection. |

For eg the below example will insert one document in the collection called ItemSales. Try it out!

db.SalesSystem.insertOne({

"item": {

"ItemNo": 13001,

"Description": "Earings 13 century",

"PriceGBP": 34000

},

"order": {

"OrderNumber": "PO1732",

"DateofPurchase": "22/12/2017",

"Delivered": "N"

},

"customer": {

"CustomerNumber": "CU211",

"CustomerName": "George Andrews",

"CustomerAddress": "27 Tree Road, Bristol, BS7 3TY",

"CustomerTelNo": 1173786935

},

"payment\_card": {

"PaymentCard": 3345470000000000,

"CardType": "Amex",

"ExpiryDate": "31/12/2024"

}

})

**Find:**

You can use the db.collection.find command to display the documents from a collection. If you don’t give any criteria, all documents with all the fields are displayed. You can also display specific field(s) with specific criteria, if you need to execute such queries.

**db.ItemSales.find()** will display all the documents in the ItemSales collection

**db.ItemSales.find({'item.PriceGBP': {$gt: 250 }})**

Displays all the fields with price £250

**db.ItemSales.find({'item.PriceGBP': {$gt: 250 }},{'item.ItemNo':1, 'item.Description':1})**

The above command will display all the documents with ItemNo and Description for the items with price = £250

You can also use various operators (Boolean, arithmetical, comparison etc) in the find command as well. For eg the below query will display Itemno, priceGBP, and description for all the items with price > 250. The details on operators and mapping with SQL operators is provided in the quick reference guide.

**db.ItemSales.find({'item.PriceGBP': {$gt: 250 }},{'item.ItemNo':1, 'item.Description':1, 'item.PriceGBP':1})**

**A screenshot of a computer program

AI-generated content may be incorrect.**

\_id works as a primary key in the MongoDB database. If you look closely, you can see \_id contains objectID(some \_number) which is a unique number! It is always displayed when you use the find command unless it is explicitly specified as \_id:0. See the below example.

**db.ItemSales.find({'item.PriceGBP': {$gt: 250 }},{'item.ItemNo':1, 'item.Description':1, 'item.PriceGBP':1,\_id:0})**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Update:**

You can update your documents in the collection using collection.updateOne(), collection.updateMany(), and collection.replaceOne(). It uses $set operator to modify values. $set will also create a field if the field does not exists!

The following command will update description field for the document with ItemNo 12502 in the ItemSales collection.

**db.ItemSales.updateOne(**

**{ 'item.ItemNo': 12502 },**

**{**

**$set: { 'item.Description': '18 century chair'}**

**})**

**A blue screen with white text

AI-generated content may be incorrect.**

The following command will add a new field deliveredDate in all the documents in the itemsales.

**db.ItemSales.updateMany(**

**{},**

**{**

**$set: { 'DeliveredDate': ''}**

**})**

**Delete**

To delete a document(s), db.collection.deleteOne(), db.collection.deleteMany() and db.collection.remove() commands are used.

|  |  |
| --- | --- |
| db.collection.deleteOne[()](https://www.mongodb.com/docs/manual/reference/method/db.collection.deleteOne/) | Delete at most a single document that matches a specified filter |
| db.collection.deleteMany[()](https://www.mongodb.com/docs/manual/reference/method/db.collection.deleteMany/) | Delete all documents that matches a specified filter. |
| db.collection.remove[()](https://www.mongodb.com/docs/manual/reference/method/db.collection.remove/) | Delete a single document or all documents that matches a specified filter. |

**Good to know:**

Please try these commands..

1. Use show.help to display help of the show as below:

A computer screen shot of text

AI-generated content may be incorrect.

Show dbs will display all the databases on the server, show collection shows the collection in the current database, and so on.

1. Use the db.help to see all the db commands, go through those one by to understand what they do.

A screenshot of a computer

Description automatically generated

1. Use db.collection.help eg. db.SalesSystem.help to go through all the commands related to collection SalesSystem.

A screenshot of a computer

Description automatically generated

Explore the MongoDB Compass interface, explore various commands in Mongosh shell to get more familiar with MongoDB.