In compass

Create a collection with the name books and insert

[

{ "\_id" : 8752, "title" : "Divine Comedy", "author" : "Dante", "copies" : 1 },

{ "\_id" : 7000, "title" : "The Odyssey", "author" : "Homer", "copies" : 10 },

{ "\_id" : 7020, "title" : "Iliad", "author" : "Homer", "copies" : 10 },

{ "\_id" : 8645, "title" : "Eclogues", "author" : "Dante", "copies" : 2 },

{ "\_id" : 8751, "title" : "The Banquet", "author" : "Dante", "copies" : 2 }

]

—----------------------------------------------------------------------------------------------------------—-

Mongo shell commands

Download from –<https://www.mongodb.com/try/download/shell2>

show dbs

create or change database

use blog

use posts

Creating a collection

db.createCollection("posts")

**inserting an object**

db.posts.insertOne(object)

There are 2 methods to insert documents into a MongoDB database.

insertOne() - single document

*db.posts.insertOne({*

*title: "Post Title 1",*

*Body : "Body of post.",*

*category: "News",*

*likes: 1,*

*tags: ["news", "events"],*

*date: Date()*

*})*

**insertMany()**

To insert multiple documents at once, use the insertMany() method.

*db.posts.insertMany([*

*{*

*title: "Post Title 2",*

*body: "Body of post.",*

*category: "Event",*

*likes: 2,*

*tags: ["news", "events"],*

*date: Date()*

*},*

*{*

*title: "Post Title 3",*

*body: "Body of post.",*

*category: "Technology",*

*likes: 3,*

*tags: ["news", "events"],*

*date: Date()*

*},*

*{*

*title: "Post Title 4",*

*body: "Body of post.",*

*category: "Event",*

*likes: 4,*

*tags: ["news", "events"],*

*date: Date()*

*}*

*])*

—------------------------------------------------------

finding the data

Find Data

There are 2 methods to find and select data from a MongoDB collection, find() and findOne().

db.posts.find()

no argument means all will return

findOne()

To select only one document, we can use the findOne() method.

db.posts.findOne() - if no argument first one

db.posts.find( {category: "News"} )

—-------------------------------------------------------------------------------------

Update Document

To update an existing document we can use the updateOne() or updateMany() methods.

updateOne()

The updateOne() method will update the first document that is found matching the provided query.

db.posts.find( { title: "Post Title 1" } )

Now let's update the "likes" on this post to 2. To do this, we need to use the $set operator.

db.posts.updateOne( { title: "Post Title 1" }, { $set : { likes: 2 } } )

db.posts.find( { title: "Post Title 1" } )

updateMany()

The updateMany() method will update all documents that match the provided query.

Update likes on all documents by 1. For this we will use the $inc (increment) operator:

db.posts.updateMany({}, { $inc: { likes: 1 } })

—------------------------

Delete Documents

We can delete documents by using the methods deleteOne() or deleteMany().

deleteOne()

The deleteOne() method will delete the first document that matches the query provided.

Example

db.posts.deleteOne({ title: "Post Title 5" })

deleteMany()

The deleteMany() method will delete all documents that match the query provided.

Example

db.posts.deleteMany({ category: "Technology" })

—-------------------------------

**Projection:**

db.createCollection("inventory")

**insert documents**

db.inventory.insertMany( [

{ item: "journal", status: "A", size: { h: 14, w: 21, uom: "cm" }, instock: [ { warehouse: "A", qty: 5 } ] },

{ item: "notebook", status: "A", size: { h: 8.5, w: 11, uom: "in" }, instock: [ { warehouse: "C", qty: 5 } ] },

{ item: "paper", status: "D", size: { h: 8.5, w: 11, uom: "in" }, instock: [ { warehouse: "A", qty: 60 } ] },

{ item: "planner", status: "D", size: { h: 22.85, w: 30, uom: "cm" }, instock: [ { warehouse: "A", qty: 40 } ] },

{ item: "postcard", status: "A", size: { h: 10, w: 15.25, uom: "cm" }, instock: [ { warehouse: "B", qty: 15 }, { warehouse: "C", qty: 35 } ] }

]);

db.inventory.find()

db.inventory.find( { status: "A" } )

db.inventory.find( { status: "A" }, { item: 1, status: 1 } )

db.inventory.find( { status: "A" }, { item: 1, status: 1, \_id: 0 } )

db.inventory.find( { status: "A" }, { status: 0, instock: 0 } )

—------------------------

Embedded document

A document inside another document.

when a collection has a document, this document contains another document, another document contains another sub-document, and so on, then such types of documents are known as embedded/nested documents.

Create a collection with the name inventory2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

insert the following records

db.inventory2.insertMany([

{ item: "journal", qty: 25, tags: ["blank", "red"], dim\_cm: [ 14, 21 ] },

{ item: "notebook", qty: 50, tags: ["red", "blank"], dim\_cm: [ 14, 21 ] },

{ item: "paper", qty: 100, tags: ["red", "blank", "plain"], dim\_cm: [ 14, 21 ] },

{ item: "planner", qty: 75, tags: ["blank", "red"], dim\_cm: [ 22.85, 30 ] },

{ item: "postcard", qty: 45, tags: ["blue"], dim\_cm: [ 10, 15.25 ] }

]);

—------------------------