Database Systems Lab

Lab#15 – MongoDB Basics

Table of contents

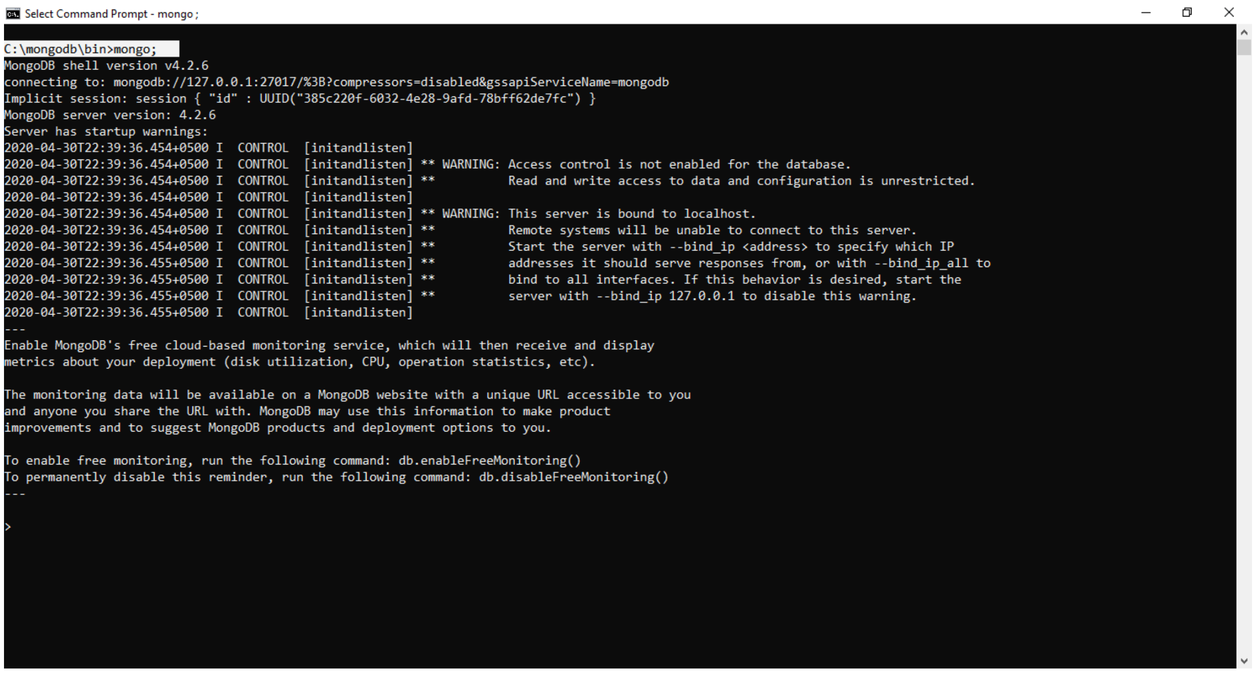
Video to watch

https://www.youtube.com/watch?v=pWbMrx5rVBE&t=1576s

Running a mongo shell

Open terminal and go to mongodb->bin folder

Write “mongo”

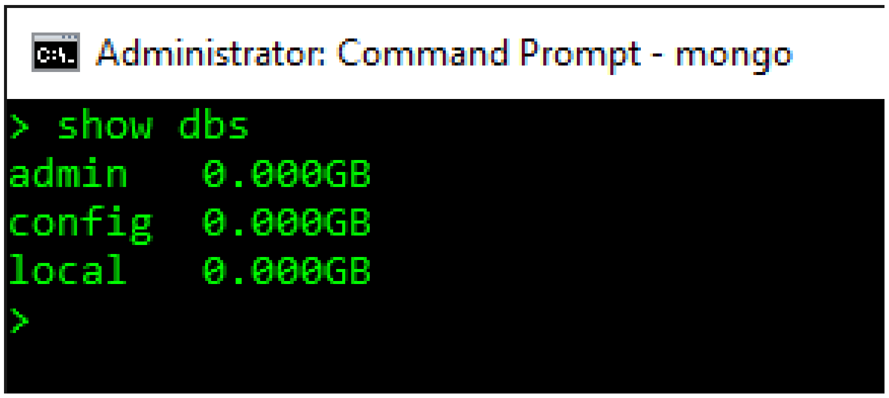


```
C:\mongodb\bin>mongo;
MongoDB shell version v4.2.6
connecting to: mongodb://127.0.0.1:27017/%3B?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("385c220f-6032-4e28-9afd-78bff62de7fc") }
MongoDB server version: 4.2.6
Server has startup warnings:
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten]
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten]
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten] ** WARNING: This server is bound to localhost.
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten] **          Remote systems will be unable to connect to this server.
2020-04-30T22:39:36.454+0500 I  CONTROL  [initandlisten] **          Start the server with --bind_ip <address> to specify which IP
2020-04-30T22:39:36.455+0500 I  CONTROL  [initandlisten] **          addresses it should serve responses from, or with --bind_ip_all to
2020-04-30T22:39:36.455+0500 I  CONTROL  [initandlisten] **          bind to all interfaces. If this behavior is desired, start the
2020-04-30T22:39:36.455+0500 I  CONTROL  [initandlisten] **          server with --bind_ip 127.0.0.1 to disable this warning.
2020-04-30T22:39:36.455+0500 I  CONTROL  [initandlisten]
---
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()
---

>
```

To clear the screen just type “cls”, it will clear the screen

Show databases:

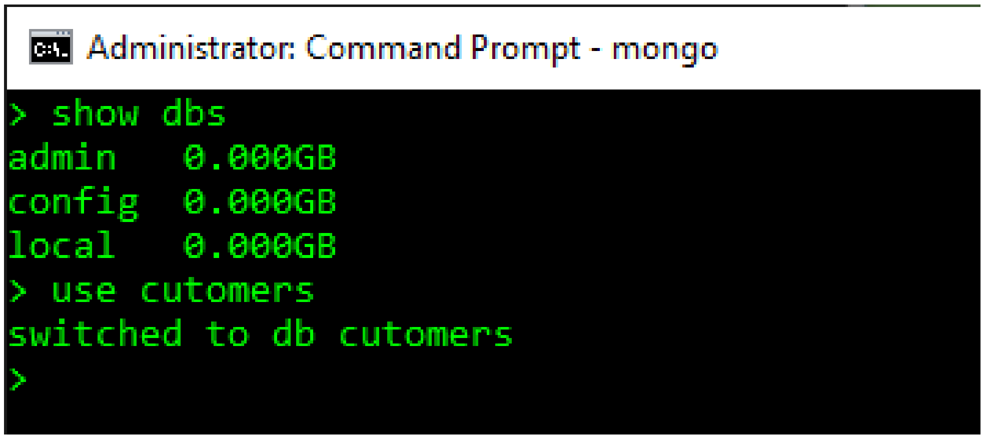
To see the databases, write **show dbs**



```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

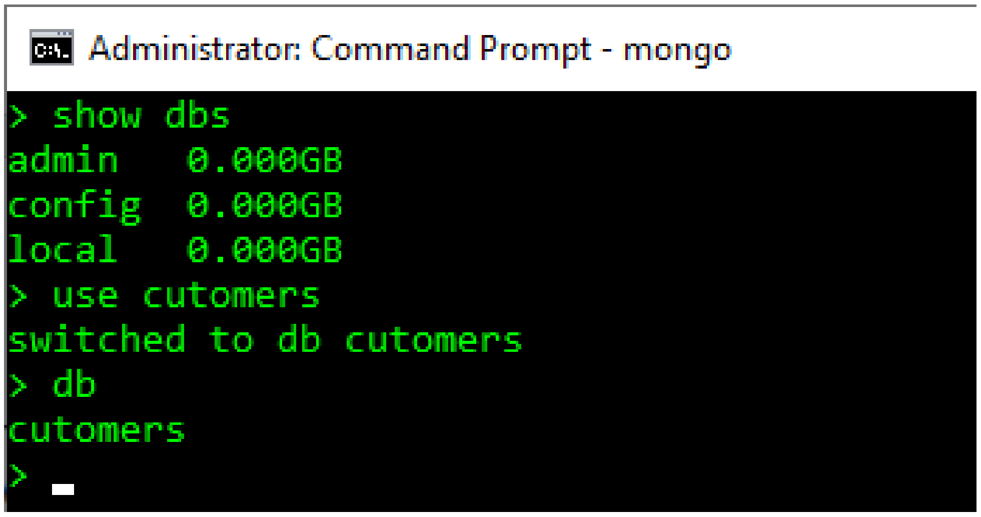
Create databases:

To create new database write “**use database_name**” e.g use customers. It will create and switch to that database


Administrator: Command Prompt - mongo

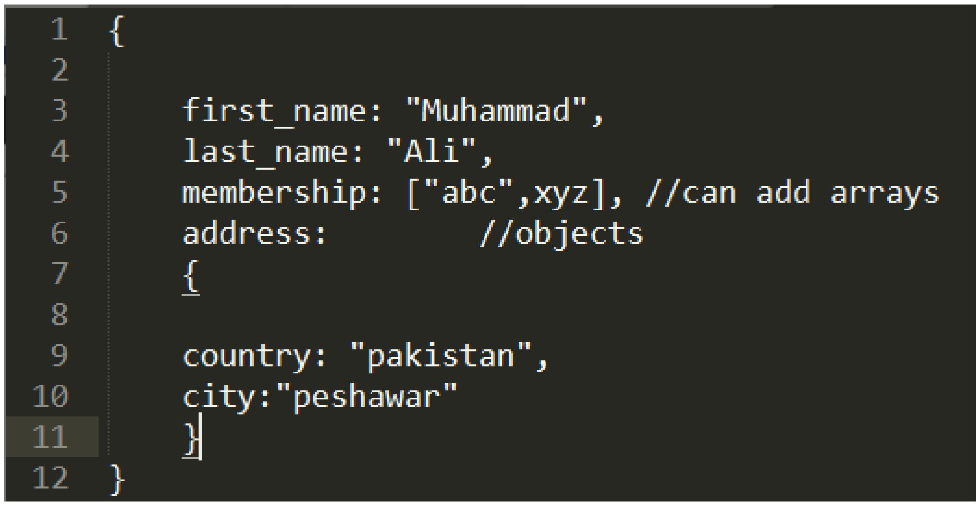
```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use cutomers
switched to db cutomers
>
```

To check the current database you are working in, write “**db**”.


Administrator: Command Prompt - mongo

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use cutomers
switched to db cutomers
> db
cutomers
> _
```

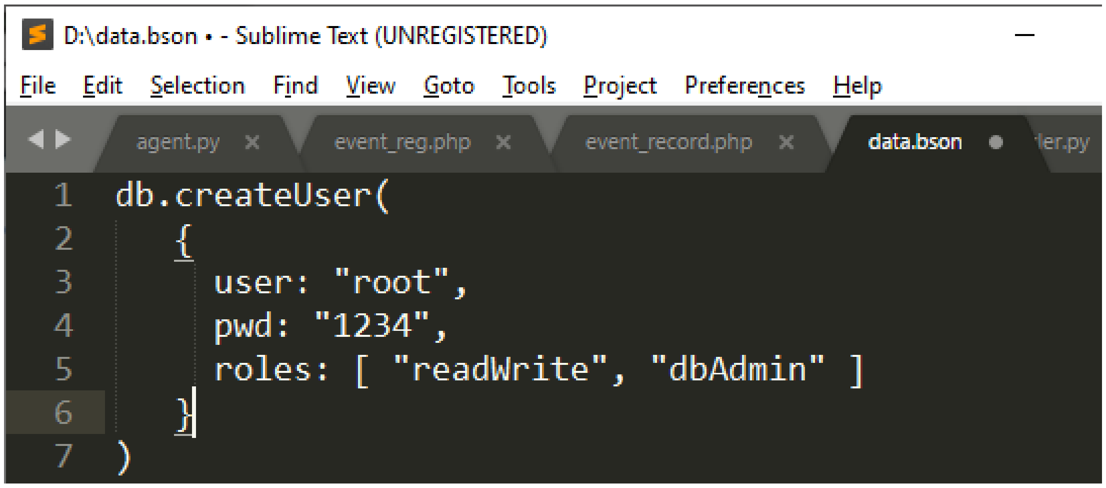
Basic syntax for the query’s. It’s like json start and end with curly braces



```
{

    first_name: "Muhammad",
    last_name: "Ali",
    membership: ["abc",xyz], //can add arrays
    address:        //objects
    {

    country: "pakistan",
    city:"peshawar"
    }
}
```

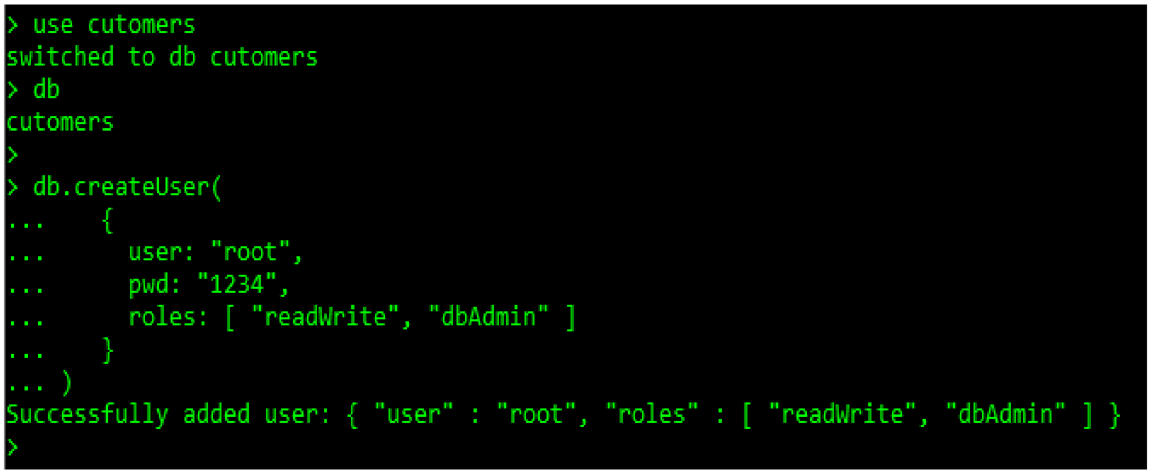
Create User:

Before inserting the data, lets make a user for the database customers. For more detail see this link : https://docs.mongodb.com/manual/reference/method/db.createUser/



D:\data.bson • - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
agent.py
event_reg.php
event_record.php
data.bson
ler.py

```
db.createUser(
   {
     user: "root",
     pwd: "1234",
     roles: [ "readWrite", "dbAdmin" ]
   }
)
```



```
> use cutomers
switched to db cutomers
> db
cutomers
>
> db.createUser(
...    {
...      user: "root",
...      pwd: "1234",
...      roles: [ "readWrite", "dbAdmin" ]
...    }
... )
Successfully added user: { "user" : "root", "roles" : [ "readWrite", "dbAdmin" ] }
>
```

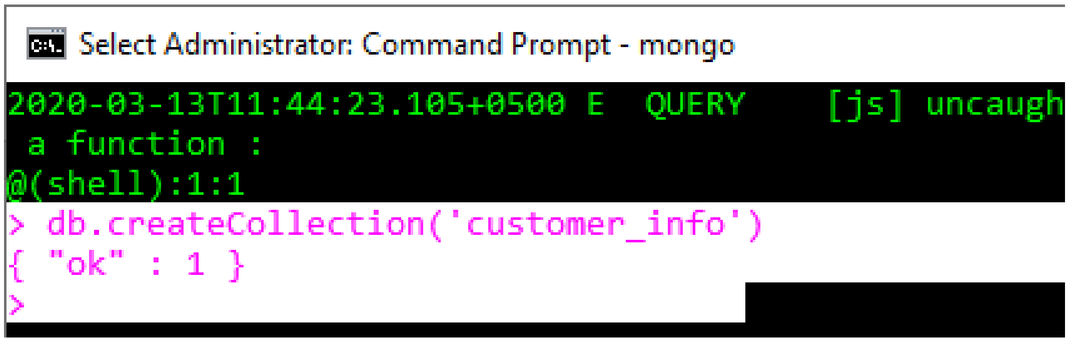
Create Collections:

They are similar to tables in a relational database. Basically the hold documents or the records. To create one type

MongoDB stores BSON documents, i.e. data records, in collections; the collections in databases.

After creating user, lets add data. But before that make a collection first

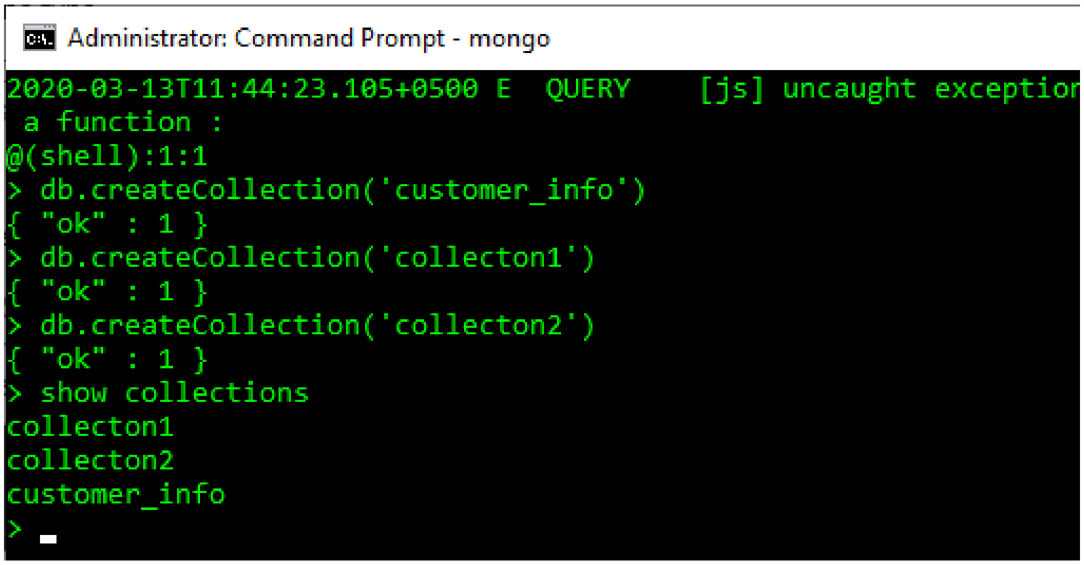
db.createCollection('collectionname')



Select Administrator: Command Prompt - mongo

```
2020-03-13T11:44:23.105+0500 E  QUERY    [js] uncaugh
 a function :
@(shell):1:1
> db.createCollection('customer_info')
{ "ok" : 1 }
>
```

To see all the collections in database write "**show collections**"



Administrator: Command Prompt - mongo

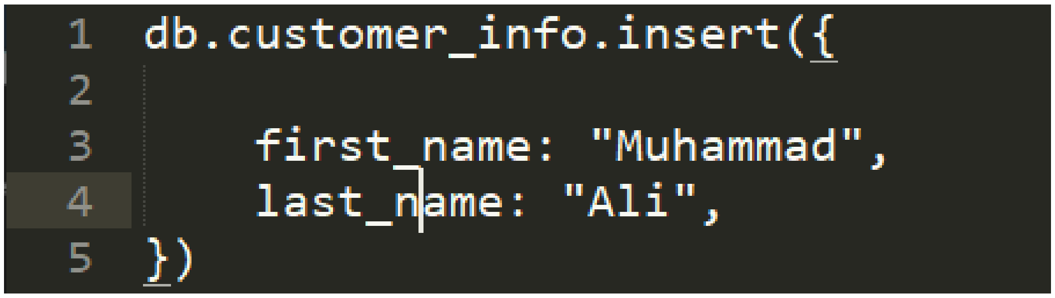
```
2020-03-13T11:44:23.105+0500 E  QUERY    [js] uncaught exceptior
 a function :
@(shell):1:1
> db.createCollection('customer_info')
{ "ok" : 1 }
> db.createCollection('collecton1')
{ "ok" : 1 }
> db.createCollection('collecton2')
{ "ok" : 1 }
> show collections
collecton1
collecton2
customer_info
> _
```

Insert documents:

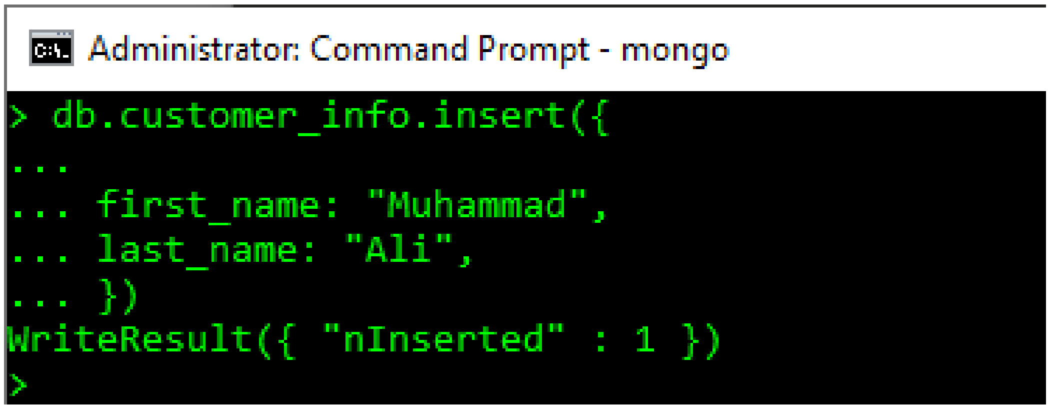
Insert document into collection

Single document

db.customer_info.insert({write data in json format as explained earlier})



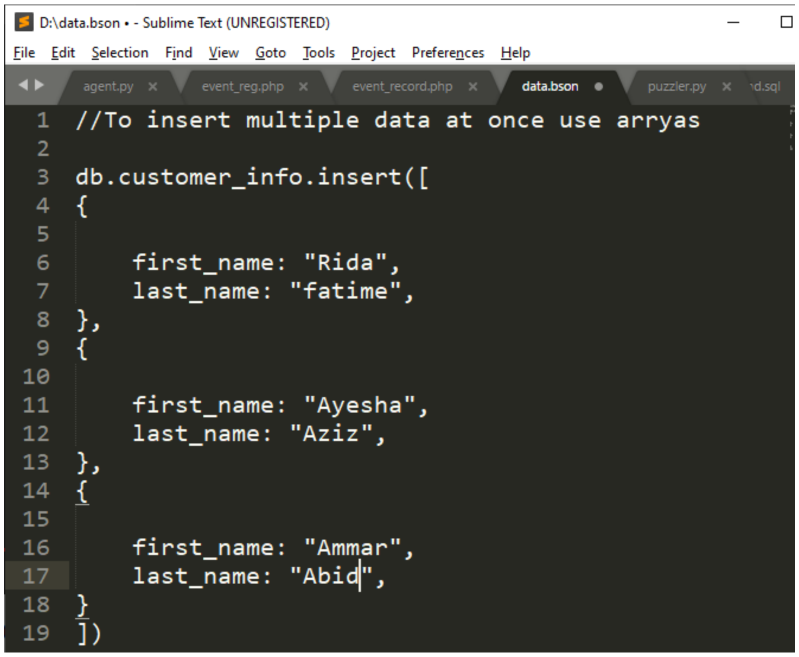
```
db.customer_info.insert({

	first_name: "Muhammad",
	last_name: "Ali",
})
```



Administrator: Command Prompt - mongo

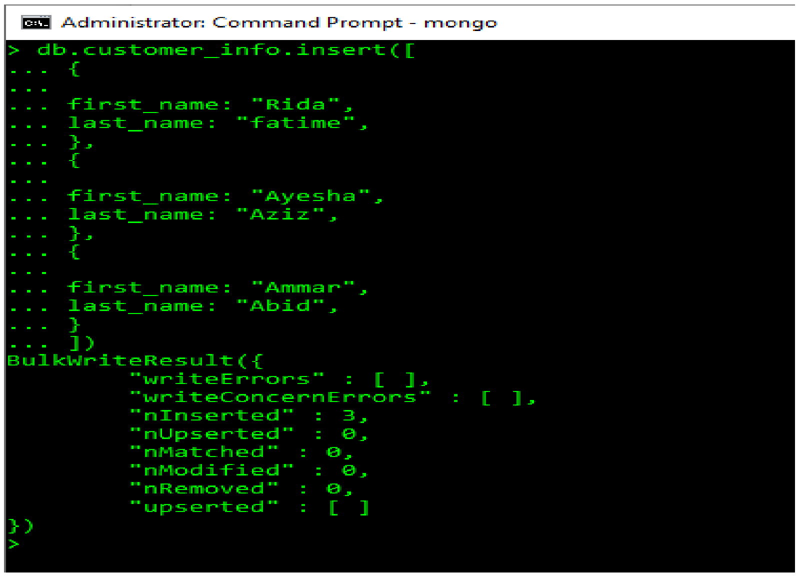
```
> db.customer_info.insert({
...
... first_name: "Muhammad",
... last_name: "Ali",
... })
WriteResult({ "nInserted" : 1 })
>
```

Multiple records



D:\data.bson • - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
agent.py
event_reg.php
event_record.php
data.bson
puzzler.py
d.sql

```
//To insert multiple data at once use arryas

db.customer_info.insert([
{

	first_name: "Rida",
	last_name: "fatime",
},
{

	first_name: "Ayesha",
	last_name: "Aziz",
},
{

	first_name: "Ammar",
	last_name: "Abid",
}
])
```

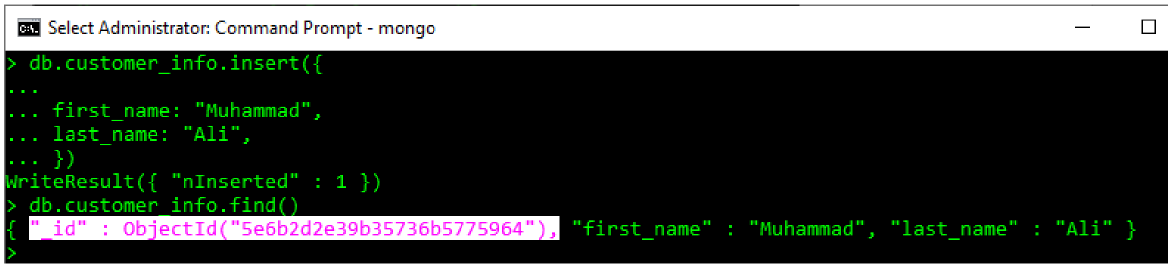

Administrator: Command Prompt - mongo

```
> db.customer_info.insert([
... {
...
... first_name: "Rida",
... last_name: "fatime",
... },
... {
...
... first_name: "Ayesha",
... last_name: "Aziz",
... },
... {
...
... first_name: "Ammar",
... last_name: "Abid",
... }
... ])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 3,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```

View records

To see the documents in collection "customer_info"

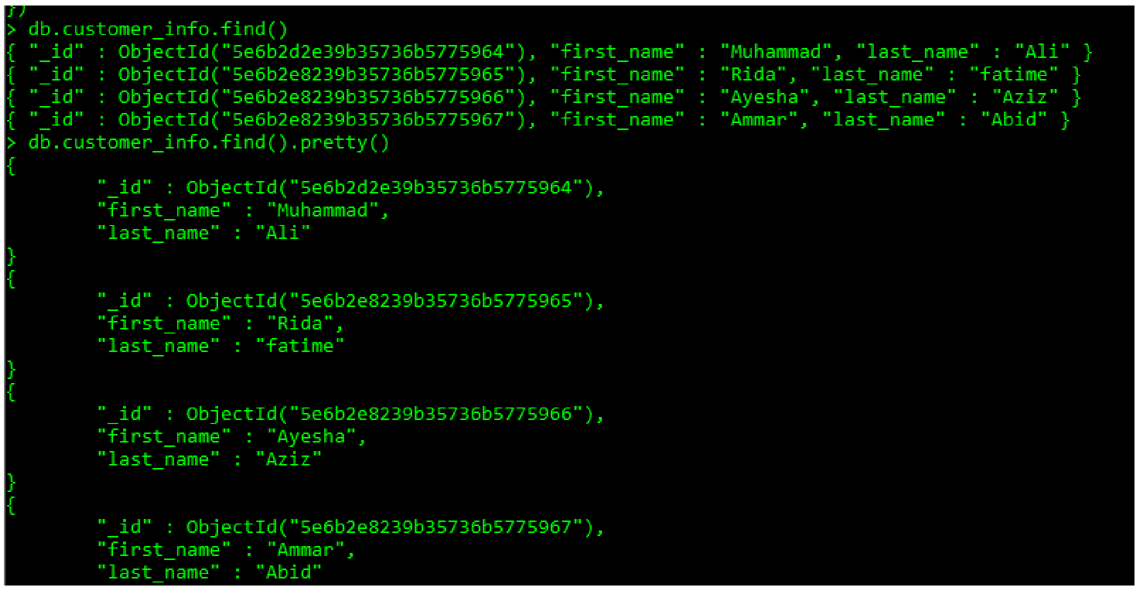
db.customer_info.find()


Select Administrator: Command Prompt - mongo

```
> db.customer_info.insert({
...
... first_name: "Muhammad",
... last_name: "Ali",
... })
WriteResult({ "nInserted" : 1 })
> db.customer_info.find()
{ "_id" : ObjectId("5e6b2d2e39b35736b5775964"), "first_name" : "Muhammad", "last_name" : "Ali" }
>
```

The data has been inserted. Here you can see the id object. It's a unique id generated automatically now we don't have to worry about primary key etc

db.customer_info.find().pretty()



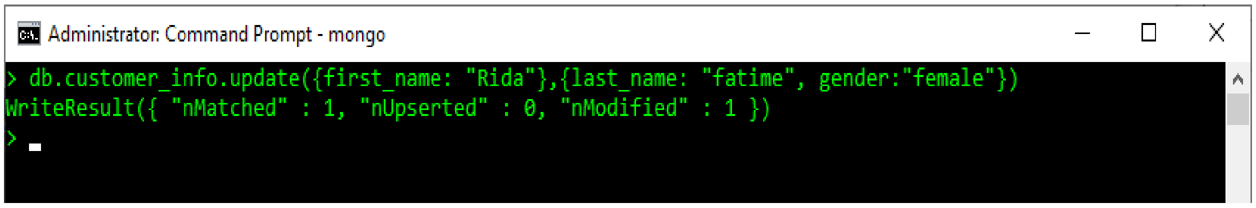
```
> db.customer_info.find()
{ "_id" : ObjectId("5e6b2d2e39b35736b5775964"), "first_name" : "Muhammad", "last_name" : "Ali" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775965"), "first_name" : "Rida", "last_name" : "fatime" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775966"), "first_name" : "Ayesha", "last_name" : "Aziz" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775967"), "first_name" : "Ammar", "last_name" : "Abid" }
> db.customer_info.find().pretty()
{
        "_id" : ObjectId("5e6b2d2e39b35736b5775964"),
        "first_name" : "Muhammad",
        "last_name" : "Ali"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775965"),
        "first_name" : "Rida",
        "last_name" : "fatime"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775966"),
        "first_name" : "Ayesha",
        "last_name" : "Aziz"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775967"),
        "first_name" : "Ammar",
        "last_name" : "Abid"
```

update

To update the record existing record:

db.customer_info.update({field name with value to find },{data to insert})

db.customer_info.update({first_name: "Rida"},{last_name: "fatime", gender:"female"})



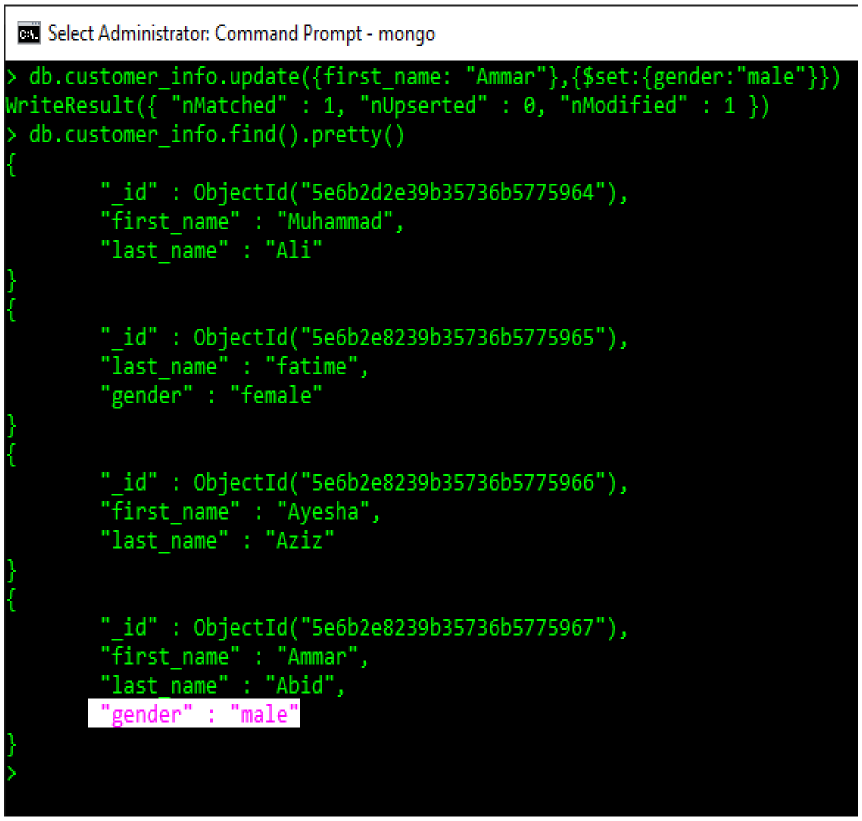
```
> db.customer_info.update({first_name: "Rida"},{last_name: "fatime", gender:"female"})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

But in above example inorder to update gender we had to write whole record, if not it will replace whole record with just gender

set

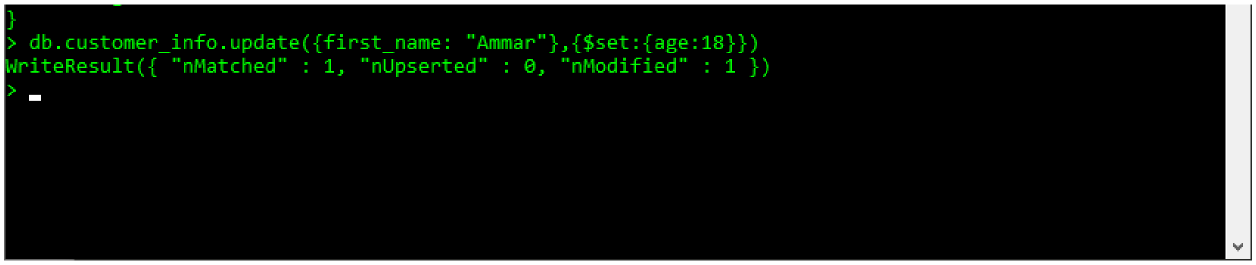
To add new field in a specific record

db.customer_info.update({first_name: "Ammar"},{$set:{ gender:"male"}})



Select Administrator: Command Prompt - mongo

```
> db.customer_info.update({first_name: "Ammar"},{$set:{gender:"male"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.customer_info.find().pretty()
{
        "_id" : ObjectId("5e6b2d2e39b35736b5775964"),
        "first_name" : "Muhammad",
        "last_name" : "Ali"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775965"),
        "last_name" : "fatime",
        "gender" : "female"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775966"),
        "first_name" : "Ayesha",
        "last_name" : "Aziz"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775967"),
        "first_name" : "Ammar",
        "last_name" : "Abid",
        "gender" : "male"
}
>
```

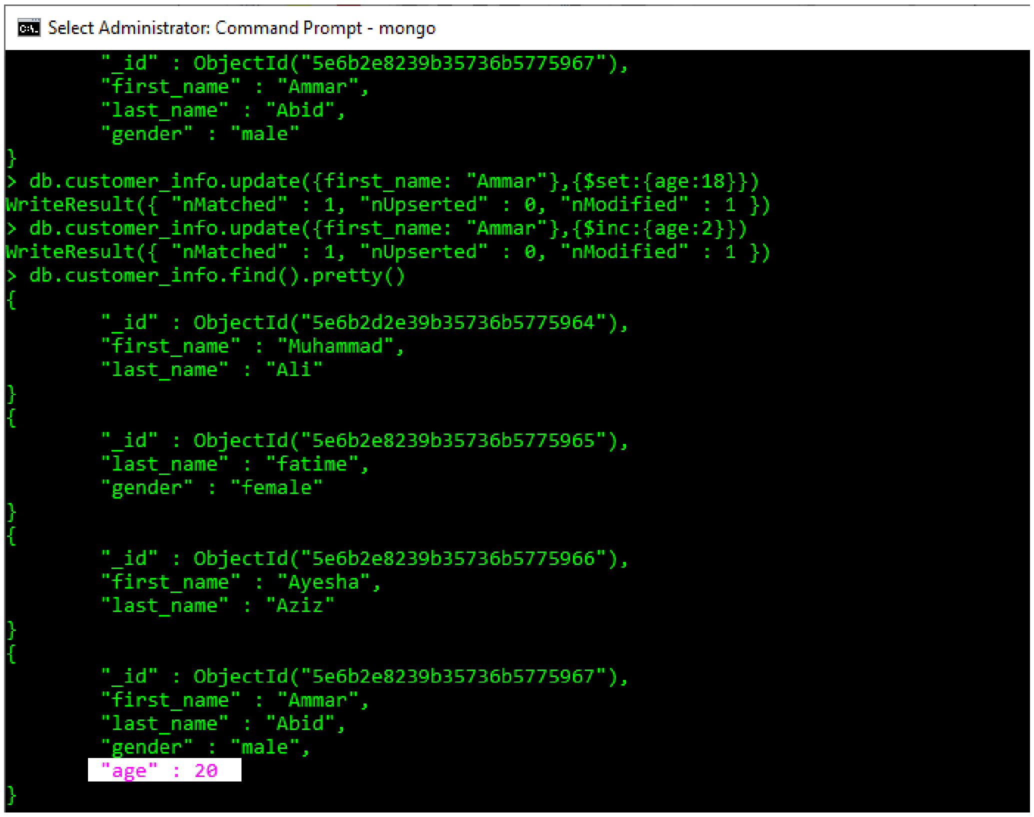


```
}
> db.customer_info.update({first_name: "Ammar"},{$set:{age:18}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> _
```

inc

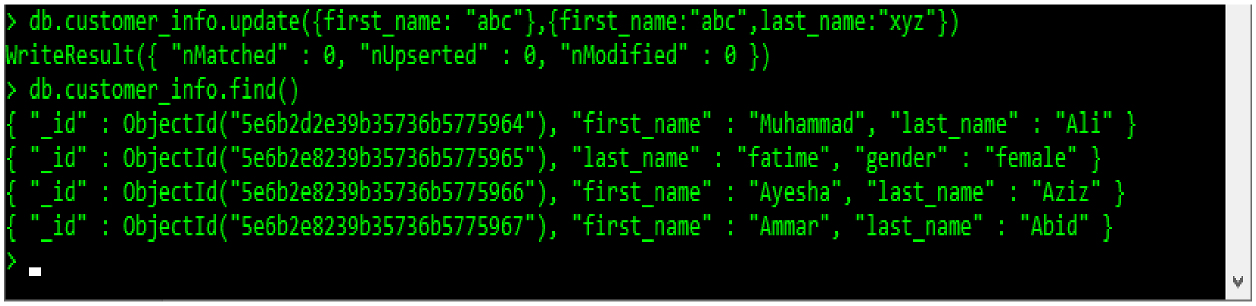
An operator inc is used to increment the value

```
db.customer_info.update({first_name: "Ammar"},{$inc:{age:2}});
```


Select Administrator: Command Prompt - mongo

```
        "_id" : ObjectId("5e6b2e8239b35736b5775967"),
        "first_name" : "Ammar",
        "last_name" : "Abid",
        "gender" : "male"
}
> db.customer_info.update({first_name: "Ammar"},{$set:{age:18}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.customer_info.update({first_name: "Ammar"},{$inc:{age:2}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.customer_info.find().pretty()
{
        "_id" : ObjectId("5e6b2d2e39b35736b5775964"),
        "first_name" : "Muhammad",
        "last_name" : "Ali"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775965"),
        "last_name" : "fatime",
        "gender" : "female"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775966"),
        "first_name" : "Ayesha",
        "last_name" : "Aziz"
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775967"),
        "first_name" : "Ammar",
        "last_name" : "Abid",
        "gender" : "male",
        "age" : 20
}
```

If we want to update a record which don't exists than what happens??

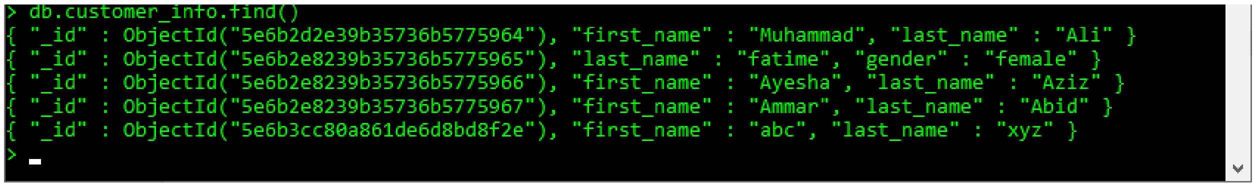


```
> db.customer_info.update({first_name: "abc"},{first_name:"abc",last_name:"xyz"})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.customer_info.find()
{ "_id" : ObjectId("5e6b2d2e39b35736b5775964"), "first_name" : "Muhammad", "last_name" : "Ali" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775965"), "last_name" : "fatime", "gender" : "female" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775966"), "first_name" : "Ayesha", "last_name" : "Aziz" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775967"), "first_name" : "Ammar", "last_name" : "Abid" }
>
```

Upsert

Let's say we want insert the data if not exists. For that we have upsert

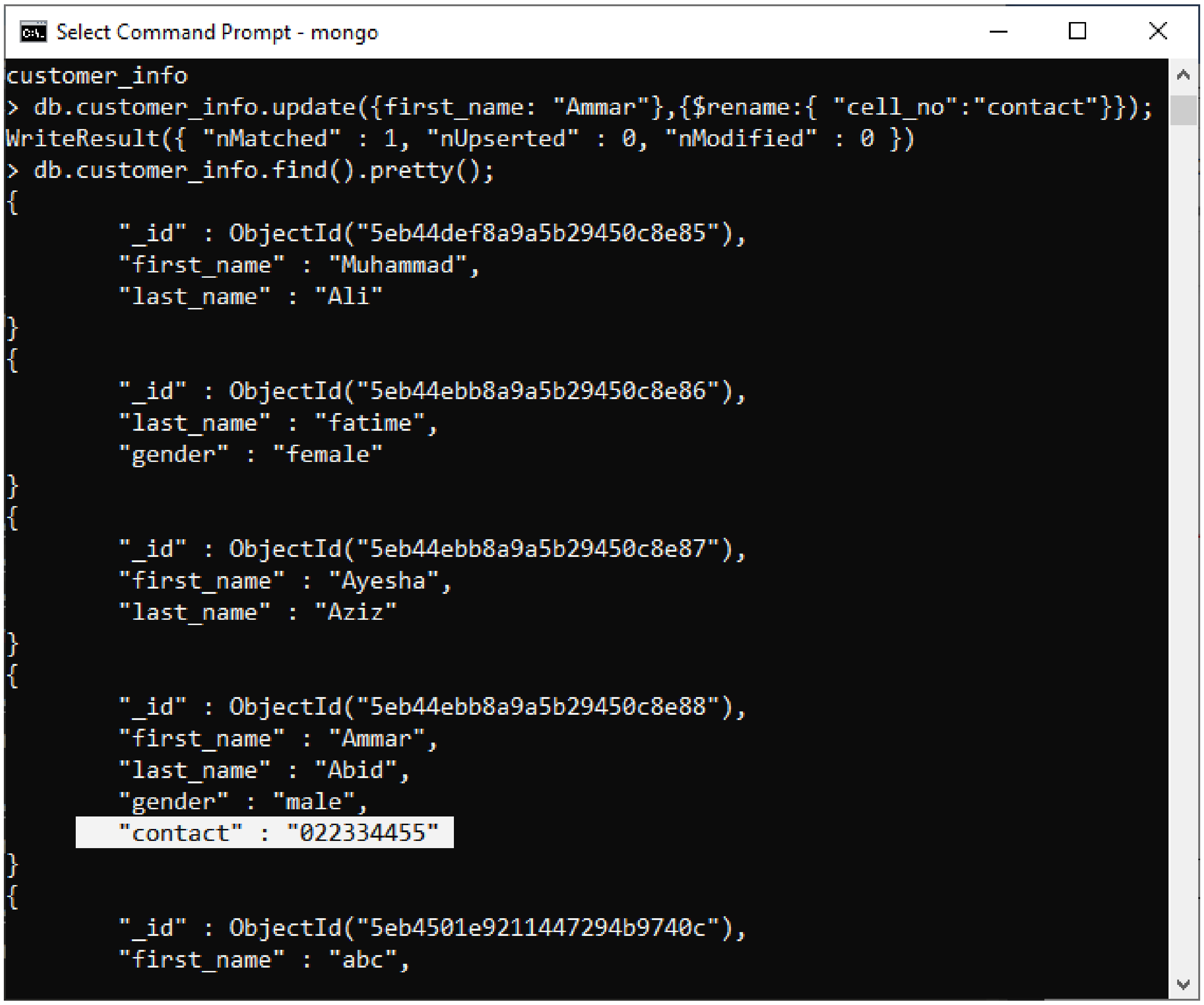
```
db.customer_info.update({first_name: "abc"},{first_name:"abc",last_name:"xyz"},{upsert: true})
```



```
> db.customer_info.find()
{ "_id" : ObjectId("5e6b2d2e39b35736b5775964"), "first_name" : "Muhammad", "last_name" : "Ali" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775965"), "last_name" : "fatime", "gender" : "female" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775966"), "first_name" : "Ayesha", "last_name" : "Aziz" }
{ "_id" : ObjectId("5e6b2e8239b35736b5775967"), "first_name" : "Ammar", "last_name" : "Abid" }
{ "_id" : ObjectId("5e6b3cc80a861de6d8bd8f2e"), "first_name" : "abc", "last_name" : "xyz" }
>
```

Rename a field

db.customer_info.update({first_name: "Ammar"},{$rename:{ "cell_no":"contact"}});



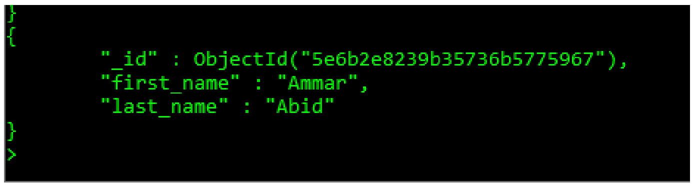
```
customer_info
> db.customer_info.update({first_name: "Ammar"},{$rename:{ "cell_no":"contact"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })
> db.customer_info.find().pretty();
{
        "_id" : ObjectId("5eb44def8a9a5b29450c8e85"),
        "first_name" : "Muhammad",
        "last_name" : "Ali"
}
{
        "_id" : ObjectId("5eb44ebb8a9a5b29450c8e86"),
        "last_name" : "fatime",
        "gender" : "female"
}
{
        "_id" : ObjectId("5eb44ebb8a9a5b29450c8e87"),
        "first_name" : "Ayesha",
        "last_name" : "Aziz"
}
{
        "_id" : ObjectId("5eb44ebb8a9a5b29450c8e88"),
        "first_name" : "Ammar",
        "last_name" : "Abid",
        "gender" : "male",
        "contact" : "022334455"
}
{
        "_id" : ObjectId("5eb4501e9211447294b9740c"),
        "first_name" : "abc",
```

Delete

Unset(Remove a field)

Here we are removing age from the record having first_name=”ammar”

```
db.customer_info.update({first_name: "Ammar"},{$unset:{age:1}})
```



```
}
{
        "_id" : ObjectId("5e6b2e8239b35736b5775967"),
        "first_name" : "Ammar",
        "last_name" : "Abid"
}
>
```

Unset(Remove a collection)

```
db.customer_info.remove({first_name: "Rida"});
```

how to handle foreign ley (embed documents), and , or , greater than , less than operators see the video.

https://www.youtube.com/watch?v=-56x56UppqQ&t=1246s [watch this video from 22:05]