Getting started

ITMC322

Installing MongoDB

- Visit https://www.mongodb.com/
- Install the community version
- Accompanied with MongoDB Compass, GUI for the mongo DB

Mongo DB GUI

- Robo 3T → Studio 3T
- Default with MongoDB download → MongoDB compass
- MongoDB shell in Visual studio Code
- You can use any of the following software, including the cmd

Installation options

- Community server
- Visual studio Extension
- MongoDB Atlas
- MongoDB Realm

Releases

- Current release → the latest stable release for production. - 5.0.9
- Development → release that are not for production, contain additional new features. Not stable -6.0.0-rc8
- Archived release → the last working stable release - 5.0.8

MongoDB

- Commands
- Queries (APIs) → Document queries

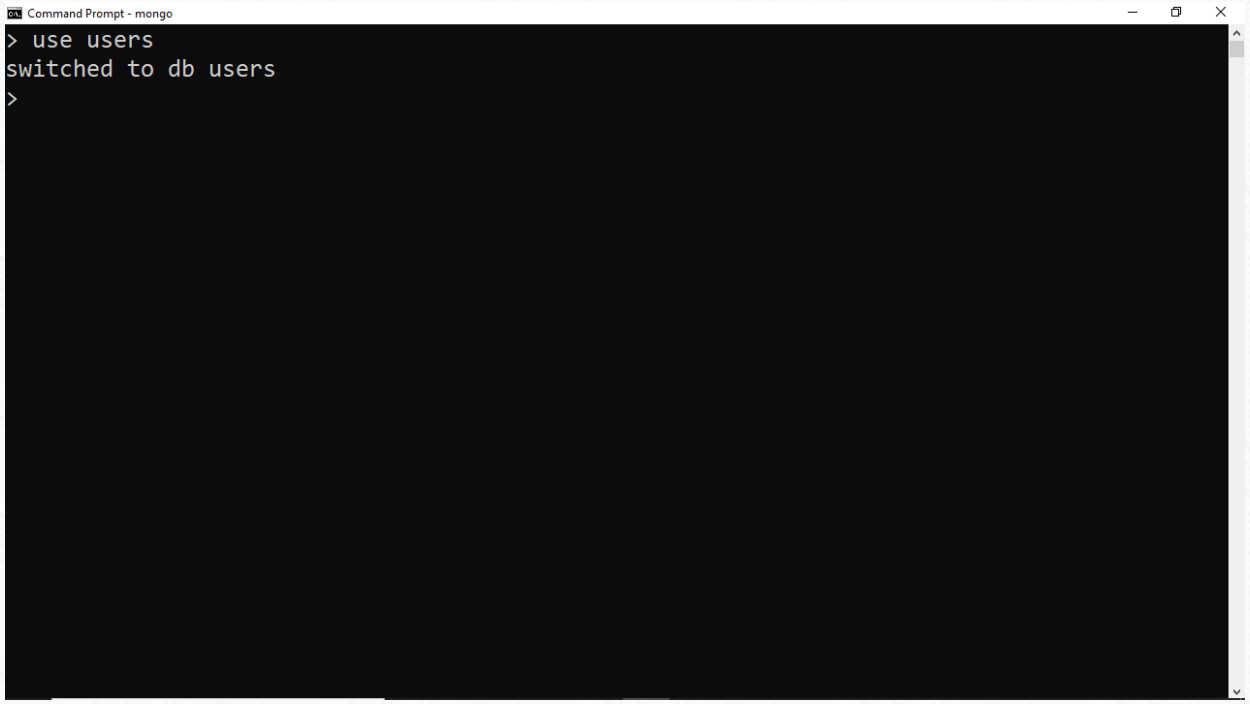
Creating a DB

- MongoDB has no "create" command for creating a database.
- it is optional to create a database manually, 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

- **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, if it does not exist.

Use command



Data type in MongoDB

- String
- Integer
- Boolean
- Double
- Min/Max keys→ used for comparison
- Array → [“xxx”:“cccc”]

Data type in MongoDB

- Timestamps →created_at, updated_at, deleted_at
- Object
- Null
- Symbol → identical to the use of strings, reserved for languages that use specific symbol type.
- Date
- Binary Data

Data type in MongoDB

- Object ID→ used to store document ID
- Code → used to store JS code
- Regex → used to store regular expressions

Show databases

- You can list all databases on a mongo server, using either of the following commands
- Show dbs
- Show databases
- If a database is empty it will not show in the list of databases

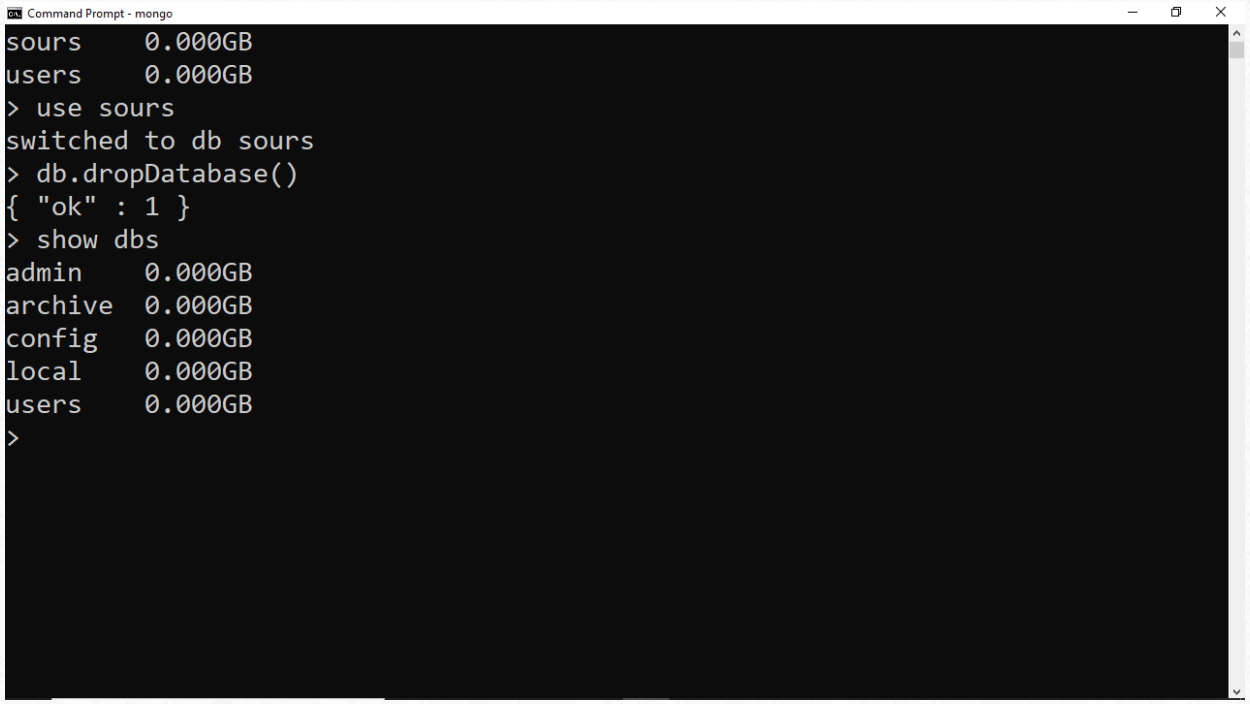
Show dbs


Command Prompt - mongo
> show dbs
admin   0.000GB
archive 0.000GB
config   0.000GB
local   0.000GB
sours   0.000GB
users   0.000GB
> show databases
admin   0.000GB
archive 0.000GB
config   0.000GB
local   0.000GB
sours   0.000GB
users   0.000GB
>

Drop a database

- To check the database you are on, you use the following command
  - db
- To drop a DB you are on
  - db.dropDatabase()

Drop database



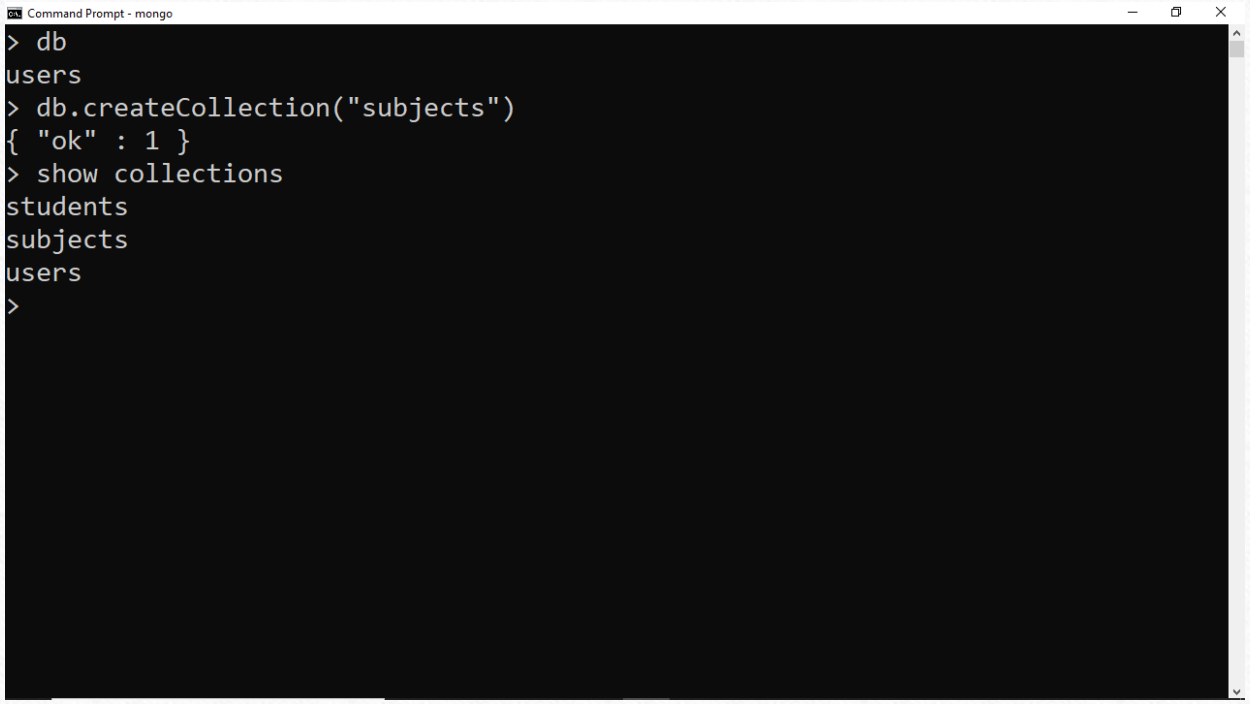
Creating a collection

- Creation of collection can be done using **db.createCollection(name, options)**
- **Db.createCollections("collection_name")**
- Or you can create a collection by using the insert function on the collection name

Display collections

- Make sure to select a DB first using
  - db
- Then use the command
  - show collections

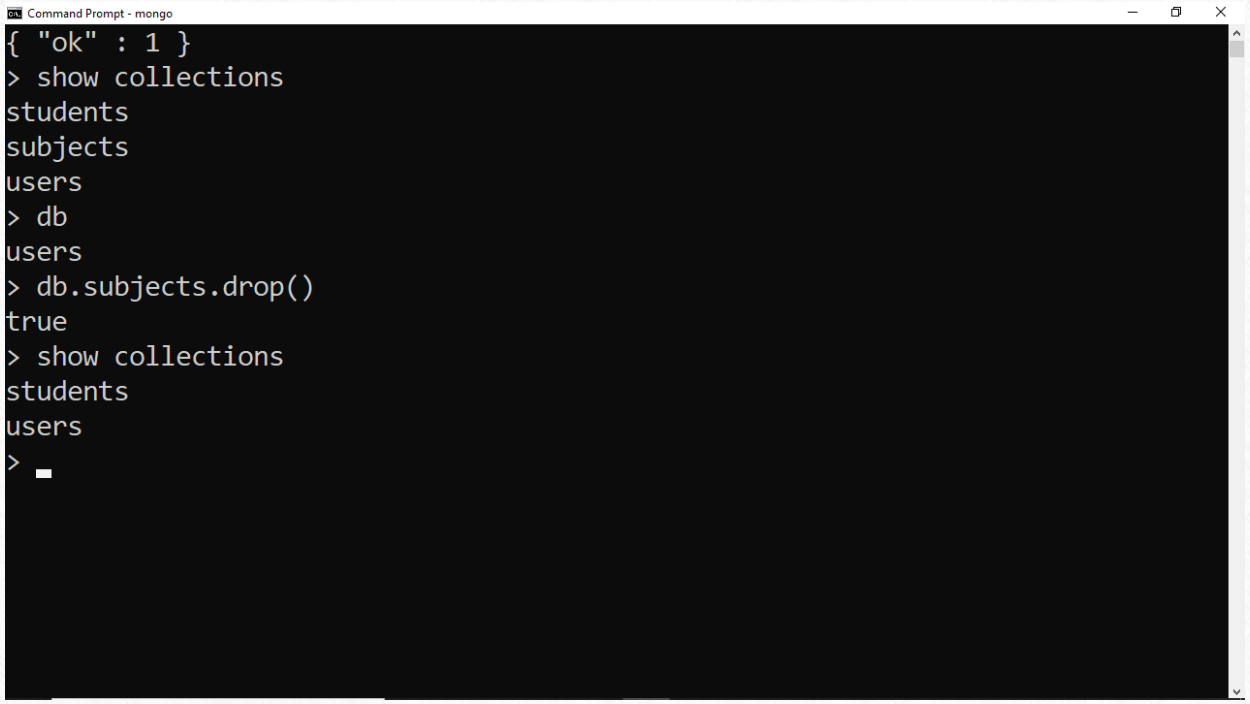
Create / display collections



Drop collections

- Make sure you are on the database base you want to remove the collection from, then type the following command
  - db.collection.drop() → collection = collection name

Drop a collection



Insert into Collection

- Insert
- insertOne → insert only one document
- insertMany → insert an array of documents

Insert into collection

- On your current db use the insertOne or insetMany
- Db.collection_name.insertOne({
- “name”: “Ahmed”
- })
- Db.collection_name.insertMany([document 1, document 2, …..])

insertOne


Command Prompt - mongo
> use users
switched to db users
> show collections
students
users
> db.users.insertOne({"name": "Ahmed", "email":"ahmed@gmail.com"})
{
"acknowledged" : true,
"insertedId" : ObjectId("62b94f946c786fd5dadef6d8")
}
>

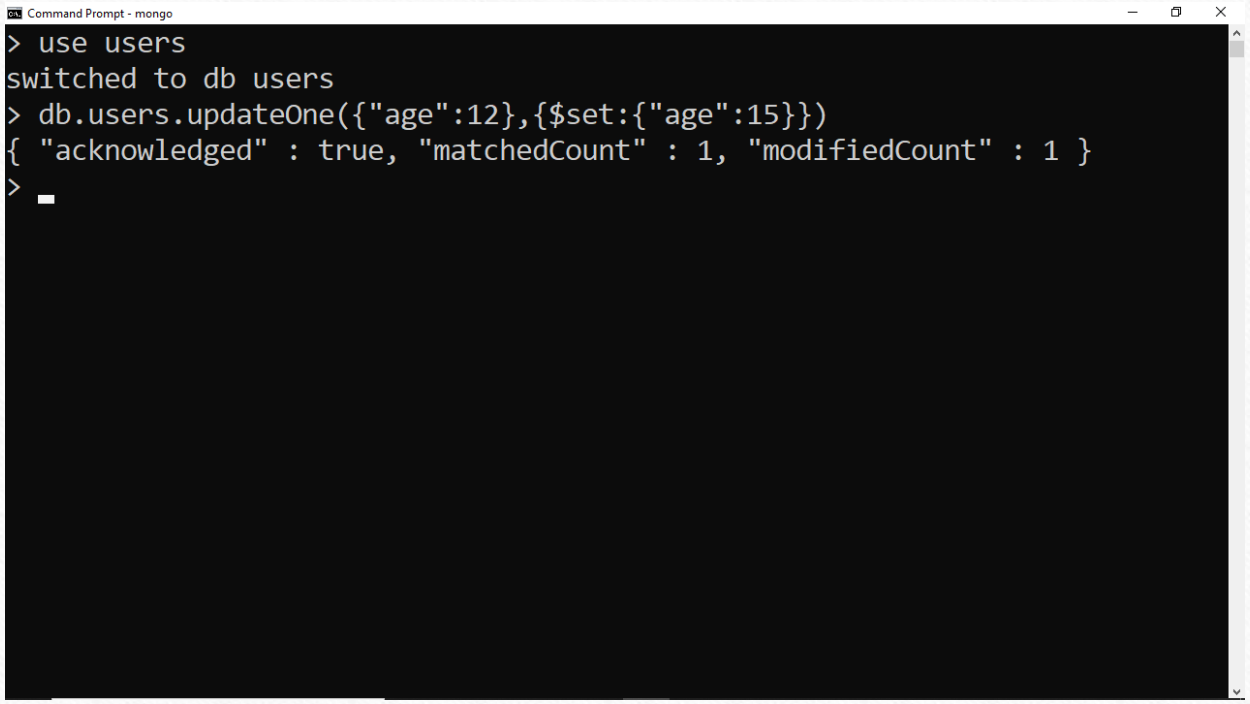
insertMany

```
}
> db.users.insertMany([{"name": "Ali", "address": "Tripoli"}, {"name": "Asma", "phone": "09222999"}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("62b94ff06c786fd5dadef6d9"),
                ObjectId("62b94ff06c786fd5dadef6da")
        ]
}
>
```

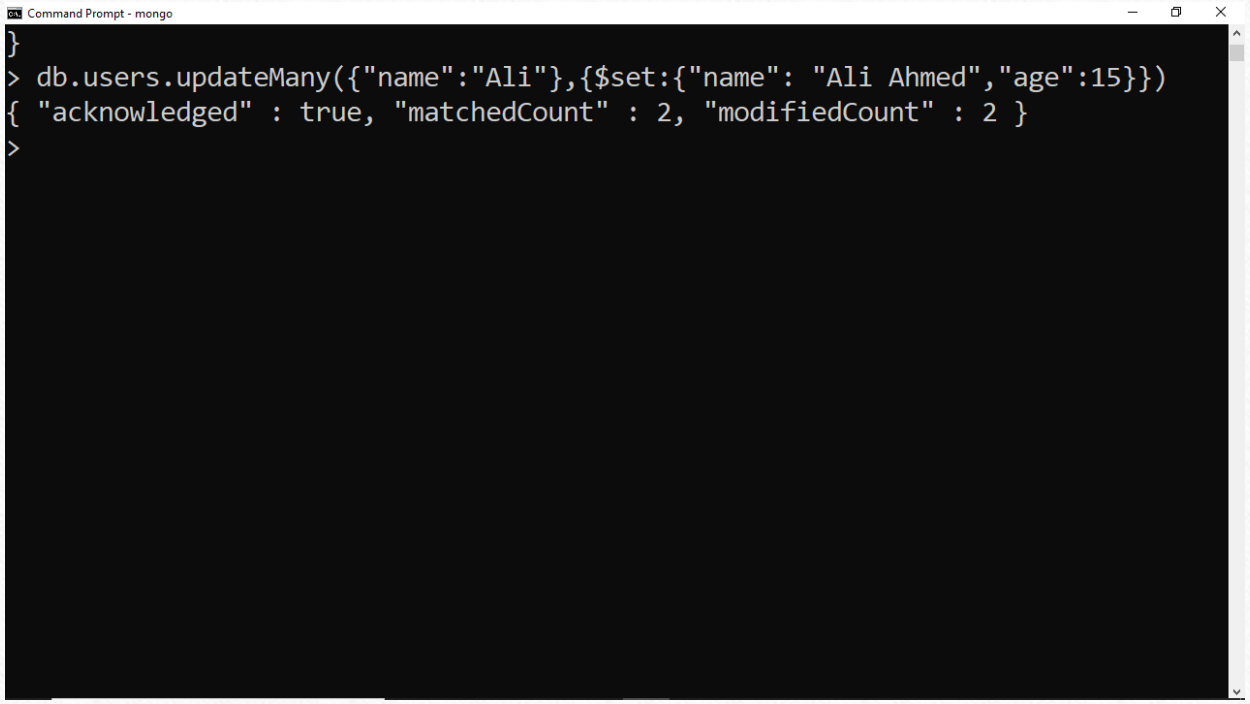
Update a document

- Function Update
- db.collection_name.updateOne(<filter>, <update>)
  - db.users.update({“id”: “8”},{$set:
  - { “name”: “Mohammed”, “mobile”: “09768888”}})
- updateMany()

updateOne


Command Prompt - mongo
> use users
switched to db users
> db.users.updateOne({"age":12},{$set:{"age":15}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
>

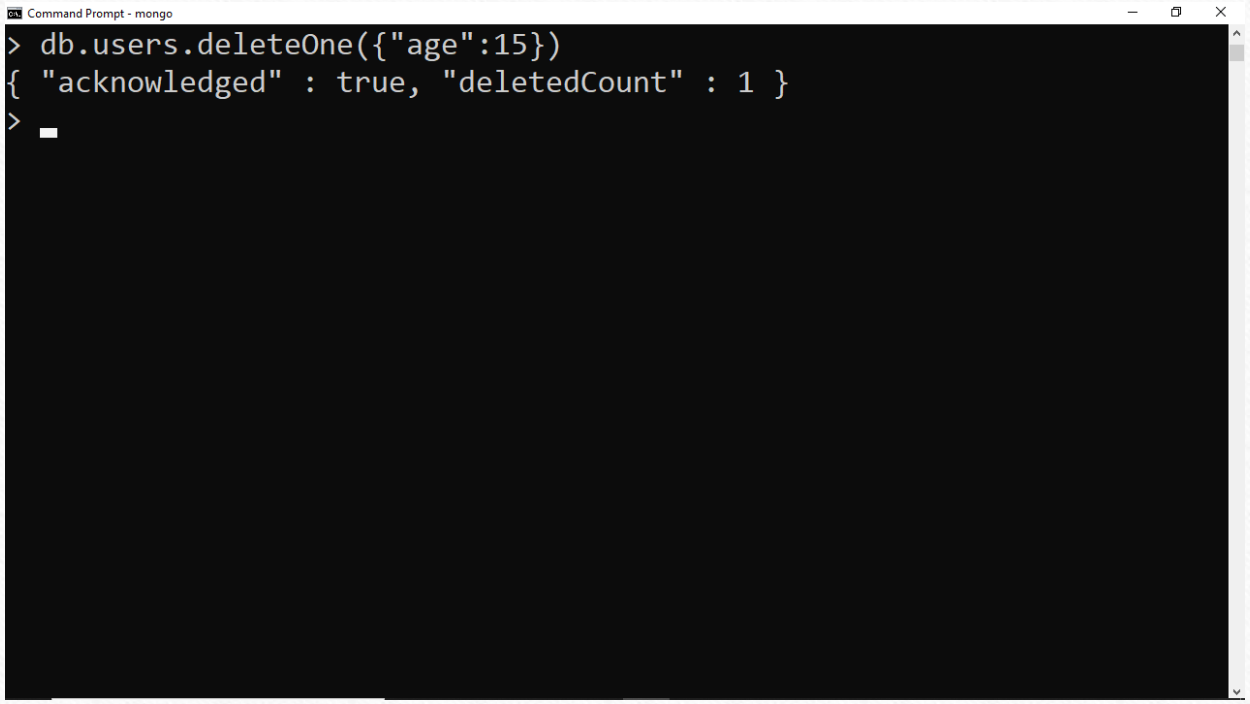
updateMany


Command Prompt - mongo
}
> db.users.updateMany({"name":"Ali"},{$set:{"name": "Ali Ahmed","age":15}})
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
>

Delete from collection

- db.collection.deleteMany({}) – when empty deletes all documents in collection
- Db.collection.deleteOne({condition})

deleteOne


Command Prompt - mongo
> db.users.deleteOne({"age":15})
{ "acknowledged" : true, "deletedCount" : 1 }
>

deleteMany


> db.users.deleteMany({"name":"Ali Ahmed"})
{ "acknowledged" : true, "deletedCount" : 2 }
>

find

- There are many uses to the find function in mongodb
- It can be used to display the documents in a collection
  - db.collection_name.find()
- Display collection data in a more visual(pretty) display
  - db.collection_name.find().pretty()

Find()


Command Prompt - mongo
> db.users.find()
{ "_id" : ObjectId("62b6de7449a9f89dc6c5e203"), "name" : "dddd", "a
{ "_id" : ObjectId("62b6e04849a9f89dc6c5e205"), "name" : "ali", "em
{ "_id" : ObjectId("62b94f946c786fd5dadef6d8"), "name" : "Ahmed", "
{ "_id" : ObjectId("62b94ff06c786fd5dadef6da"), "name" : "Asma", "p
> db.users.find().pretty()
{
"_id" : ObjectId("62b6de7449a9f89dc6c5e203"),
"name" : "dddd",
"address" : "with",
"class" : "jjjjj"
}
{
"_id" : ObjectId("62b6e04849a9f89dc6c5e205"),
"name" : "ali",
"emails" : [
{

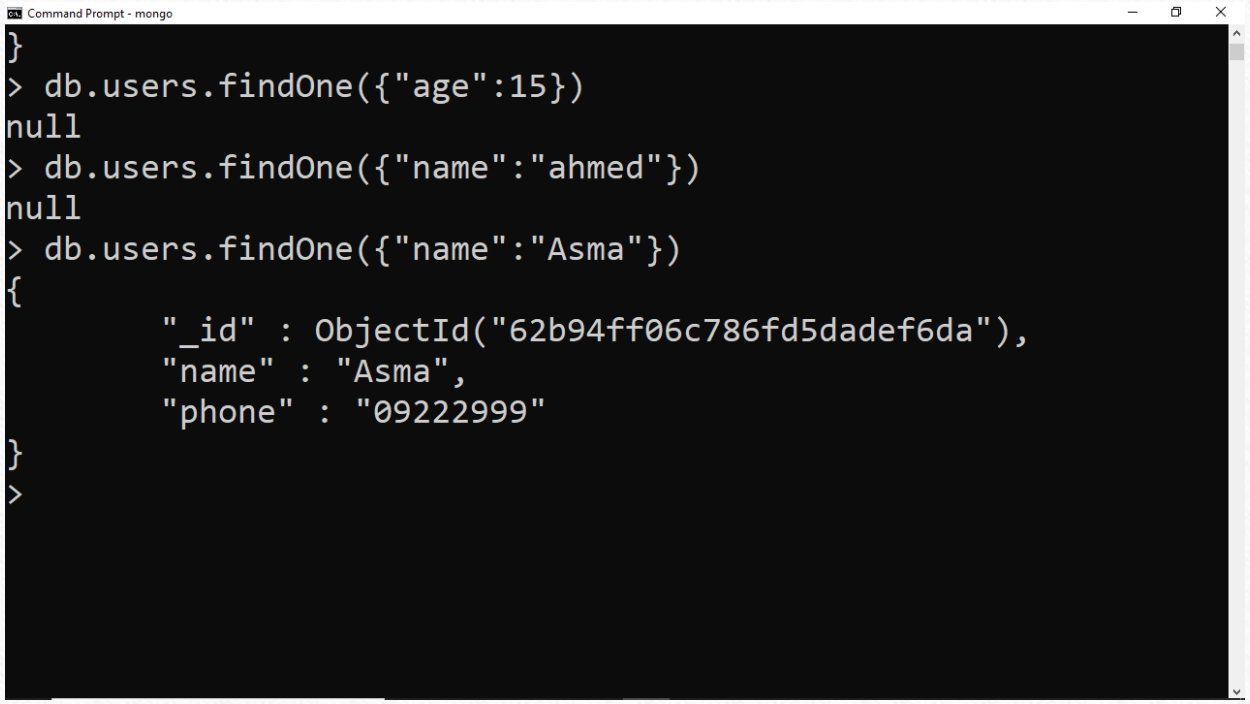
findOne

- Display the first document in the collection

findOne

```
> db.users.findOne()
{
        "_id" : ObjectId("62b6de7449a9f89dc6c5e203"),
        "name" : "dddd",
        "address" : "with",
        "class" : "jjjjj"
}
>
```

findOne with a query



Find with a query

- In the same way you can use find with a query, but find returns any result the matches the query, however findOne returns the first result that matches the query

Find vs. findOne

Find One and Delete

- Like the name suggests this function is used to find a document, display it to the user and then deletes
- It finds and deletes the first document that matches the filter/query


Command Prompt - mongo
}
> db.users.findOneAndDelete({"name":"Asma"})
{
"_id" : ObjectId("62b94ff06c786fd5dadef6da"),
"name" : "Asma",
"phone" : "09222999"
}
> db.users.find()
{ "_id" : ObjectId("62b6de7449a9f89dc6c5e203"), "name" : "dddd"
{ "_id" : ObjectId("62b6e04849a9f89dc6c5e205"), "name" : "ali",
{ "_id" : ObjectId("62b94f946c786fd5dadef6d8"), "name" : "Ahmed
{ "_id" : ObjectId("62b9551b9397b25c00081155"), "name" : "Asma"
>

Other commands

- db.version()→ check version of db
- Db.stats()→ get db stats