DIFFERENT COMMANDS AND THEIR USAGE

- Use <database name> - To create a database
-
```
test> use vaibhav
switched to db vaibhav
```
- db.createCollection("<collection name>") – To create a collection
```
vaibhav> db.createCollection("student")
{ ok: 1 }
```
- db.<collection name>.insertOne({}) – to insert data inside a collection
```
vaibhav> db.student.insertOne({"Name":"Vaibhav","University Roll no.":"201500764","
Class Roll no.":"69","Contact no.":"7078882110","Subject":["FUll Stack using NodeJs
","Machine Learning","Design and Analysis of Algorithm"],"Hobbies":["Music","Coding
","Reading"]});
{
  acknowledged: true,
  insertedId: ObjectId("63be772177378215651fa1a6")
}
```
- db.<collection name>.insertMany([{},{},{}]) – to insert many records at a time
```
vaibhav> db.student.insertMany([{"Name":"ABC","ROll no":1},{"Name":"xyz","Roll no":
2},{"Name":"PQR","Roll no":3}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63be7b4277378215651fa1a7"),
    '1': ObjectId("63be7b4277378215651fa1a8"),
    '2': ObjectId("63be7b4277378215651fa1a9")
  }
}
```
- db – it shows in which database we are working
```
vaibhav> db
vaibhav
```
- show collection/db.getCollectionName – to get all the collection inside a databse
```
vaibhav> show collections
student
```
- db.<collection name>.find() – to find all the data stored in the collection

```
vaibhav> db.student.find()
[
  {
    _id: ObjectId("63be772177378215651fa1a6"),
    Name: 'Vaibhav',
    'University Roll no.': '201500764',
    'Class Roll no.': '69',
    'Contact no.': '7078882110',
    Subject: [
      'FUll Stack using NodeJs',
      'Machine Learning',
      'Design and Analysis of Algorithm'
    ],
    Hobbies: [ 'Music', 'Coding', 'Reading' ]
  }
]
```

- db.<collection name>.find({“key”:”value”})

```
vaibhav> db.student.find({"Name":"Vaibhav"})
[
  {
    _id: ObjectId("63be772177378215651fa1a6"),
    Name: 'Vaibhav',
    'University Roll no.': '201500764',
    'Class Roll no.': '69',
    'Contact no.': '7078882110',
    Subject: [
      'FUll Stack using NodeJs',
      'Machine Learning',
      'Design and Analysis of Algorithm'
    ],
    Hobbies: [ 'Music', 'Coding', 'Reading' ]
  }
]
```

This command will check the exact value of key and pair (case sensitive)

When it is not able to find any such data then it will print a blank line

```
vaibhav> db.student.find({"Name":"vaibhav"})

vaibhav> db.student.find({"name":"Vaibhav"})

vaibhav>
```

In case there are many keys with the same value than it will show all the keys having the same values

When we use find , it gives all the data related to the key like on finding tha name it gives all the data associated with the name.
So as to tackle this and to check only that key value exist other than all the value we use find with projection.

- db.testdata.find({key:value},{key:1}}

```
testdb> db.testdata.find({name:"Html"},{name:1});
[ { _id: ObjectId("63cf4d4ac053c33b835b1e3f"), name: 'Html' } ]
```

If we don't want even the id of it we will use

```
testdb> db.testdata.find({name:"Html"},{_id:0,name:1});
[ { name: 'Html' } ]
```

- To find only one document without using findone we will use

```
testdb> db.testdata.find({active:true}).limit(1);
[
  {
    _id: ObjectId("63cf4d4ac053c33b835b1e3c"),
    name: 'ReactJS',
    type: 'Front End',
    learning_hour: 40,
    active: true
  }
]
```

- To find one document but not the first one we will use skip

```
testdb> db.testdata.find({active:true}).limit(1).skip(1);
[
  {
    _id: ObjectId("63cf4d4ac053c33b835b1e3d"),
    name: 'MongoDB',
    type: 'Back End',
    learning_hour: 38,
    active: true
  }
]
```

We can change the value of skip to skip the first x found document

For printing only one we will use

- db.<collectionname>.findOne({“key” : ”value”})
  pretty doesn’t work with findone

- db.<collection name>.updateOne({“key1” : “value1” , {$set:{“key2” : “value2”}});

  this command will check which data has key and value exact similar to the key1 and value1. If it finds it will update the value of key2 by value2. In case there exist no key2 than it will add new entry in the collection

  On inserting unmatched key2 it will create a new entry

```
vaibhav> db.student.updateOne({"Roll no":2},{$set:{"name":"XYZ"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
vaibhav> db.student.find()
[
  {
    _id: ObjectId("63be772177378215651fa1a6"),
    Name: 'Vaibhav',
    'University Roll no.': '201500764',
    'Class Roll no.': '69',
    'Contact no.': '7078882110',
    Subject: [
      'FUll Stack using NodeJs',
      'Machine Learning',
      'Design and Analysis of Algorithm'
    ],
    Hobbies: [ 'Music', 'Coding', 'Reading' ]
  },
  {
    _id: ObjectId("63be7b4277378215651fa1a7"),
    Name: 'ABC',
    'ROll no': 1
  },
  {
    _id: ObjectId("63be7b4277378215651fa1a8"),
    Name: 'xyz',
    'Roll no': 2,
    name: 'XYZ'
  },
  {
    _id: ObjectId("63be7b4277378215651fa1a9"),
    Name: 'PQR',
    'Roll no': 3
  }
]
```

On inserting correct value of key2

```
vaibhav> db.student.updateOne({"Roll no":2},{$set:{"Name":"WXYZ"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

- db.countDocument/db.estimatedDoucmentCount

```
vaibhav> db.student.find()
[
  {
    _id: ObjectId("63be772177378215651fa1a6"),
    Name: 'Vaibhav',
    'University Roll no.': '201500764',
    'Class Roll no.': '69',
    'Contact no.': '7078882110',
    Subject: [
      'FUll Stack using NodeJs',
      'Machine Learning',
      'Design and Analysis of Algorithm'
    ],
    Hobbies: [ 'Music', 'Coding', 'Reading' ]
  },
  {
    _id: ObjectId("63be7b4277378215651fa1a7"),
    Name: 'ABC',
    'ROll no': 1
  },
  {
    _id: ObjectId("63be7b4277378215651fa1a8"),
    Name: 'WXYZ',
    'Roll no': 2,
    name: 'XYZ'
  },
  {
    _id: ObjectId("63be7b4277378215651fa1a9"),
    Name: 'PQR',
    'Roll no': 3
  }
]
```

Operators in MongoDB
To retirieve the document with exact id_value ‘L0009100’
Db.inventory.find({“_id”:{$eq :”L0009100”}})

We make user to perform different task to various users
There are different types of user

1) read
2) write
3) dbAdmin
4) dbowner
5) userAdmin

Now to create any user we will use this command
db.createUser({name,password,role})

db.createUser({user:”xyz”,pwd:”pass@123”,roles:[role:”read”,db:”supermarket”]})
where supermarket is the name of the database

Now using above approach password is visible therefore we will use this command
db.createUser({user:”xyz”,pwd:passwordPrompt(),roles:[role:”read”,db:”supermarket”]})
using this on hitting enter it will ask for password in astrik

Query Operators

Comparison Operators
$eq - equal to
$gt - greater than

$gte - greater than or equal to
$lt - less than
$lte - less than or equal to
$ne - not equal to

Logical Operators
$and - logical and
$not - logical not
$or - logical or

Element Operators
$exists - exists
$type - type

Existence Operators
$in - in
$nin - not in

Mongo DB Queries

Basic Queries
show dbs // show all databases
use <db_name> // use database
db.dropDatabase() // to delete database (must present in database)

Create, show & delete collections
db.createCollection("<collection_name>") // create collection
show collections // show all collections
db.getCollectionNames() // show all collections
db.<collection_name>.drop() // delete collection

Inserting Documents
db.<collection_name>.insertOne({<document>}) // insert one document
db.<collection_name>.insertMany([{<document>}, {<document>}, ...]) // insert many documents

Deleting Documents
db.<collection_name>.deleteOne({<document}>) // delete one document
db.<collection_name>.deleteMany({<document>}) // delete many documents

Updating Documents
db.<collection_name>.updateOne({<document>} //to find , {//operator : {<document>}}) // update one document
db.<collection_name>.updateMany([{<document>}, {<document>}]) // update many documents

Count
db.<collection_name>.documentCount() // count all documents
db.<collection_name>.documentCount({<document>}) // count all documents having matched key value pair
db.<collection_name>.estimatedDocumentCount()

To import json database
mongoimport --jsonArray --db supermarket --collection product --file "C:\Users\Khushal_agarwal\Downloads\pro.json"

Retrieve Data

db.<collection_name>.find() // show all documents in collection
db.<collection_name>.find({key:value}) // all records having matched key value pair
db.<collection_name>.findOne({key:value}) // first record having matched key value pair
db.testdata.find({name:'Html'},{name:1}) // Show only name field
db.testdata.find({name:'Html'},{_id:0,name:1}) // _id also not shown
db.testdata.find({active:true}).limit(1) //find only one document
db.testdata.find({active:true}).limit(1).skip(2) // to skip any number of documents exist

User commands
db.createUser({user:"khush",pwd:passwordPrompt(),roles:[{role:"read",db:supermarket}]})

mongo installation
mongosh setup
mongo cli tools - export import
basic CRUD operation
18 general operators (regex,logical,array,arithmetic,indexes)
index - text index,ascending,descending,multi key
data modelling
authentication - adding a user with roles
administration - backup restore