```
{
  _id: "5cf0029caff5056591b0ce7d",
  firstname: 'Jane',
  lastname: 'Wu',
  address: {
    street: '1 Circle Rd',
    city: 'Los Angeles',
    state: 'CA',
    zip: '90404'
  }
}
```

Agenda :

MongoDB

- MongoDB installation
- MongoShell installation
- Set Environment variable PATH
- How to use Mongsh
- VSCode w/ Mongosh
- databases
- insert
- data types
- sorting and limiting
- find
- update
- delete
- comparison operators
- logical operators
- indexes
- collections

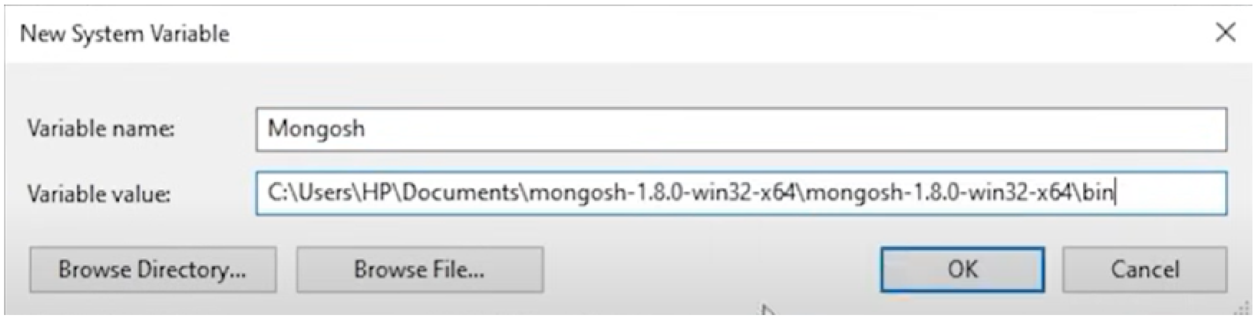
Its an Nosql database

Mongodb installation :
https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/

Then we need to install mongodb shell, inside the same document left corner there is an mongodb shell ..

https://www.mongodb.com/try/download/shell

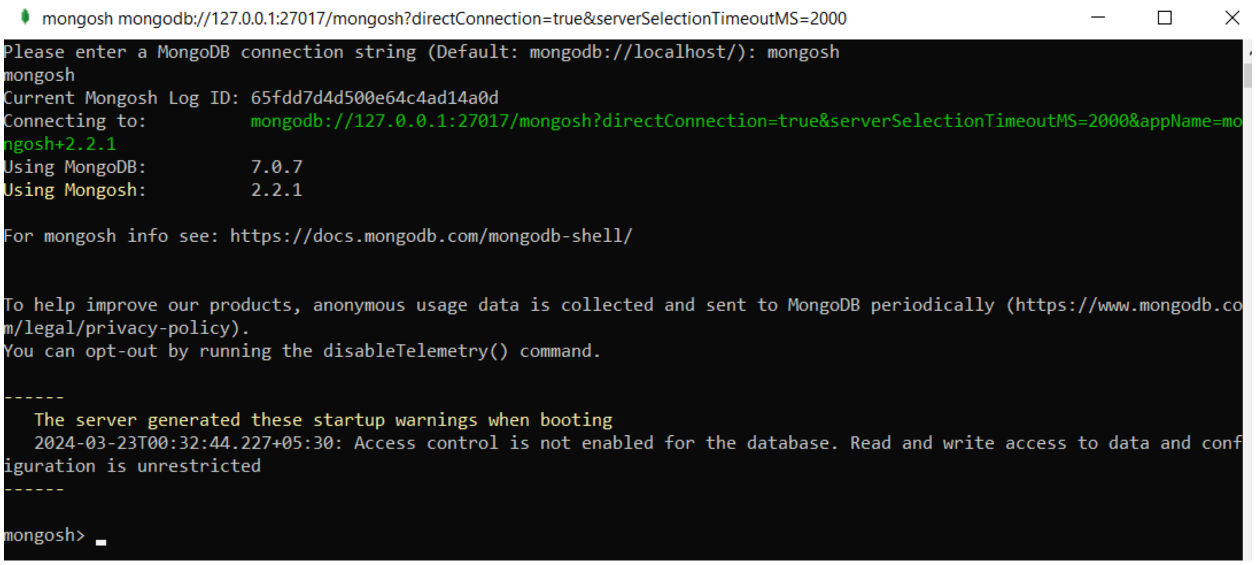
Open environment variable and give mongodb shell path..



Variable name: Mongosh

Variable value: C:\Users\HP\Documents\mongosh-1.8.0-win32-x64\mongosh-1.8.0-win32-x64\bin

After completion of env , make sure to run mangodb shell..
In monghosh.exe use this commands :



```
mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000
Please enter a MongoDB connection string (Default: mongodb://localhost/): mongosh
mongosh
Current Mongosh Log ID: 65fdd7d4d500e64c4ad14a0d
Connecting to:          mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000&appName=mo
ngosh+2.2.1
Using MongoDB:          7.0.7
Using Mongosh:          2.2.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.co
m/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2024-03-23T00:32:44.227+05:30: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
------

mongosh> 
```

To clear your screen type **cls then give exit**

I am just going to open in vscode

Install the mongodb extension there.. And click on mongodb icon in left corner and connect.

In connections -> right click then there is an option Launch mongodb shell.
Use exit command to terminate the running mongodb shell..
To start use mongosh
Cls command to clear

Here now we are going to study how to create & use databases in mongodb ..

Use show dbs cmd - wil give the list of all current databases.

To use the selective database then use <db name>

Eg : use Admin

If you are using database name that does not exist then it is going to create a new database.

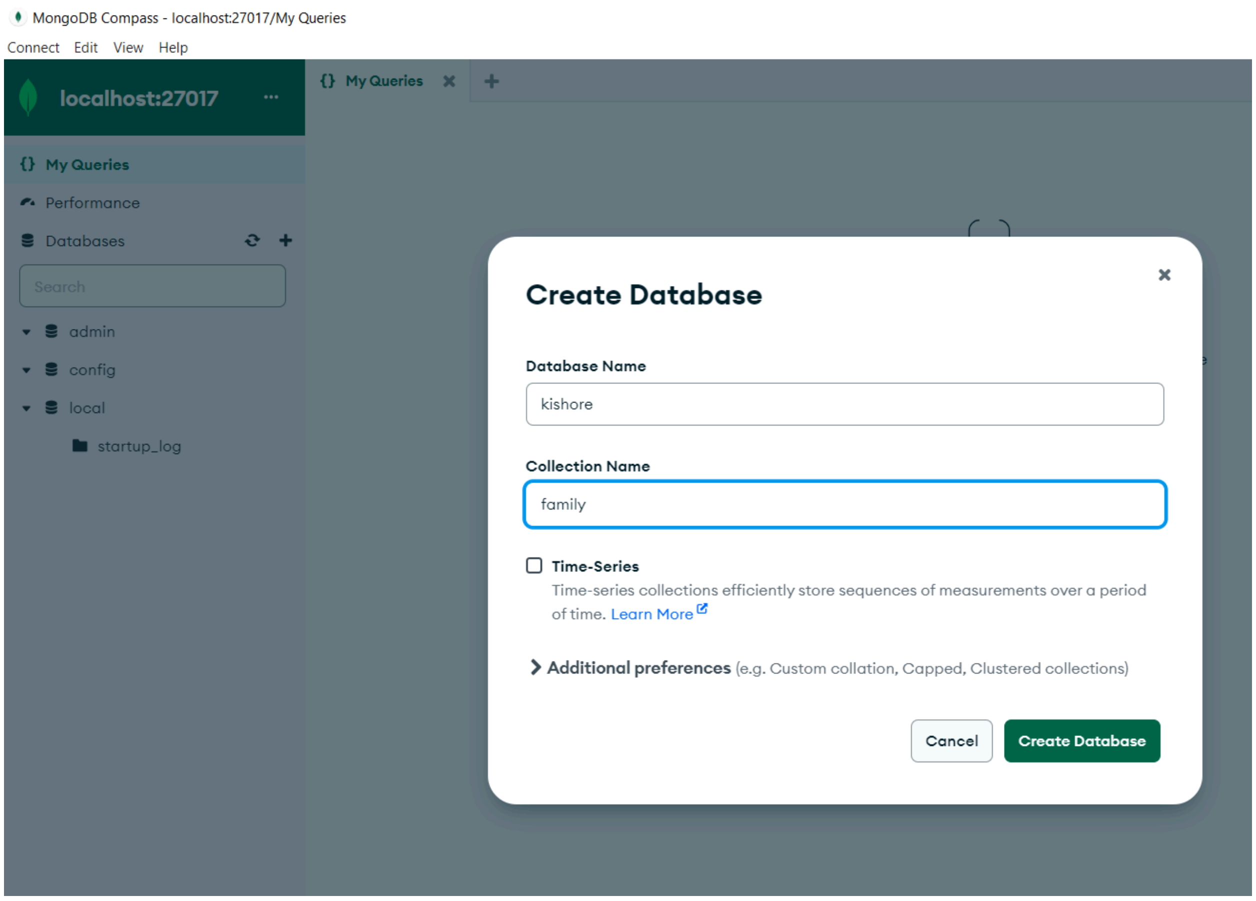
Use kishore — now kishore db is created.

Check using show dbs cmd..

Let’s add some collection to our kishore database (.) dot will be used to create collection method and it ends with parentheses ().
db.createCollection(“Family”) — here inside Family is an argument
Then clear the screen using cls
Then show dbs

I am going to drop the database using this command :
db.dropDatabase()

Then we going to mongodb compass tool :



Create Database

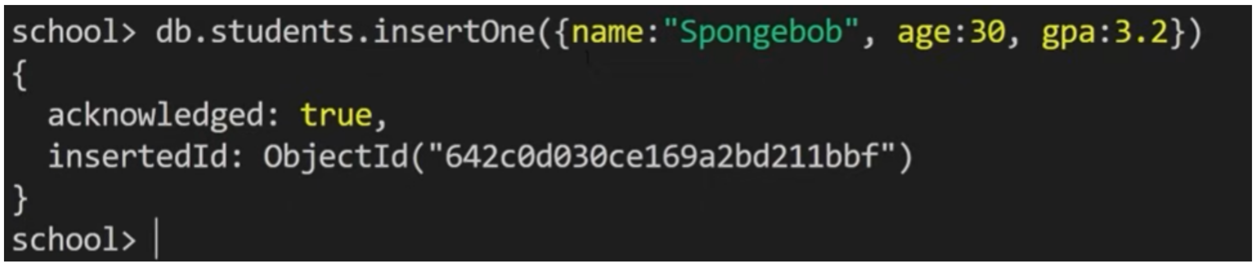
Database Name: kishore

Collection Name: family

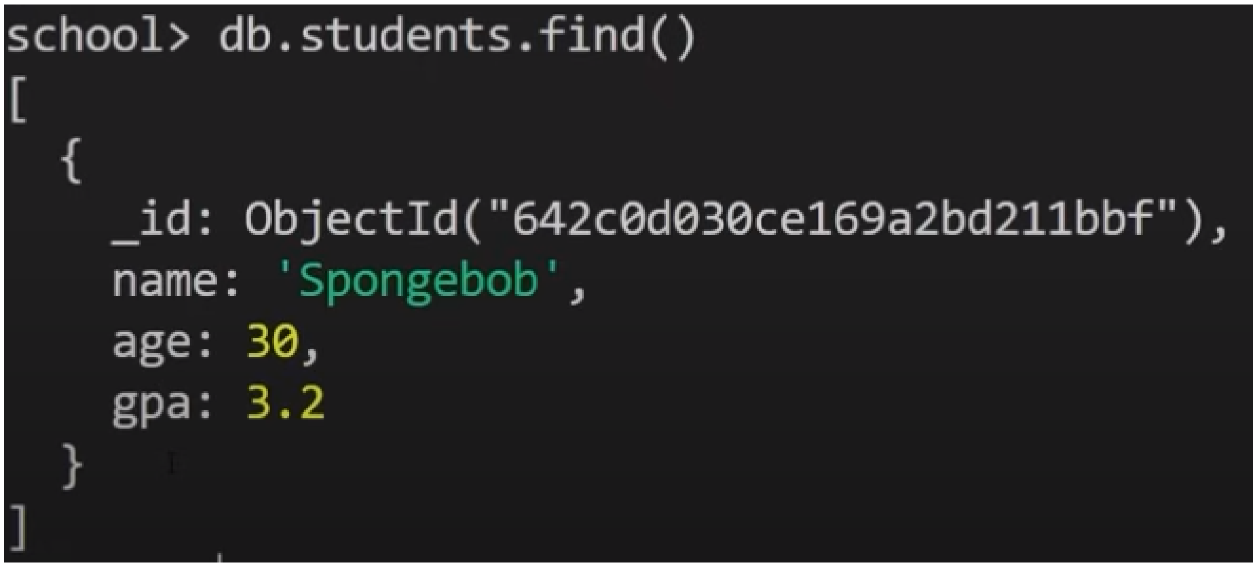
☐ Time-Series
Time-series collections efficiently store sequences of measurements over a period of time. Learn More

Additional preferences (e.g. Custom collation, Capped, Clustered collections)

Insert : - shell

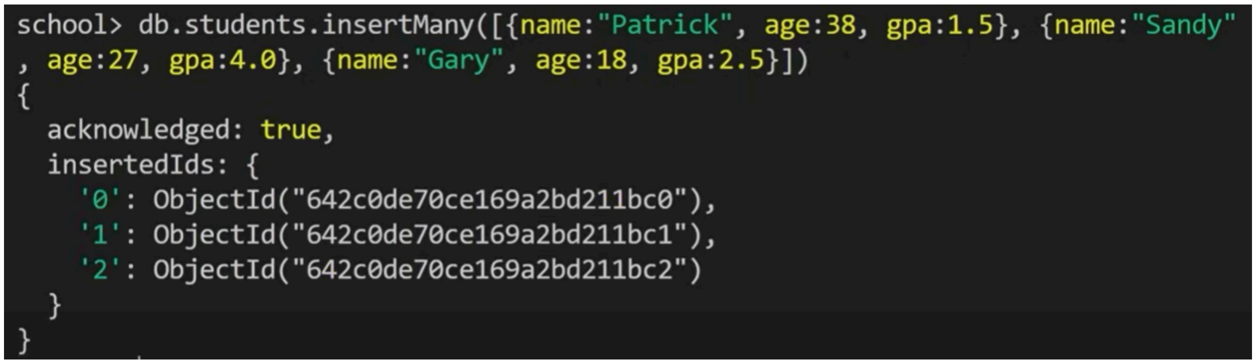


```
school> db.students.insertOne({name:"Spongebob", age:30, gpa:3.2})
{
  acknowledged: true,
  insertedId: ObjectId("642c0d030ce169a2bd211bbf")
}
school>
```



```
school> db.students.find()
[
  {
    _id: ObjectId("642c0d030ce169a2bd211bbf"),
    name: 'Spongebob',
    age: 30,
    gpa: 3.2
  }
]
```

Then clear the shell using cls command.

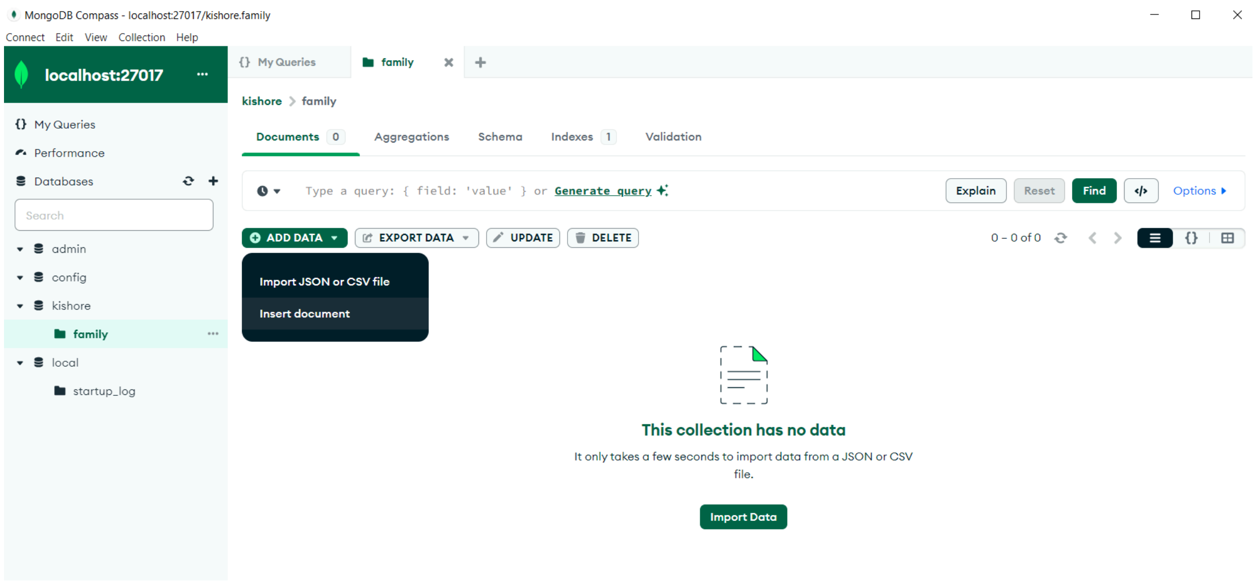


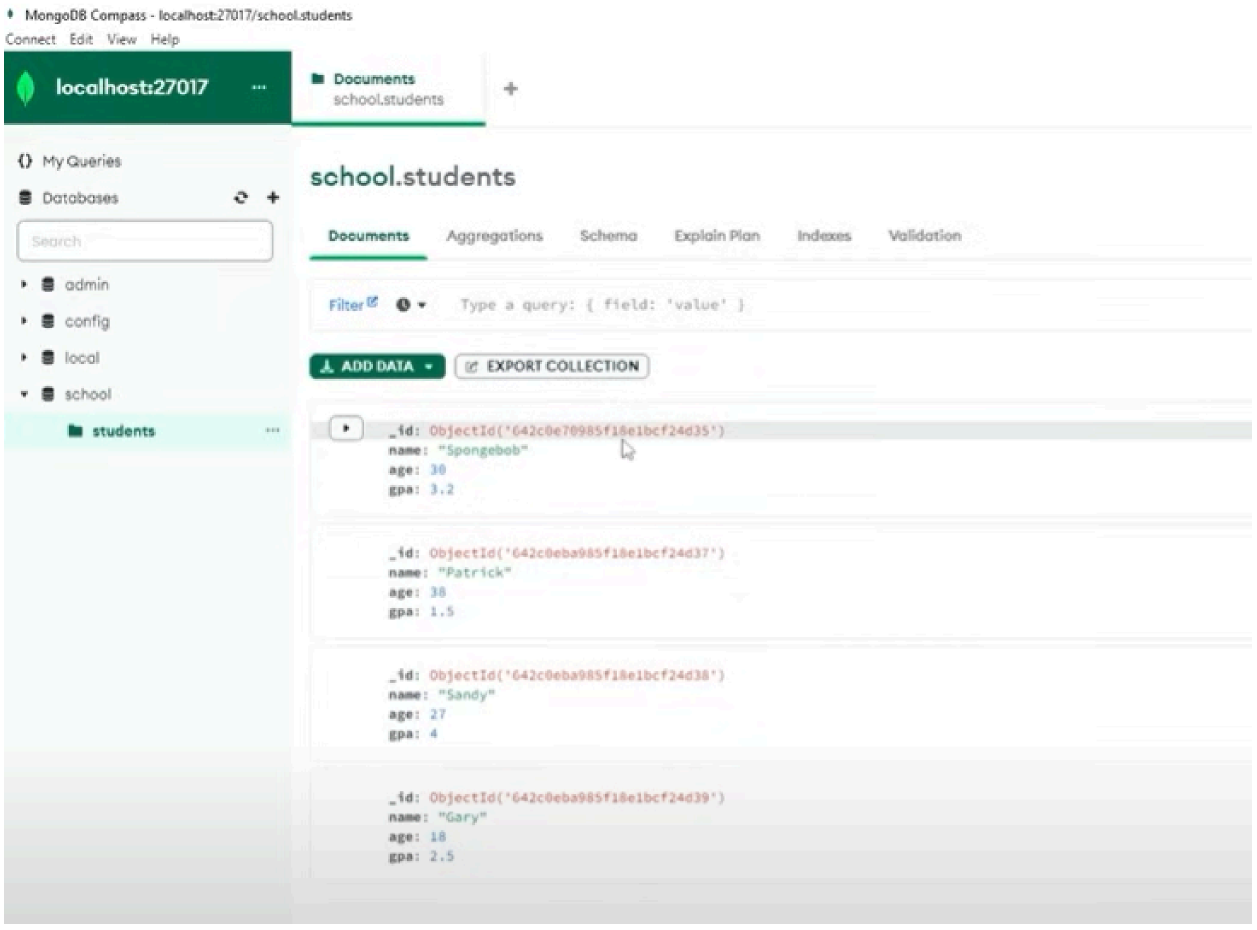
```
school> db.students.insertMany([{name:"Patrick", age:38, gpa:1.5}, {name:"Sandy"
, age:27, gpa:4.0}, {name:"Gary", age:18, gpa:2.5}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("642c0de70ce169a2bd211bc0"),
    '1': ObjectId("642c0de70ce169a2bd211bc1"),
    '2': ObjectId("642c0de70ce169a2bd211bc2")
  }
}
```

Then check with

db.students.find()

Using compass :

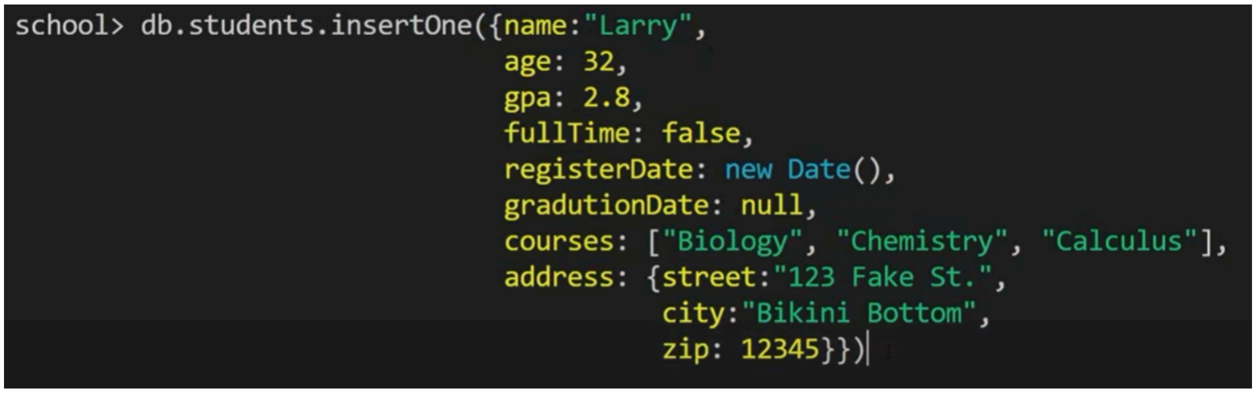




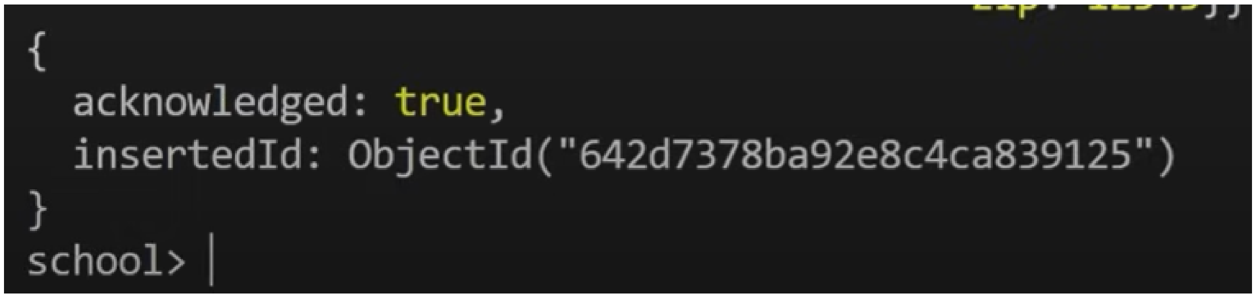
```
_id: ObjectId('642c0e70985f18e1bcf24d35')
name: "Spongebob"
age: 30
gpa: 3.2

_id: ObjectId('642c0eba985f18e1bcf24d37')
name: "Patrick"
age: 38
gpa: 1.5

_id: ObjectId('642c0eba985f18e1bcf24d38')
name: "Sandy"
age: 27
gpa: 4

_id: ObjectId('642c0eba985f18e1bcf24d39')
name: "Gary"
age: 18
gpa: 2.5
```

Data types :

https://www.w3schools.in/mongodb/data-types



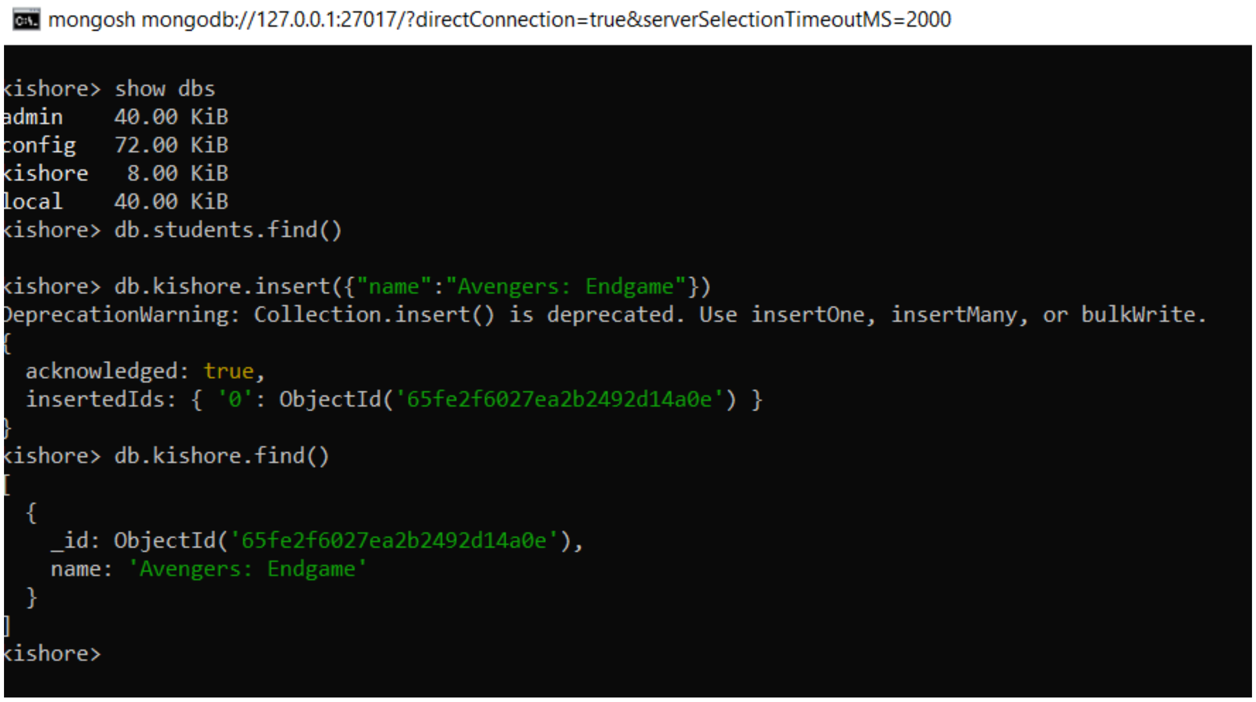
```
school> db.students.insertOne({name:"Larry",
                                age: 32,
                                gpa: 2.8,
                                fullTime: false,
                                registerDate: new Date(),
                                gradutionDate: null,
                                courses: ["Biology", "Chemistry", "Calculus"],
                                address: {street:"123 Fake St.",
                                          city:"Bikini Bottom",
                                          zip: 12345}})
```



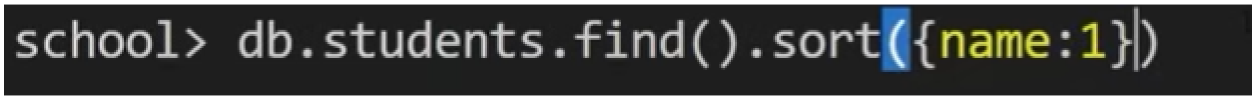
```
{
  acknowledged: true,
  insertedId: ObjectId("642d7378ba92e8c4ca839125")
}
school> 
```

Then clear the page using cls .

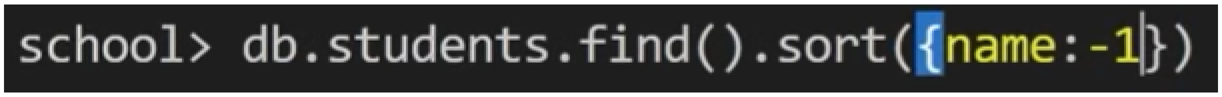
Sorting and limiting :



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000

kishore> show dbs
admin    40.00 KiB
config   72.00 KiB
kishore   8.00 KiB
local    40.00 KiB
kishore> db.students.find()

kishore> db.kishore.insert({"name":"Avengers: Endgame"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65fe2f6027ea2b2492d14a0e') }
}
kishore> db.kishore.find()
[
  {
    _id: ObjectId('65fe2f6027ea2b2492d14a0e'),
    name: 'Avengers: Endgame'
  }
]
kishore>
```



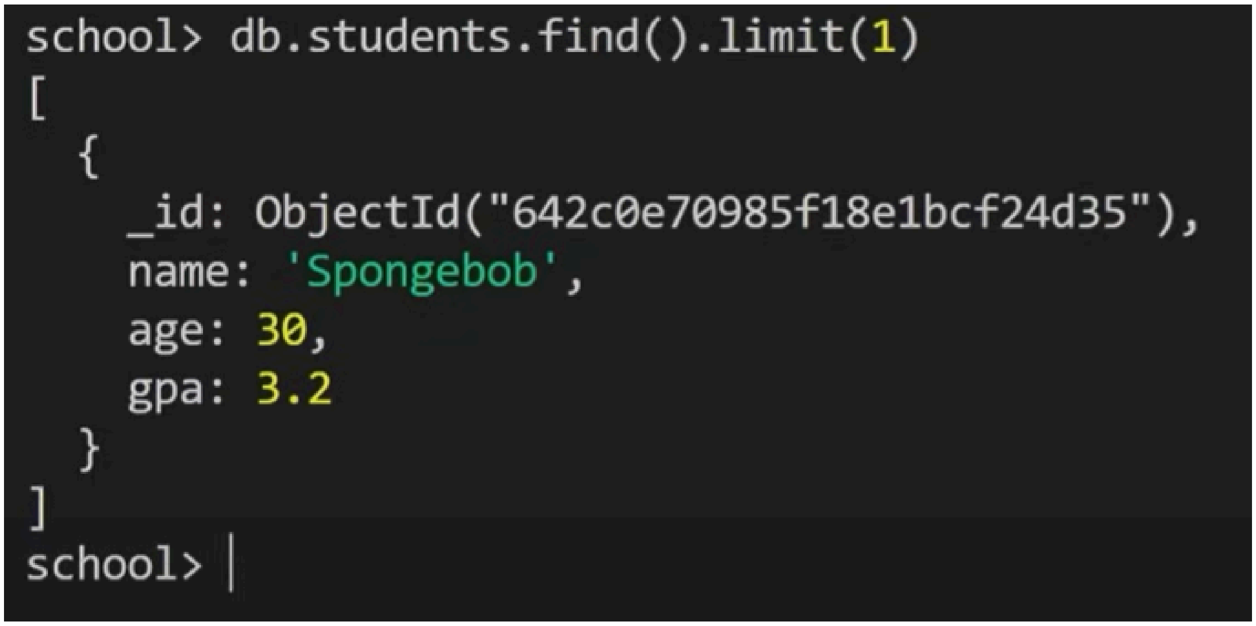
```
school> db.students.find().sort({name:1})
```



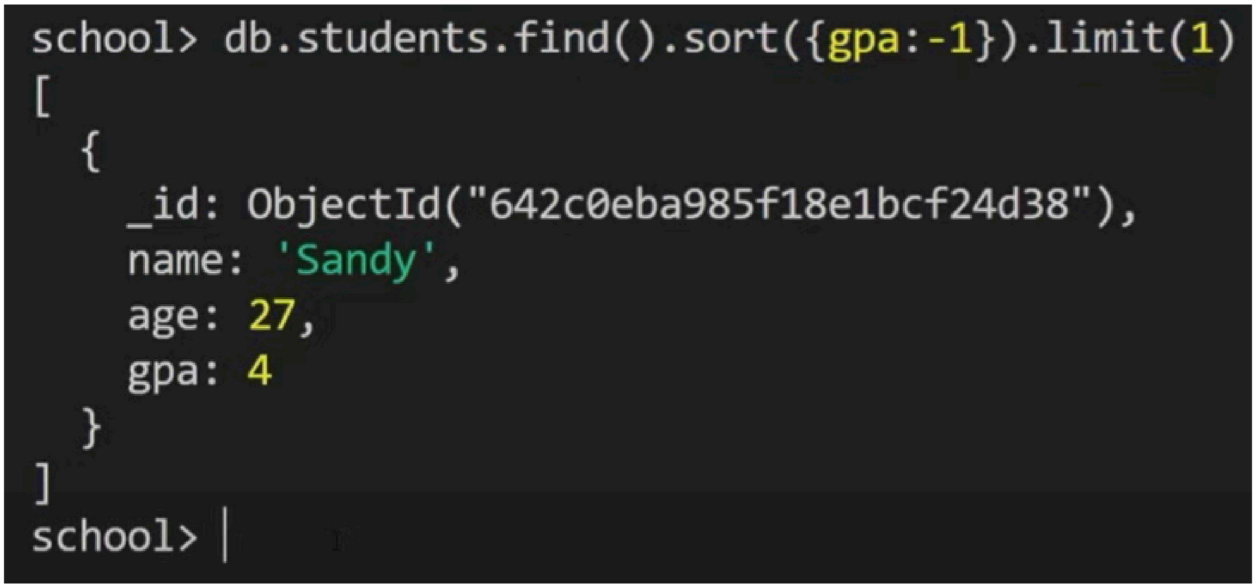
```
school> db.students.find().sort({name:-1})
```

Positive one - gives same as ascending order - low to high
Negative one - give same as descending order - high to low

Here i can limit my view of output :

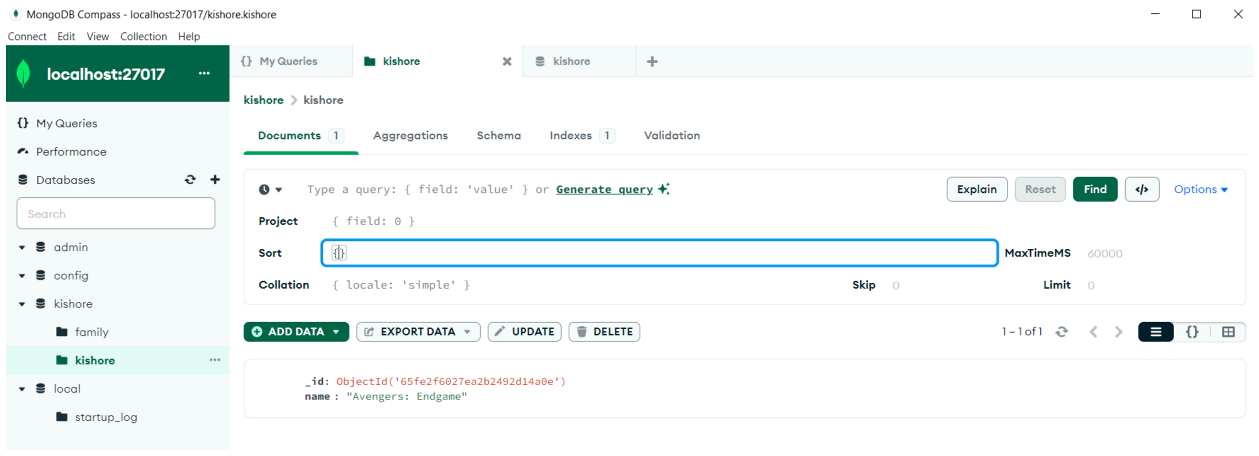


```
school> db.students.find().limit(1)
[
  {
    _id: ObjectId("642c0e70985f18e1bcf24d35"),
    name: 'Spongebob',
    age: 30,
    gpa: 3.2
  }
]
school> 
```



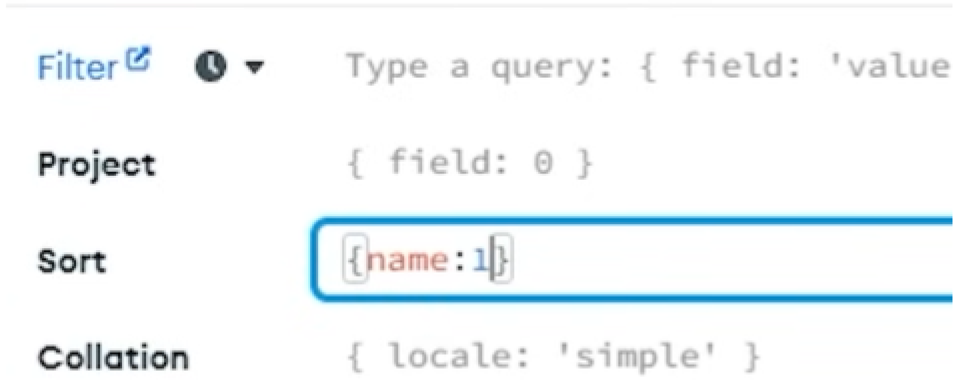
```
school> db.students.find().sort({gpa:-1}).limit(1)
[
  {
    _id: ObjectId("642c0eba985f18e1bcf24d38"),
    name: 'Sandy',
    age: 27,
    gpa: 4
  }
]
school> 
```

In above sample : we sorted who has highest gpa and display the view result as 1.



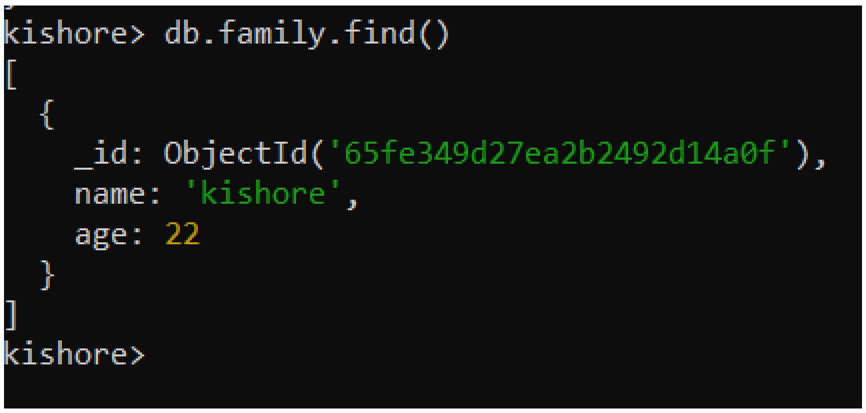
```
Type a query: { field: 'value' } or Generate query
Project    { field: 0 }
Sort       {}                         MaxTimeMS  60000
Collation  { locale: 'simple' }       Skip  0    Limit  0
```

```
_id: ObjectId('65fe2f6027ea2b2492d14a0e')
name : "Avengers: Endgame"
```

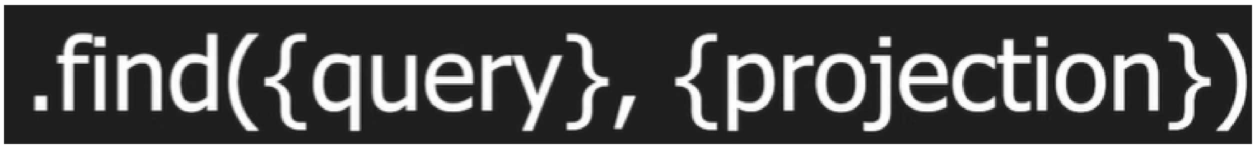


```
Filter     Type a query: { field: 'value
Project    { field: 0 }
Sort       {name:1}
Collation  { locale: 'simple' }
```

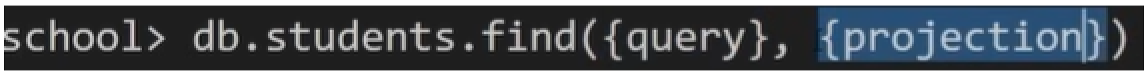
Find :



```
kishore> db.family.find()
[
  {
    _id: ObjectId('65fe349d27ea2b2492d14a0f'),
    name: 'kishore',
    age: 22
  }
]
kishore>
```

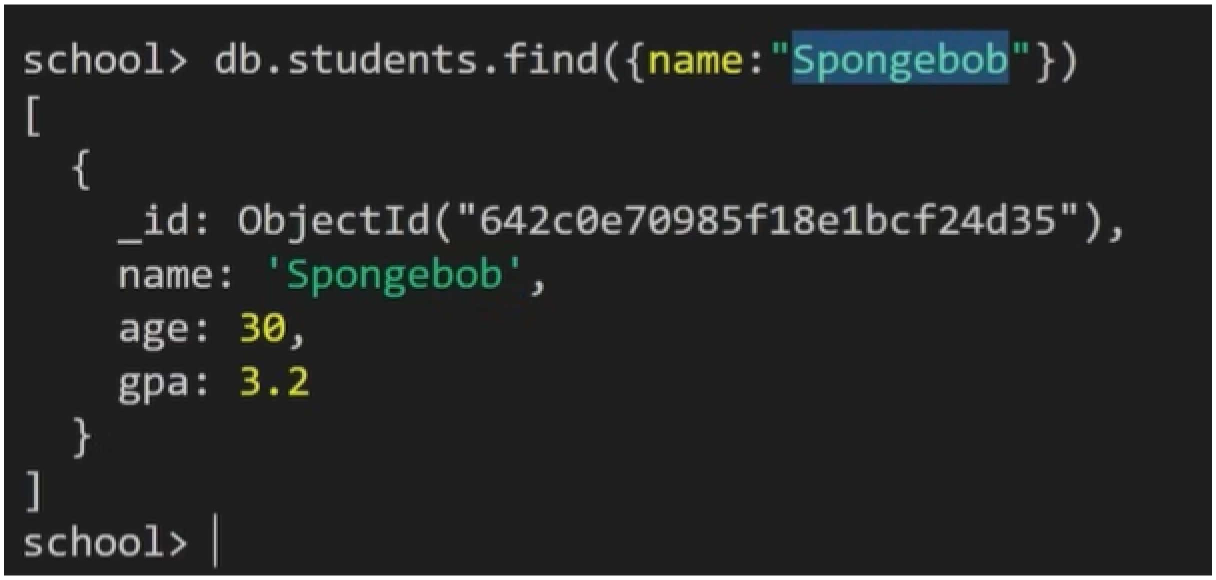


```
.find({query}, {projection})
```



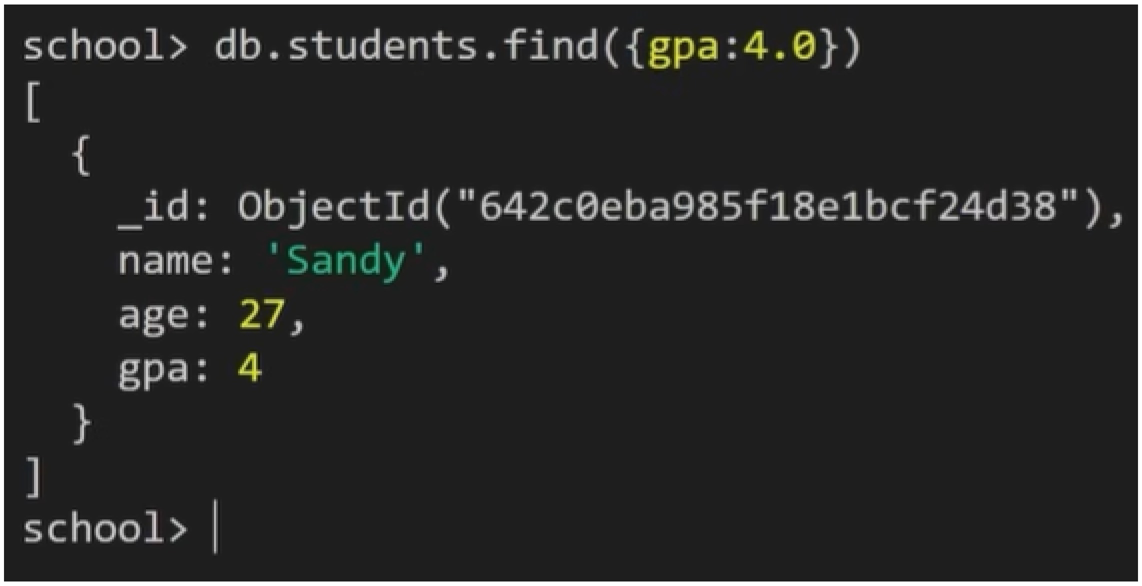
```
school> db.students.find({query}, {projection})
```

Its very similar to where class in sql ..

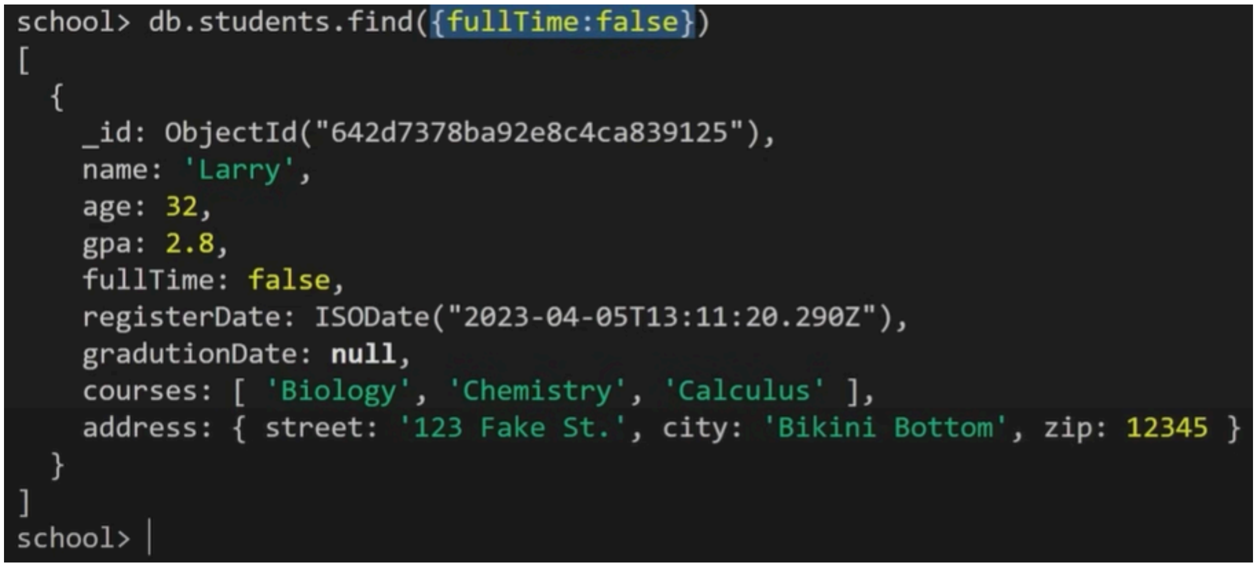


```
school> db.students.find({name:"Spongebob"})
[
  {
    _id: ObjectId("642c0e70985f18e1bcf24d35"),
    name: 'Spongebob',
    age: 30,
    gpa: 3.2
  }
]
school> |
```

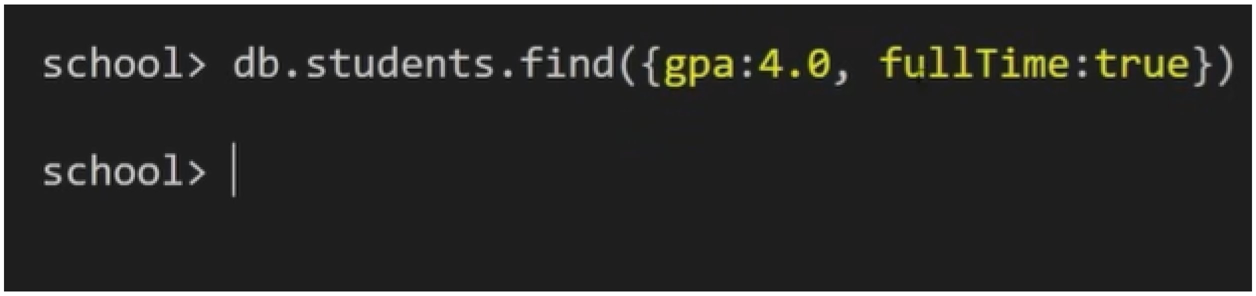
Then clear the shell using cls



```
school> db.students.find({gpa:4.0})
[
  {
    _id: ObjectId("642c0eba985f18e1bcf24d38"),
    name: 'Sandy',
    age: 27,
    gpa: 4
  }
]
school> |
```

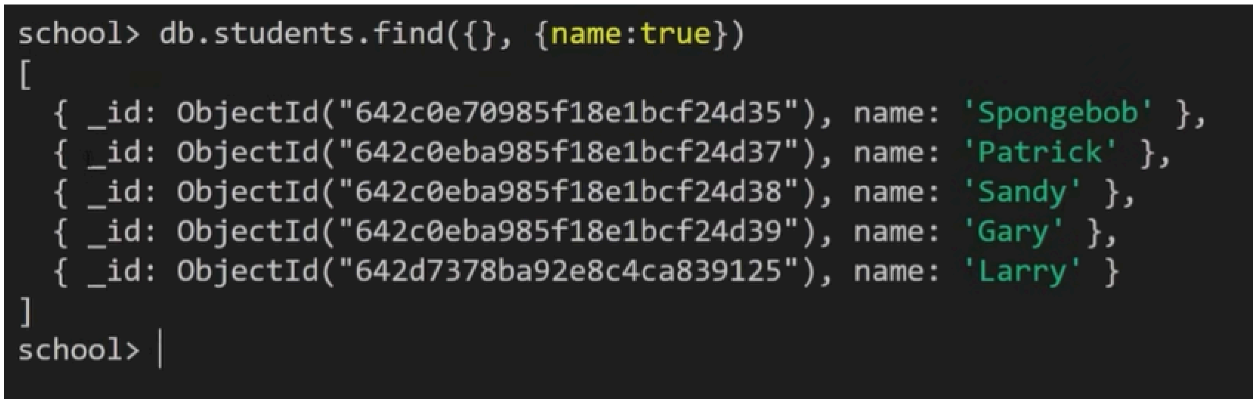


```
school> db.students.find({fullTime:false})
[
  {
    _id: ObjectId("642d7378ba92e8c4ca839125"),
    name: 'Larry',
    age: 32,
    gpa: 2.8,
    fullTime: false,
    registerDate: ISODate("2023-04-05T13:11:20.290Z"),
    gradutionDate: null,
    courses: [ 'Biology', 'Chemistry', 'Calculus' ],
    address: { street: '123 Fake St.', city: 'Bikini Bottom', zip: 12345 }
  }
]
school> 
```

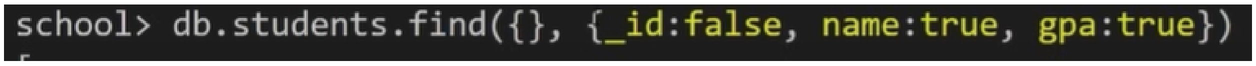


```
school> db.students.find({gpa:4.0, fullTime:true})

school> 
```

Its projection - second concept :

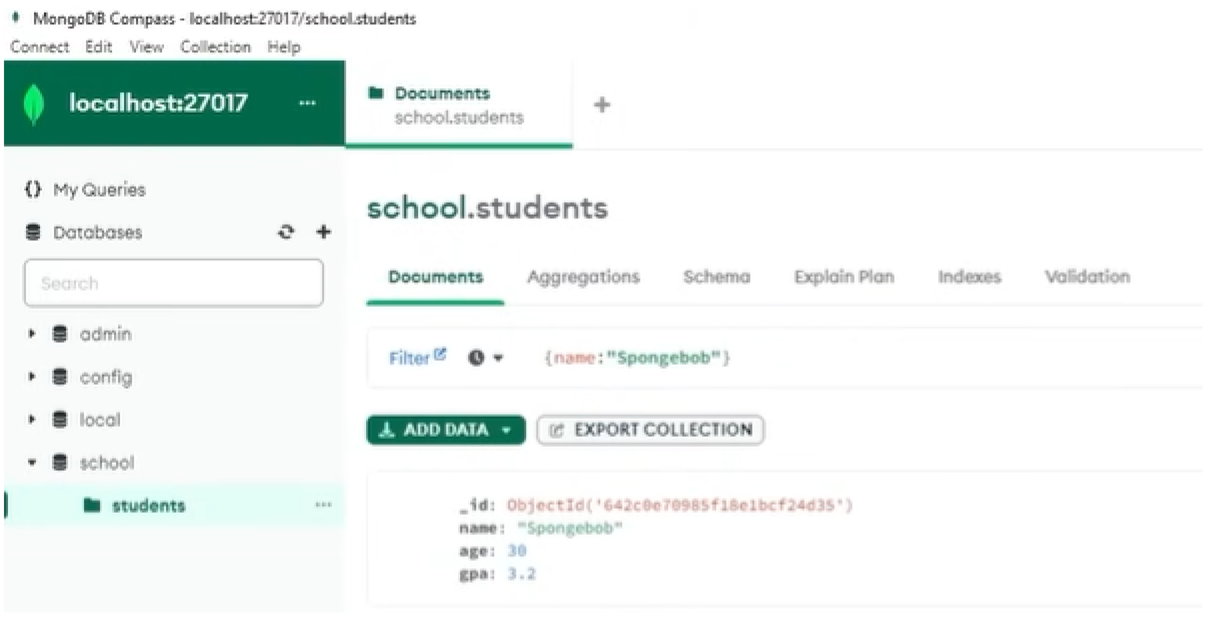


```
school> db.students.find({}, {name:true})
[
  { _id: ObjectId("642c0e70985f18e1bcf24d35"), name: 'Spongebob' },
  { _id: ObjectId("642c0eba985f18e1bcf24d37"), name: 'Patrick' },
  { _id: ObjectId("642c0eba985f18e1bcf24d38"), name: 'Sandy' },
  { _id: ObjectId("642c0eba985f18e1bcf24d39"), name: 'Gary' },
  { _id: ObjectId("642d7378ba92e8c4ca839125"), name: 'Larry' }
]
school> 
```



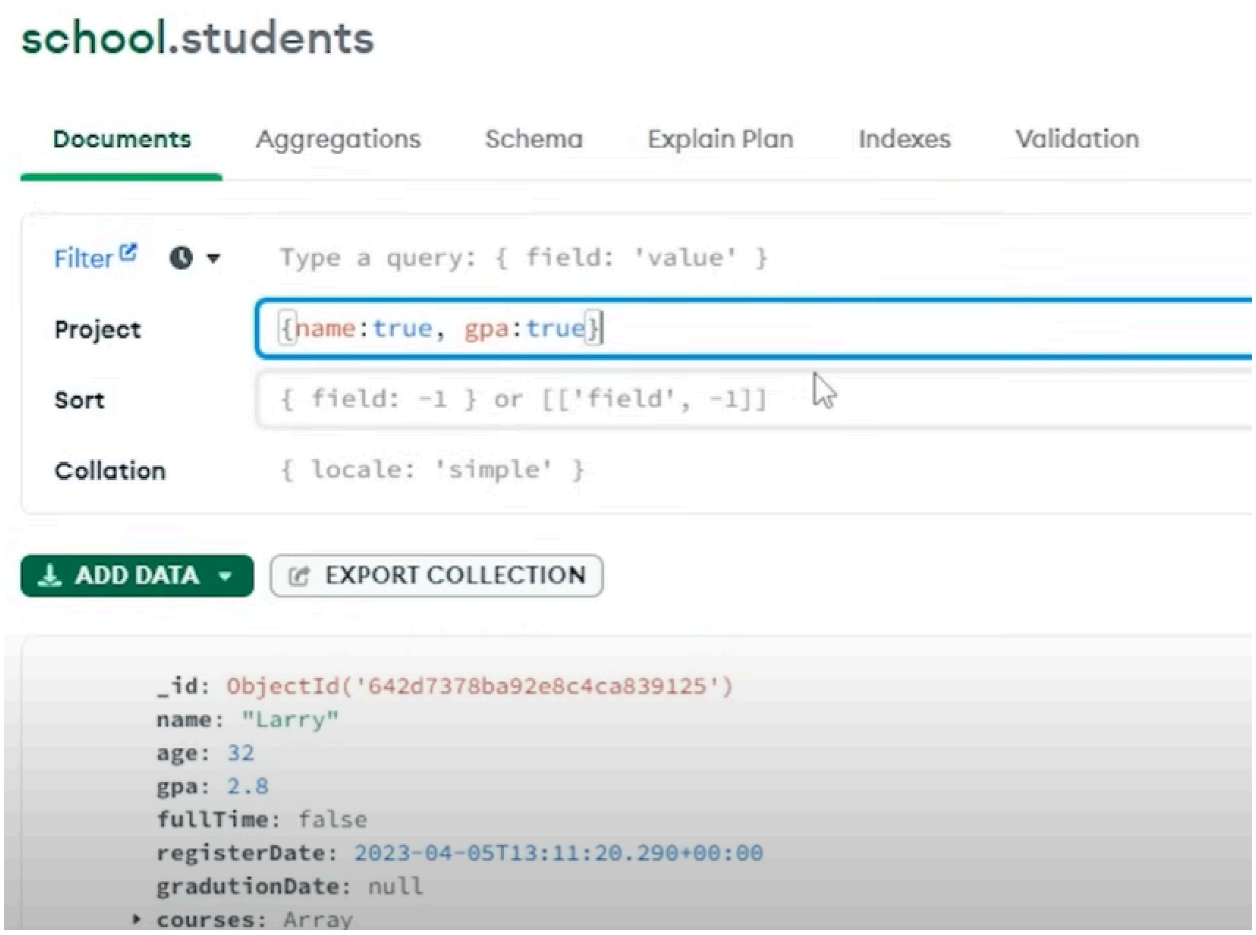
```
school> db.students.find({}, {_id:false, name:true, gpa:true})
```

Then clear the shell using cls command.


MongoDB Compass - localhost:27017/school.students
Connect Edit View Collection Help
localhost:27017
Documents
school.students
My Queries
Databases
Search
admin
config
local
school
students
school.students
Documents Aggregations Schema Explain Plan Indexes Validation
ADD DATA
EXPORT COLLECTION

Filter `{name:"Spongebob"}`

```
_id: ObjectId('642c0e70985f18e1bcf24d35')
name: "Spongebob"
age: 30
gpa: 3.2
```


school.students
Documents Aggregations Schema Explain Plan Indexes Validation
ADD DATA
EXPORT COLLECTION

Filter `Type a query: { field: 'value' }`

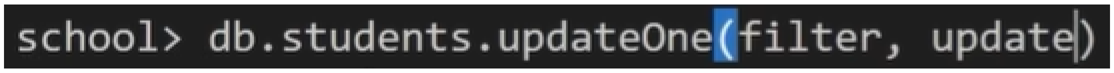
Project `{name:true, gpa:true}`

Sort `{ field: -1 } or [['field', -1]]`

Collation `{ locale: 'simple' }`

```
_id: ObjectId('642d7378ba92e8c4ca839125')
name: "Larry"
age: 32
gpa: 2.8
fullTime: false
registerDate: 2023-04-05T13:11:20.290+00:00
gradutionDate: null
courses: Array
```

Update :



```
school> db.students.updateOne(filter, update)
```

Here filter is the selection criteria for the update.

```
.updateOne(filter, update)
```

All Examples :
# MongoDB Cheat Sheet

## Show All Databases

```
show dbs
```

## Show Current Database

```
db
```

## Create Or Switch Database

```
use acme
```

## Drop

```
db.dropDatabase()
```

## Create Collection

```
db.createCollection('posts')
```

## Show Collections

```
show collections
```

## Insert Row

```
db.posts.insert({
  title: 'Post One',
  body: 'Body of post one',
  category: 'News',
  tags: ['news', 'events'],
  user: {
    name: 'John Doe',
    status: 'author'
  },
  date: Date()
})
```

## Insert Multiple Rows

```
db.posts.insertMany([
  {
    title: 'Post Two',
    body: 'Body of post two',
    category: 'Technology',
    date: Date()
  },
  {
    title: 'Post Three',
    body: 'Body of post three',
    category: 'News',
    date: Date()
  },
  {
    title: 'Post Four',
    body: 'Body of post three',
    category: 'Entertainment',
    date: Date()
  }
])
```

## Get All Rows

```
db.posts.find()
```

## Get All Rows Formatted

```
db.posts.find().pretty()
```

## Find Rows

```
db.posts.find({ category: 'News' })
```

## Sort Rows

```
# asc
db.posts.find().sort({ title: 1 }).pretty()
# desc
db.posts.find().sort({ title: -1 }).pretty()
```

## Count Rows

```
db.posts.find().count()
db.posts.find({ category: 'news' }).count()
```

## Limit Rows

```
db.posts.find().limit(2).pretty()
```

## Chaining

```
```

db.posts.find().limit(2).sort({ title: 1 }).pretty()
```

## Foreach

```
db.posts.find().forEach(function(doc) {
  print("Blog Post: " + doc.title)
})
```

## Find One Row

```
db.posts.findOne({ category: 'News' })
```

## Find Specific Fields

```
db.posts.find({ title: 'Post One' }, {
  title: 1,
  author: 1
})
```

## Update Row

```
db.posts.update({ title: 'Post Two' },
{
  title: 'Post Two',
  body: 'New body for post 2',
  date: Date()
},
{
  upsert: true
})
```

## Update Specific Field

```
db.posts.update({ title: 'Post Two' },

```
{
  $set: {
    body: 'Body for post 2',
    category: 'Technology'
  }
})
```

## Increment Field (\$inc)

```
db.posts.update({ title: 'Post Two' },
{
  $inc: {
    likes: 5
  }
})
```

## Rename Field

```
db.posts.update({ title: 'Post Two' },
{
  $rename: {
    likes: 'views'
  }
})
```

## Delete Row

```
db.posts.remove({ title: 'Post Four' })
```

## Sub-Documents

```
db.posts.update({ title: 'Post One' },
{
  $set: {
    comments: [
      {
        body: 'Comment One',
        user: 'Mary Williams',
        date: Date()
      },
      {
        body: 'Comment Two',
        user: 'Harry White',
        date: Date()
      }
    ]
  }
})
```

## Find By Element in Array (\$elemMatch)

```
db.posts.find({
  comments: {
     $elemMatch: {
       user: 'Mary Williams'
       }
    }
  }
)
```

## Add Index

```
db.posts.createIndex({ title: 'text' })
```

## Text Search

```
db.posts.find({
  $text: {
    $search: "\"Post O\""
    }
})
```

## Greater & Less Than

```
db.posts.find({ views: { $gt: 2 } })
db.posts.find({ views: { $gte: 7 } })
db.posts.find({ views: { $lt: 7 } })
db.posts.find({ views: { $lte: 7 } })
```