**Practical-1**

**Aim: Case Study on Big Data and its applications**

### **Big Data Case Study – Uber**

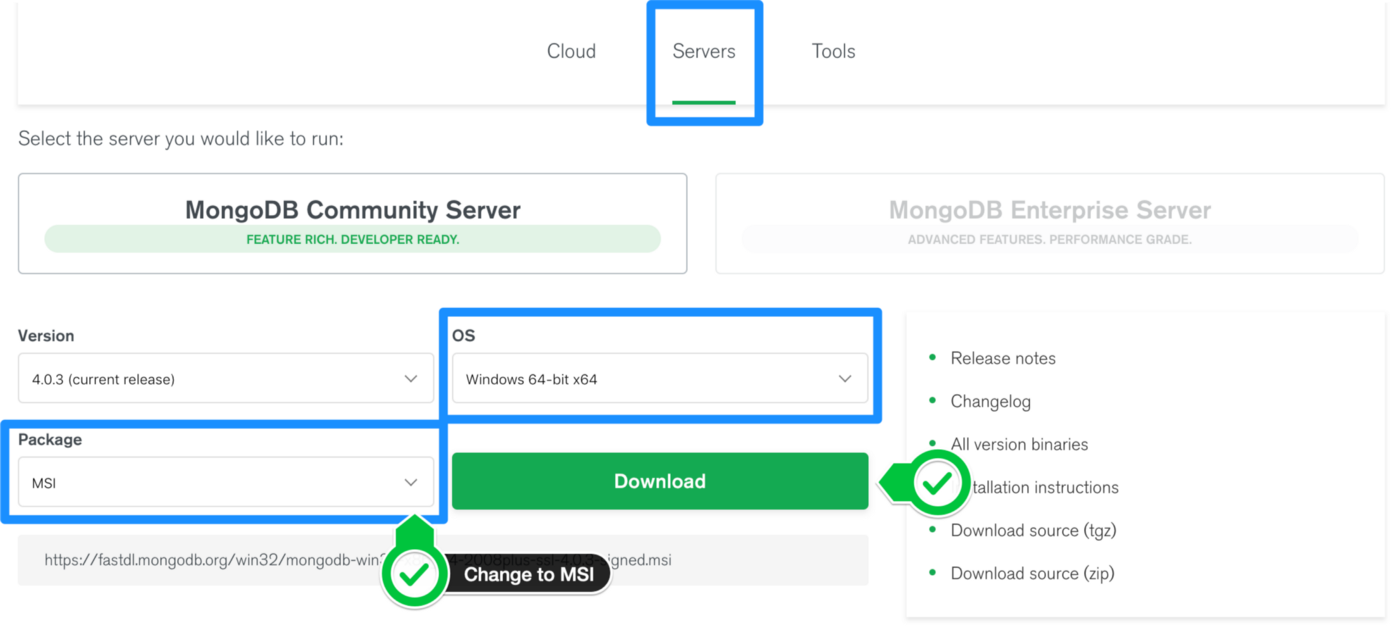
* Uber focuses on the supply and demand of the services due to which the prices of the services provided changes. Therefore one of Uber’s biggest uses of data is surge pricing. For instance, if you are running late for an appointment and you book a cab in a crowded place then you must be ready to pay twice the amount.
* Uber is the first choice for people around the world when they think of moving people and making deliveries. It uses the personal data of the user to closely monitor which features of the service are mostly used, to analyse usage patterns and to determine where the services should be more focused.
* For example, On New Year’s Eve, the price for driving for one mile can go from 200 to 1000. In the short term, surge pricing affects the rate of demand, while long term use could be the key to retaining or losing customers. *Machine learning algorithms* are considered to determine where the demand is strong.

**Practical -2**

**Aim: Installation of MongoDB**

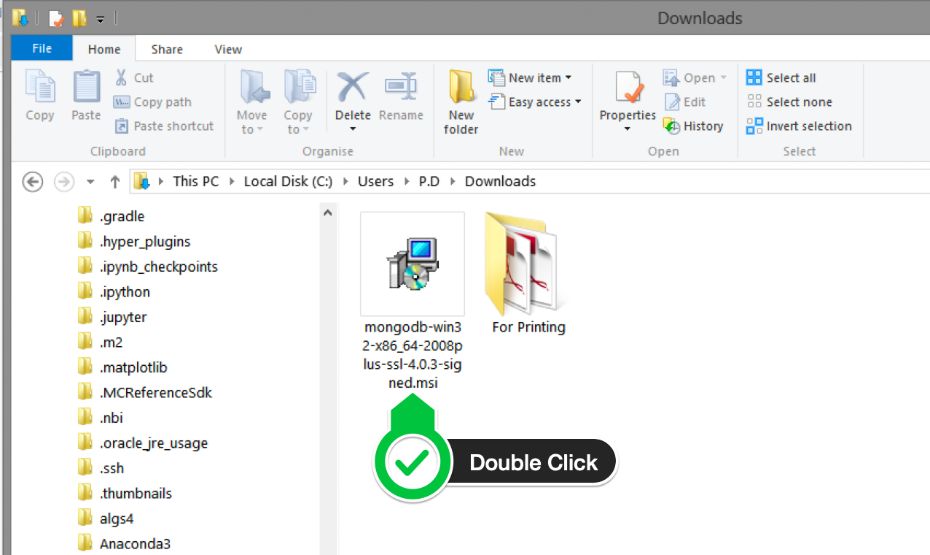
**Step 1 :- Download the MongoDB MSI Installer Package**

Head over [here](https://www.mongodb.com/download-center/community) and download the current version of MongoDB. Make sure you select MSI as the package you want to download.

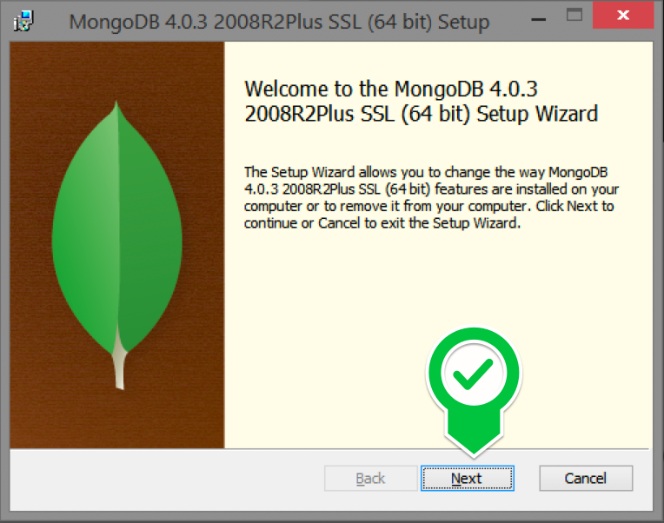
****

**Step 2 :- Install MongoDB with the Installation Wizard**

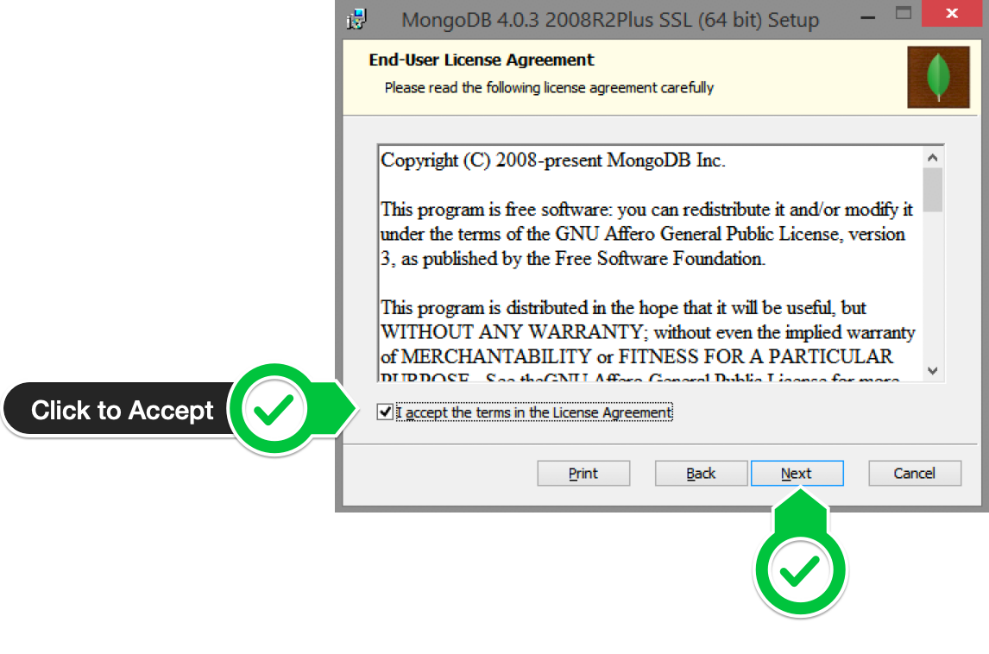
1. Make sure you are logged in as a user with Admin privileges. Then navigate to your downloads folder and double click on the .msi package you just downloaded. This will launch the installation wizard.

****

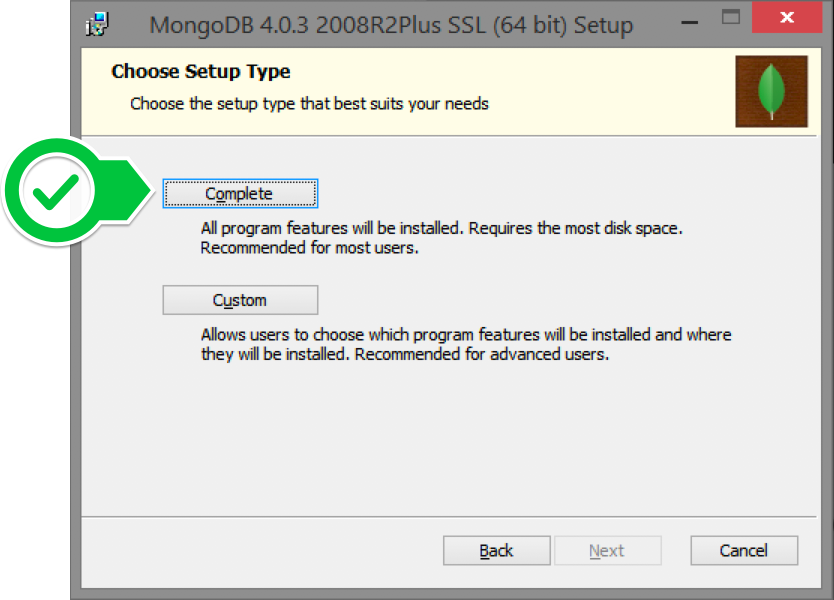
1. Click Next to start installation.

****

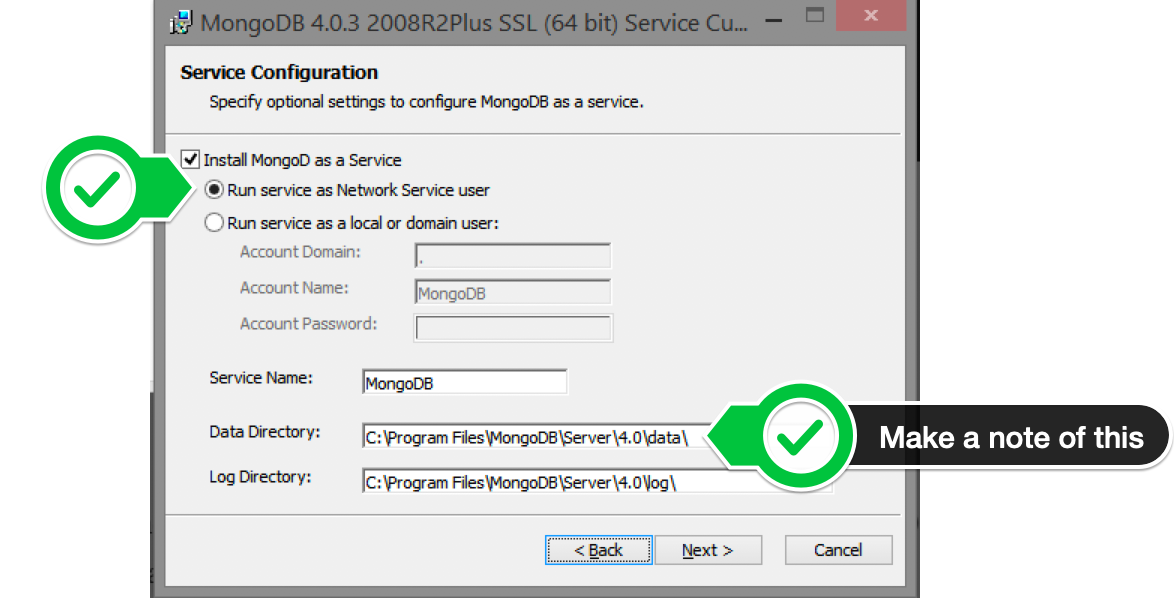
1. Accept the licence agreement then click Next.

****

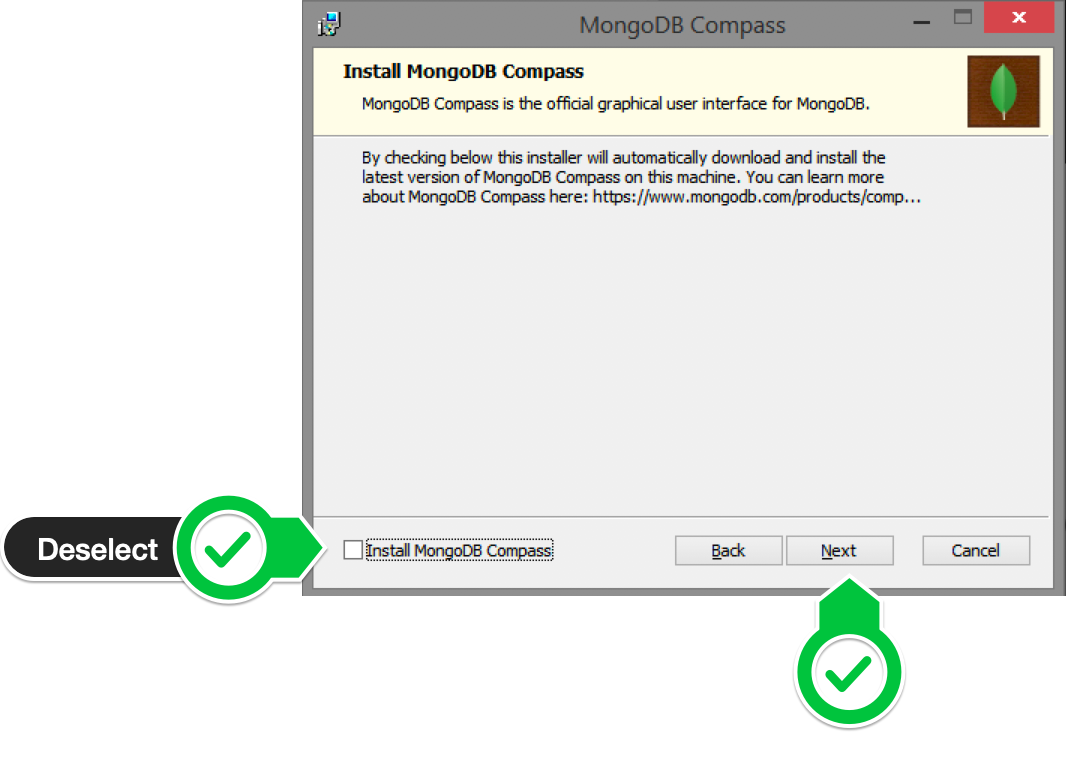
1. Select the Complete setup.

****

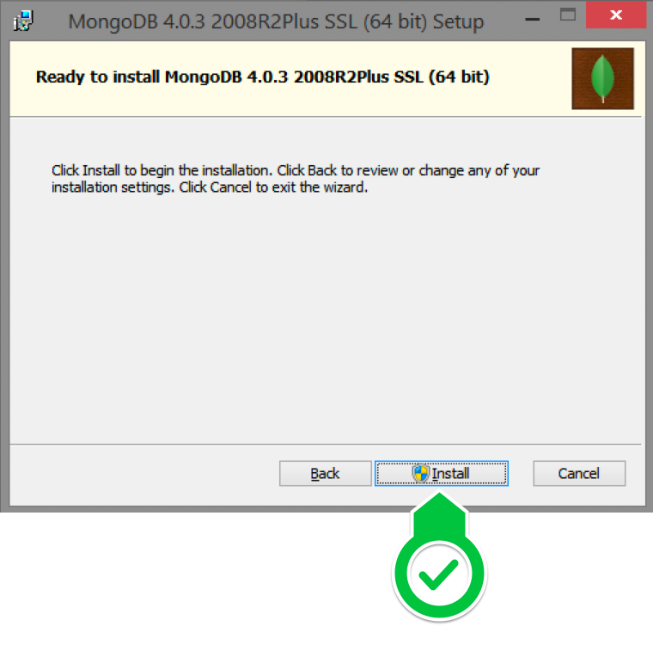
1. Select “Run service as Network Service user” and make a note of the data directory, we’ll need this later.

****

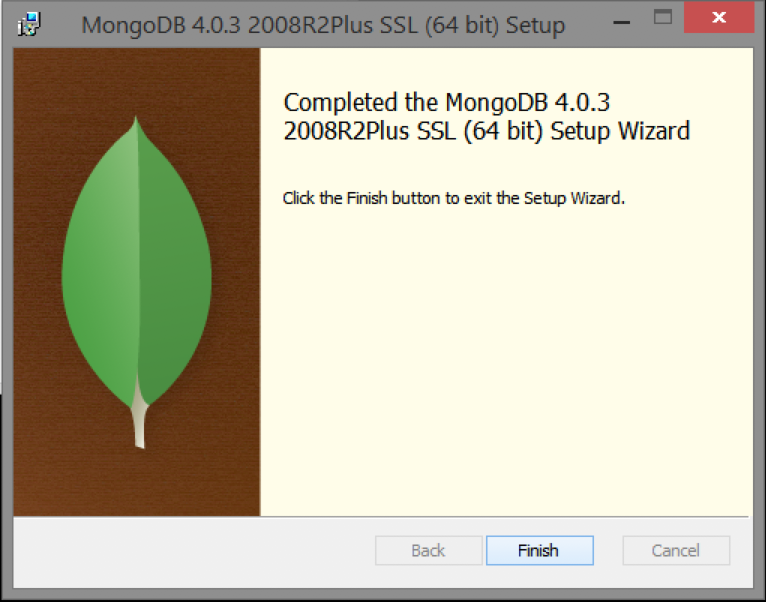
1. We won’t need Mongo Compass, so deselect it and click Next.

****

1. Click Install to begin installation.

****

1. Hit Finish to complete installation.

****

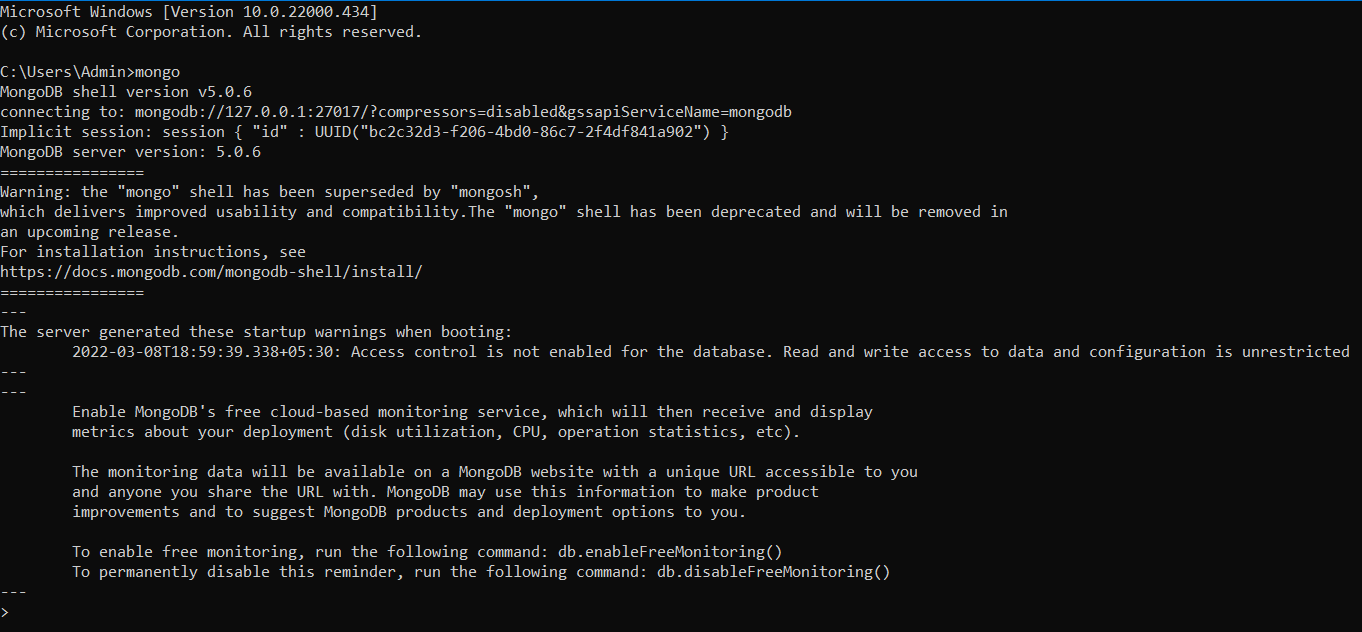
**Practical -3**

**Aim: CRUD operations on MONGODB**

1. **How to start the Mongodb server.**

**Command :- mongo**

**Output :-**



1. **Create Database**

**Syntax :- use <DATABASE NAME>**

**Example :- To create the database named mydb**

**Command :- use student\_details**

**Output :-**



1. **Create collection**

**Syntax :-** db.createCollection(<name of collection>)

**Example** :- To create the collection name student

**Command** :- db.createCollection("student\_details")

**Output :-**



1. **Check the collection is created or not**

**Command :- show collections**

**Output :-**



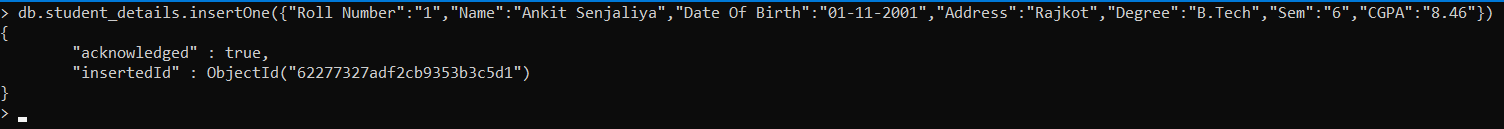
1. **Insert the data in collection(InsertOne and InsertMany)**

**Syntax :-** db.<name of collection >.insert(data)

**Example** :- To insert student’s detail like roll no, name, date of birth, address, degree, sem, CGPA

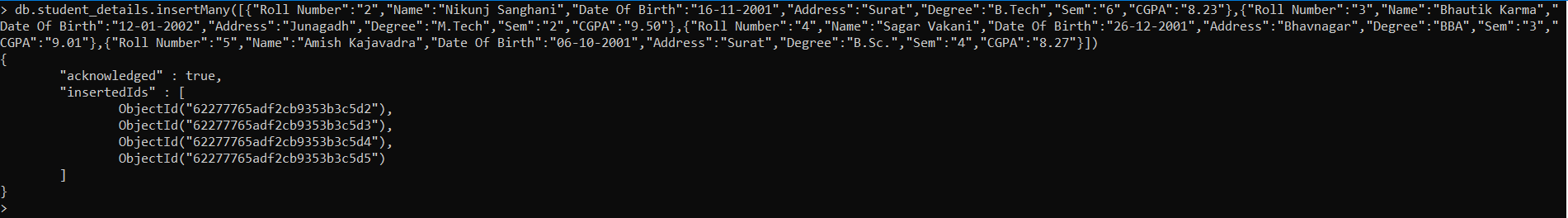
**Command** :- db.student\_details.insertOne()

**Output :-**



**Command** :- db.student\_details.insertMany([{}])

**Output :-**

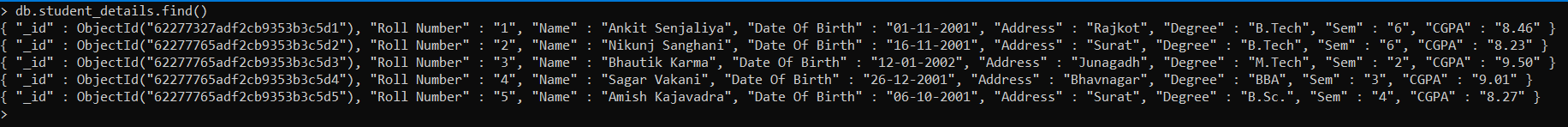


1. **To check the data store or not**

**Syntax :-** db.<name of collection>.find()

**Command :-** db.student\_details.find()

**Output :-**

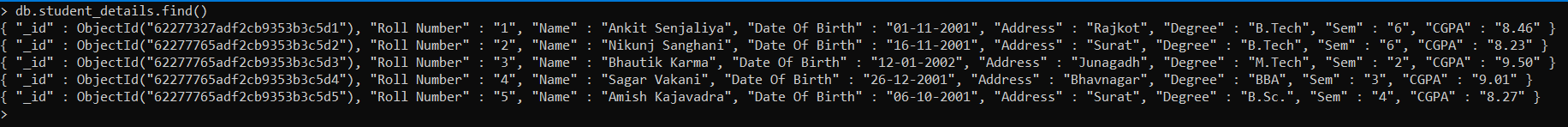


1. **Show the collection’s data in formatted**

**Syntax :-** db.<name of collection>.find()

**Command :-** db.student\_details.find()

**Output :-**

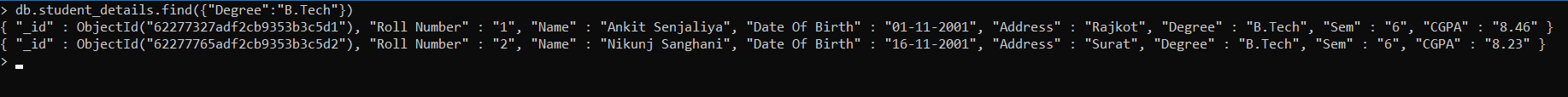


1. **Display all the students in degree BTECH**

Syntax :-

Command :- db.student\_details.find({"Degree":"B.Tech"})

Output :-

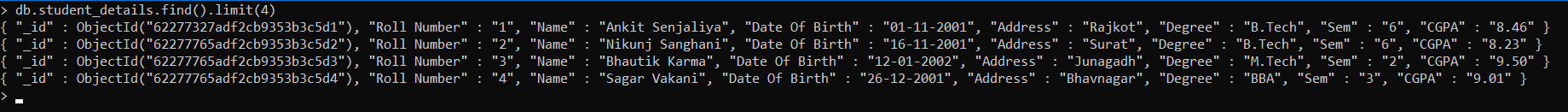


1. **Display 5 students**

Syntax :-

Command :- db.student\_details.find().limit(4)

Output :-

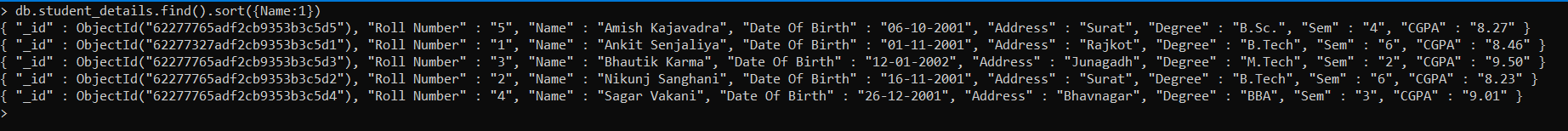


1. **Display all students in Ascending order**

Syntax :-

Command :- db.student\_details.find().sort({Name:1})

Output :-

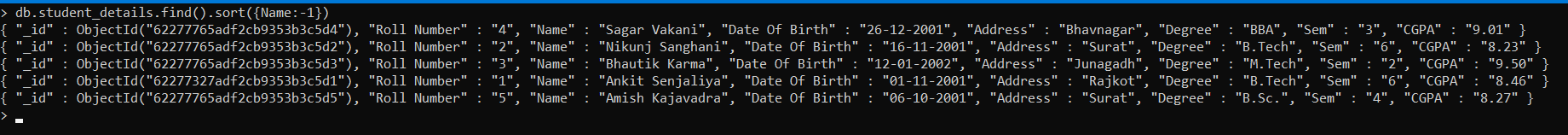


**11. Display all students in Descending order**

Syntax :-

Command :- db.student\_details.find().sort({Name:-1})

Output :-

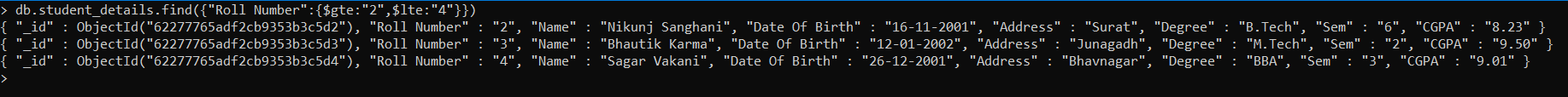


**12. Display students 5,6,7**

Syntax :-

Command :- db.student\_details.find({"Roll Number":{$gte:"2",$lte:"4"}})

Output :-

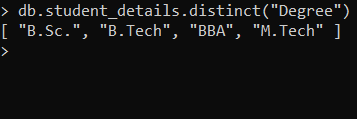


**13. Display all the distinct** degrees

Syntax :-

Command :- db.student\_details.distinct("Degree")

Output :-

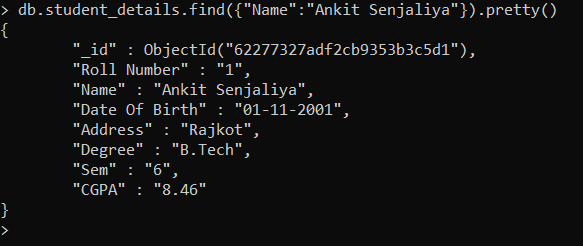


**14. Find the document anuj from the student collection**

**Syntax :-** db.<name of collection>.find({<name of document with key and value>})

**Command** :- db.student\_details.find({"Name":"Ankit Senjaliya"}).pretty()

**Output :-**

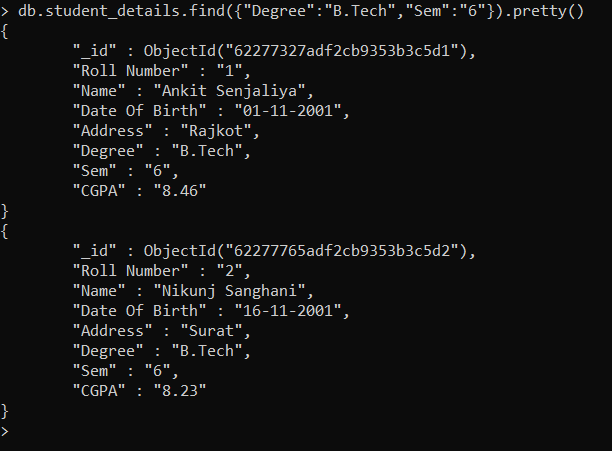


**15. Display all the students in Degree B.Tech and in semester 6th**

**Syntax :-**

**Command :-** db.student\_details.find({"Degree":"B.Tech","Sem":"6"}).pretty()

**Output :-**

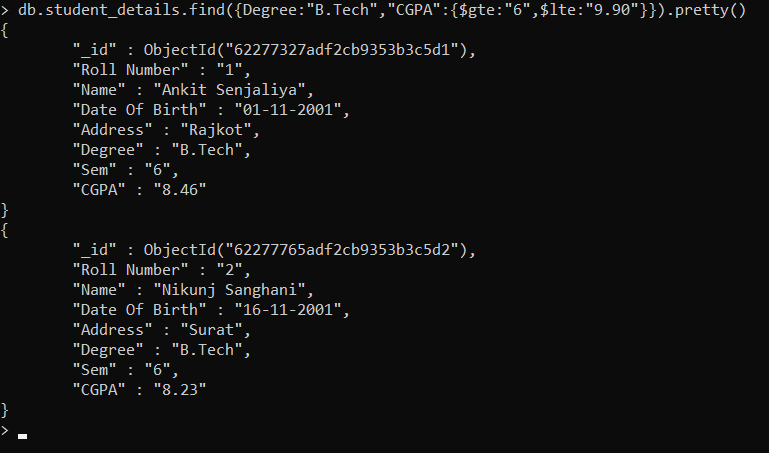


**16. Display all the B.TECH students in with CGPA greater than 6 but less than 7.5**

**Syntax :-**

**Command :-** db.student\_details.find({Degree:"B.Tech","CGPA":{$gte:"6",$lte:"9.90"}}).pretty()

**Output :-**

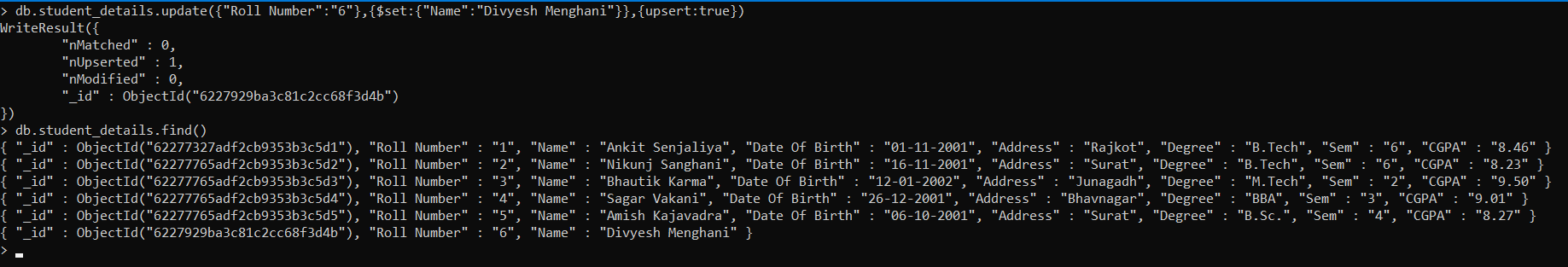


**17. Insert the document for RAJAT into the student collection only if it does not already exist in the collection. If it is present in the collection then update the document with new value.**

**Syntax :-** db.<collection name>.update({ data},{$set:{update data}},{upsert:true})

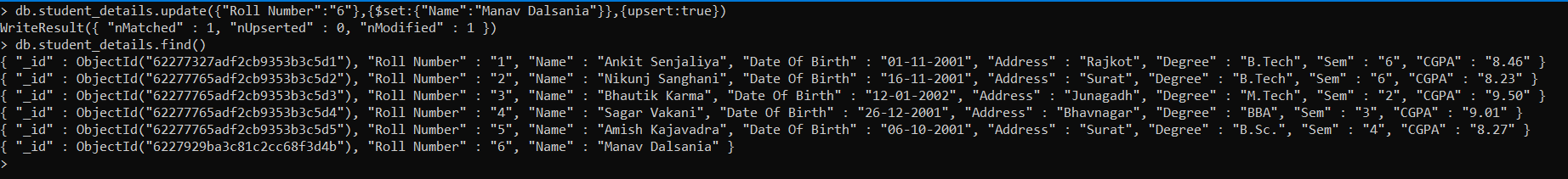
**Command** :- db.student\_details.update({"Roll Number":"6"},{$set:{"Name":"Divyesh Menghani"}},{upsert:true})

**Output :-**



**Command** :- db.student\_details.update({"Roll Number":"6"},{$set:{"Name":"Manav Dalsania"}},{upsert:true})

**Output :-**

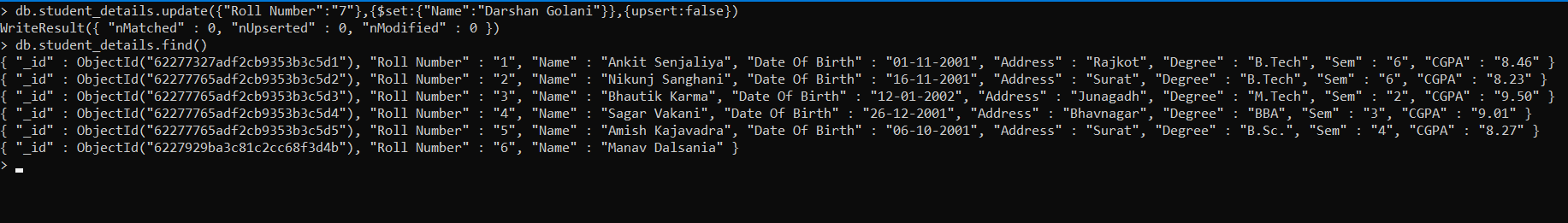


**18. update the document for roll\_no is 112 and name: vikram into the student collection it does not already exist in the collection then donot insert if it is exit then update with new value.**

**Syntax :-** db.<collection name>.update({ data},{$set:{update data}},{upsert:false})

**Command** :- db.student\_details.update({"Roll Number":"7"},{$set:{"Name":"Darshan Golani"}},{upsert:false})

**Output :-**



**19. Remove the branch from the rollo number 101**

**Syntax:** db.<name of collection>.update({key:{criteria value}} //update criteria

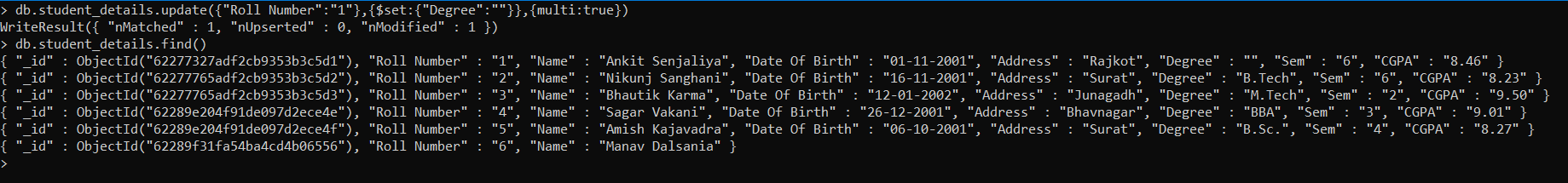
{$set: {key:value} //update action

{multi:true} // update option

})

**Command :-** db.student\_details.update({"Roll Number":"1"},{$set:{"Degree":""}},{multi:true})

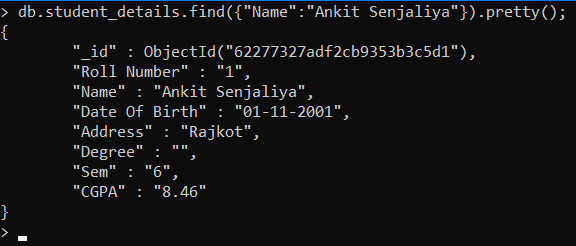
**Output :-**



**20. Find the name of students of the student collection**

**Command :-** db.student.find({},{name:1}).pretty();

**Output :-**



**21. Find the name of students of the student collection without id**

**Command :-** db.student.find({},{name:1,\_id:0}).pretty();

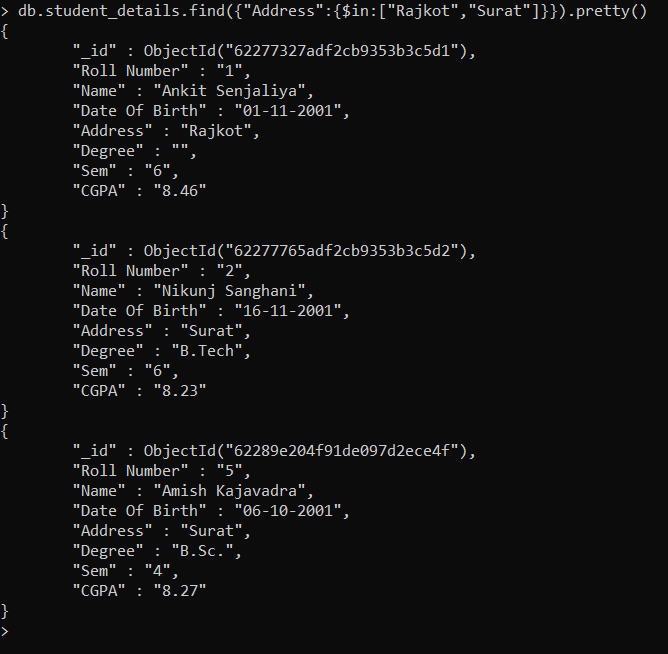
**Output :-**



**22. Find those documents where the state is gujrat and rajasthan from student collection**

**Command :-** db.student\_details.find({"Address":{$in:["Rajkot","Surat"]}}).pretty()

**Output :-**



**23. Find those documents where the state is not gujrat and rajasthan from student collection**

**Command :-** db.student\_details.find({"Address":{$nin:["Rajkot","Surat"]}}).pretty()

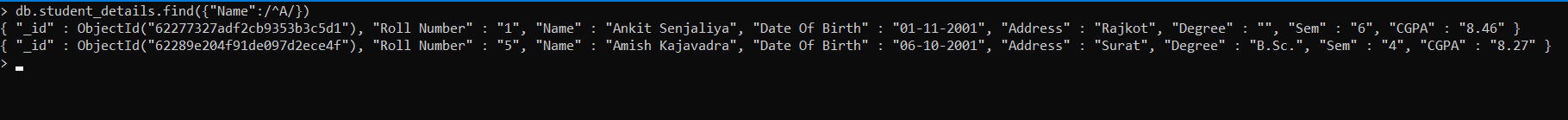
**Output :-**



**24. To find the document from the student collection where name begin with “s”**

**Command :-** db.student\_details.find({"Name":/^A/})

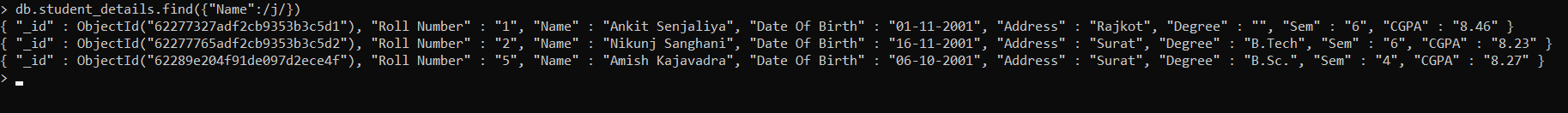
**Output :-**



**25. To find the document from the student collection where name has an “a” in any position**.

**Command :-** db.student\_details.find({"Name":/j/})

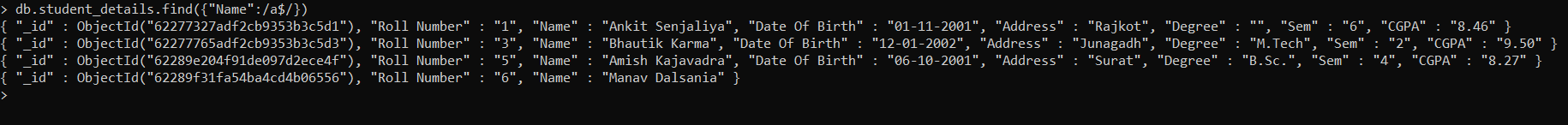
**Output :-**



**26. To find the document from the student collection where name end with “n”**

**Command :-** db.student\_details.find({"Name":/a$/})

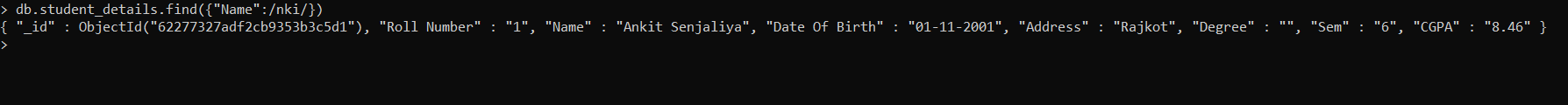
**Output :-**



**27. Displaying details of student who are having the word kush in their name field:**

**Command :-** db.student\_details.find({"Name":/nki/})

**Output :-**



**28. Displaying details of student who are a software engineer with case insensitive by using i <options>:**

**Command**

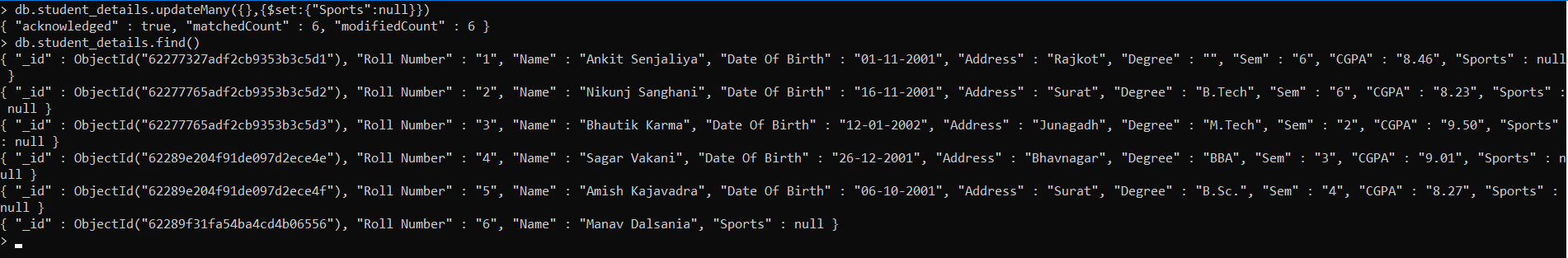
**Output**

**27. To add a new field with null value in existing document of student collection.**

**Syntax :-**

**Command :-** db.student\_details.updateMany({},{$set:{"Sports":null}})

**Output :-**



**28. To search for NULL value in branch column.**

**Command:**

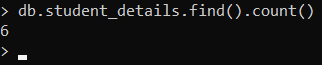
**Output:**

**29. To find the number of students in the Student collection list.**

**Syntax:** db.<Collection Name>.find().count()

**Command :-** db.student\_details.find().count()

**Output :-**



**30. To find the number of documents in the students collection wherein the state is gujrat.**

**Syntax :-** db.<Collection Name>.find({<Key>:’<Value>’}).count()

**Command :-** db.student\_details.find({"Address":"Rajkot"}).count()

**Output :-**



**Practical-4**

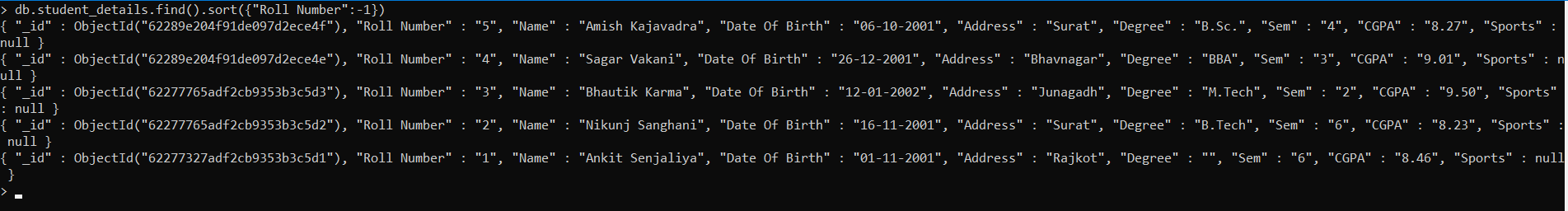
**Aim: Advance CRUD operations in MongoDB**

1. **To sort the documents from the Students collection first on roll\_no. In descending order and then on cities in the descending order.**

**Syntax :-** db.<Collection Name>.find().sort({<Key>:<-1 Or 1>})

**Command :-** db.student\_details.find().sort({"Roll Number":-1})

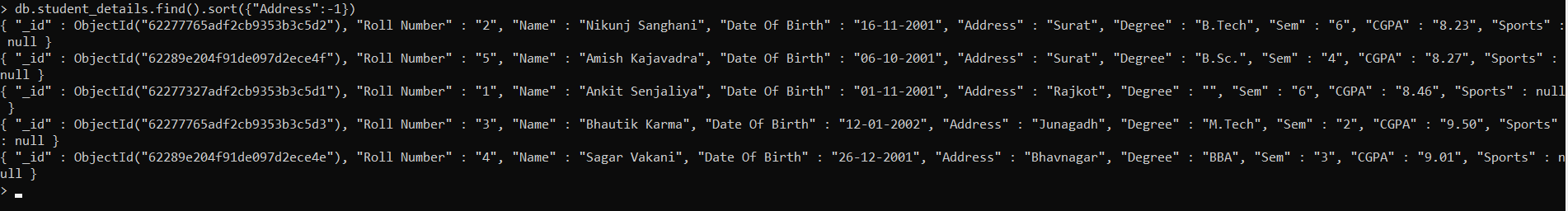
**Output :-**



**Syntax :-** db.<Collection Name>.find().sort({<Key>:<-1 Or 1>})

**Command :-** db.student\_details.find().sort({"Address":-1})

**Output :-**

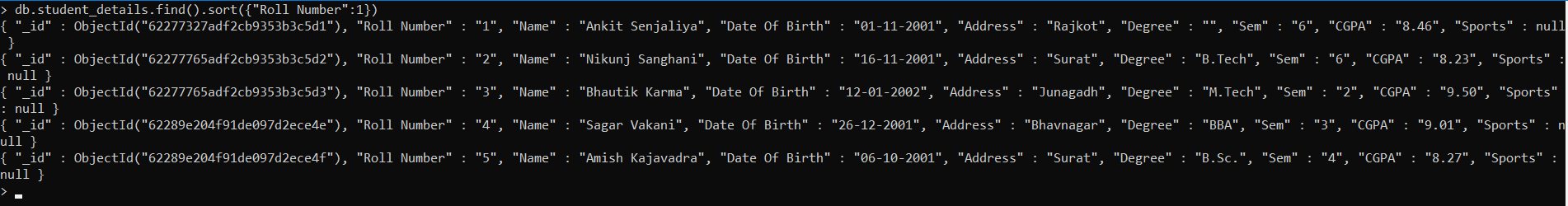


1. **To sort the documents from the Students collection first on rollno. In ascending order and then on cities in the ascending order.**

**Syntax :-** db.<Collection Name>.find().sort({<Key>:<-1 Or 1>})

**Command :-** db.student\_details.find().sort({"Roll Number":1})

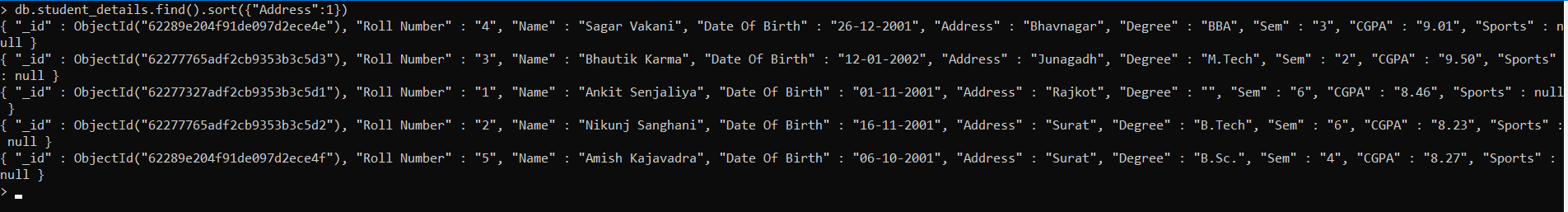
**Output :-**



**Syntax :-** db.<Collection Name>.find().sort({<Key>:<-1 Or 1>})

**Command :-** db.student\_details.find().sort({"Address":1})

**Output :-**

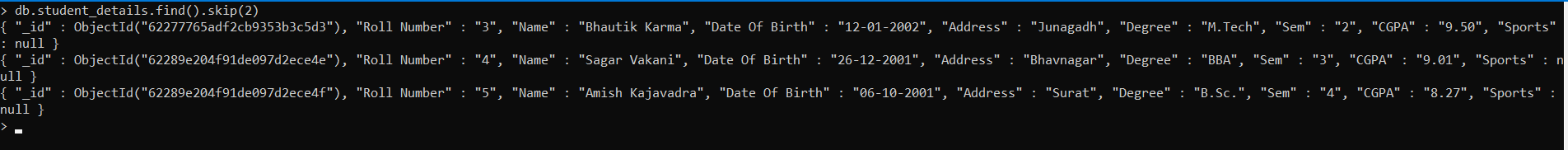


1. **To skip the first two documents from the students collection.**

**Syntax :-** db.<Collection Name>.find().skip(<Number>)

**Command :-** db.student\_details.find().skip(2)

**Output :-**

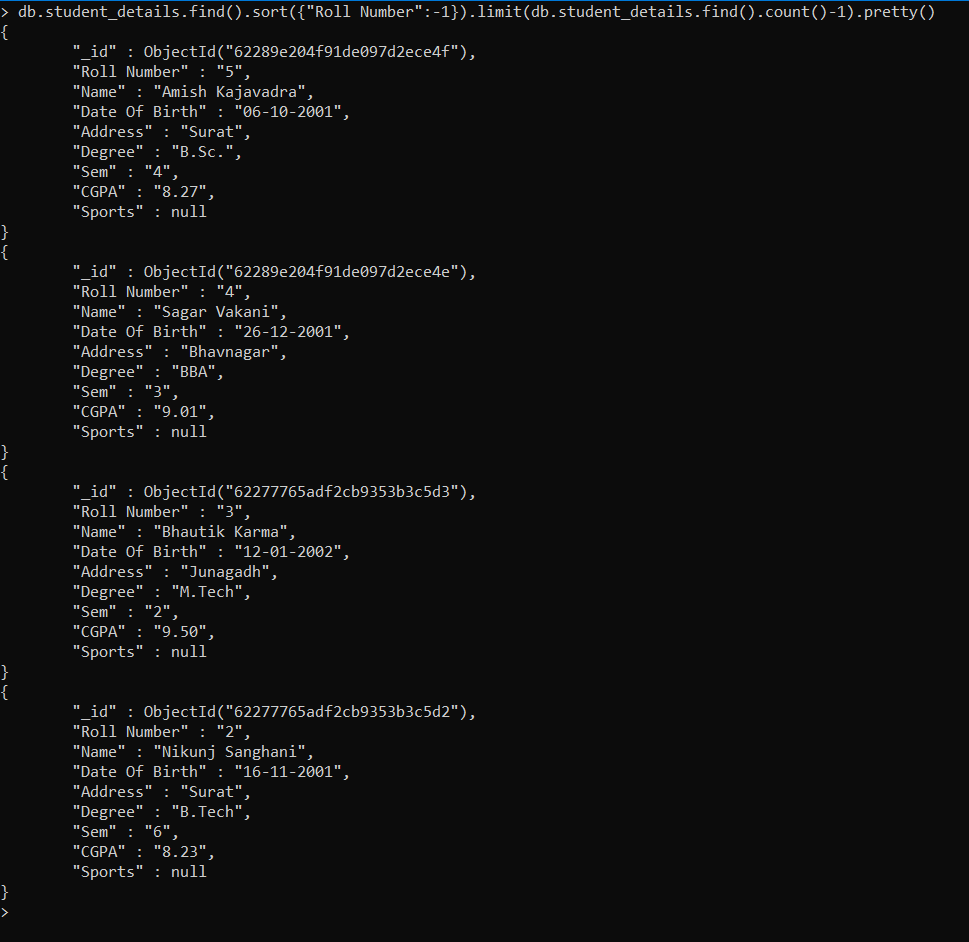


1. **To sort the documents from the Students collection and skip the last document from the output.**

**Syntax:** db.<Collection Name>.find().sort({<Key>:<-1 Or 1>}).limit(<Number>)

**Command:** db.student\_details.find().sort({"Roll Number":-1}).limit(db.student\_details.find().count()-1).pretty()

**Output:**

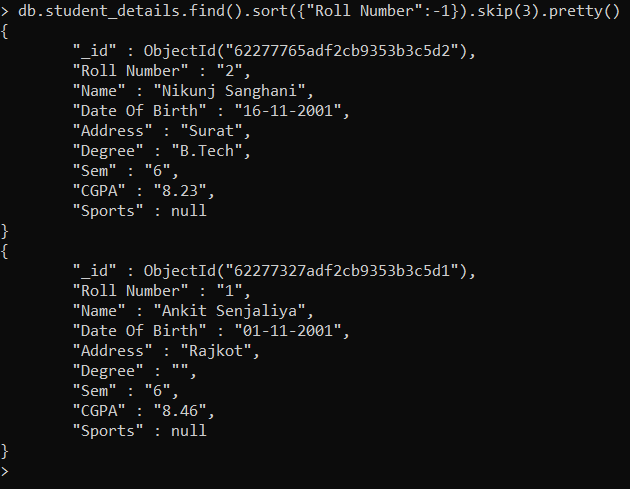


1. **To display the last 3 records from the Students collection.**

**Syntax:** db.<Collection Name>.find().sort({<Key>:<-1 Or 1>}).skip(<Number>)

**Command:** db.student\_details.find().sort({"Roll Number":-1}).skip(3)

**Output:**



1. **To create a collection by the name “student” and then insert documents into the “student” collection. Each document should have a “fruits” array.**

**Syntax:** db.<Collection Name>.update({<Key>:<Value>},{$set{<Key>:[‘<val 1-n>’]}})

**Command:** db.student\_details.update({"Address":"Rajkot"},{$set:{"fruits":["Mango","Apple","Orange"]}})

**Output:**

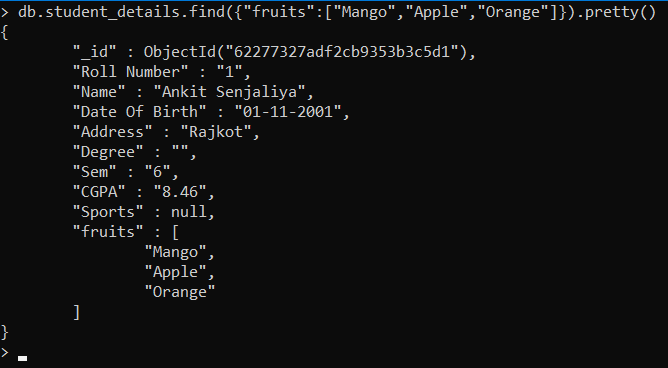


1. **To find those documents from the “student” collection which has the “fruits array” constituted of “banana”, “apple” and “cherry”.**

**Syntax:** db.<Collection Name>.find({<Key>:[‘<val1>’, ‘<val2>’]})

**Command:** db.student\_details.find({"fruits":["Mango","Apple","Orange"]}).pretty()

**Output:**

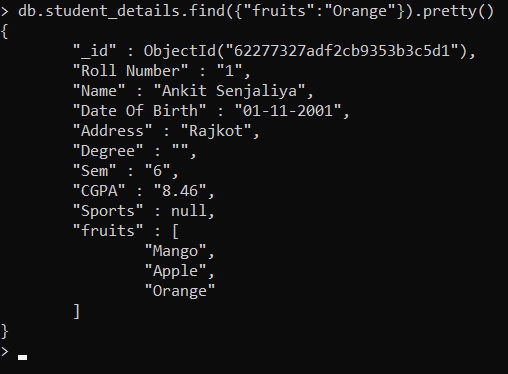


1. **To find those documents from the “student” collection which has the “fruits“ array having “banana “, as an element.**

**Syntax:** db.<Collection Name>.find({<Key>:<Value>]})

**Command:** db.student\_details.find({"fruits":"Orange"}).pretty()

**Output:**

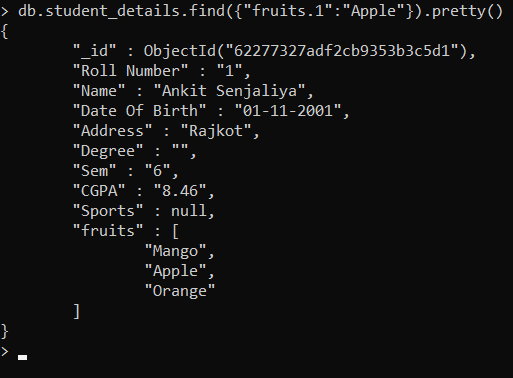


1. **To find those documents from the “student” collection which has the “fruits“array having apple in the first index position begin set 0?**

**Syntax:** db.<Collection Name>.find({<Key.IndexNumber>:<Value>})

**Command:** db.student\_details.find({"fruits.1":"Apple"}).pretty()

**Output:**



1. **To find those documents from the “student” collection where banana is present in the second index position in fruits array.**

**Syntax:** db.<Collection Name>.find({<Key.IndexNumber>:<Value>})

**Command:** db.student\_details.find({"fruits.1":"Banana"}).pretty()

**Output:**

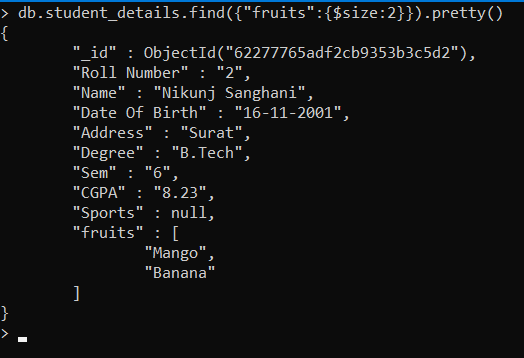


1. **To find those documents from student collection where the size of array is 2. The size implies that the array holds only two values.**

**Syntax:** db.<Collection Name>.find({<Key>:{$size:<Number>}})

**Command:** db.student\_details.find({"fruits":{$size:2}}).pretty()

**Output:**

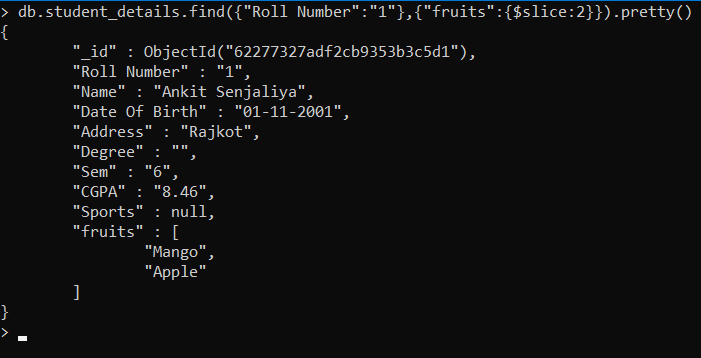


1. **To find those documents with (roll\_no:101) from student collection and the display the first two elements from the array fruits.**

**Syntax:** db.<Collection Name>.find({<Key>:<Value>},{<Key>:$slice:<Number>})

**Command:** db.student\_details.find({"Roll Number":"1"},{"fruits":{$slice:2}}).pretty()

**Output:**



1. **To find the all document form the student collection which have element orange and grapes in the array fruits**

**Syntax:** db.<Collection Name>.find({<Key>:{$all:[‘<Value1>’,‘<Value2>’]}})

**Command:** db.student\_details.find({"fruits":{$all:["Mango","Apple"]}}).pretty()

**Output:**

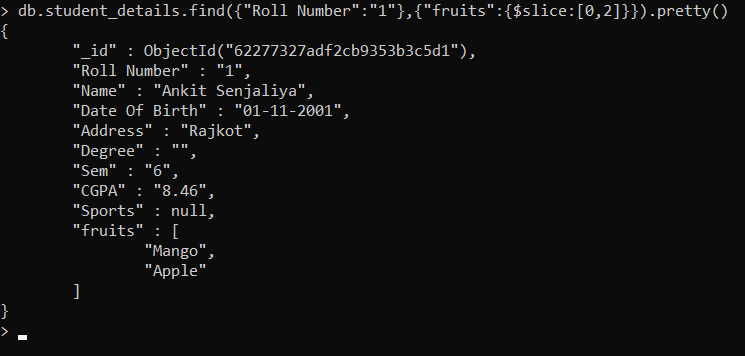


1. **To find documents with (roll\_no:101) from student collection and the display the two elements from the array fruits starting with the element at 0th index position.**

**Syntax:** db.<Collection Name>.find({<Key>:{$slice:[<StartingIndex>,<TotalNumber>]}})

**Command:** db.student\_details.find({"Roll Number":"1"},{"fruits":{$slice:[0,2]}}).pretty()

**Output:**

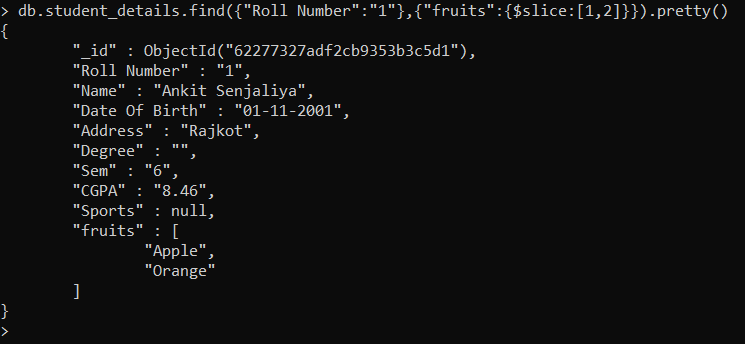


1. **To find documents with (roll\_no:101) from student collection and the display the three elements from the array fruits starting with the element at 2nd index position.**

**Syntax:** db.<Collection Name>.find({<Key>:{$slice:[<StartingIndex>,<TotalNumber>]}})

**Command:** db.student\_details.find({"Roll Number":"1"},{"fruits":{$slice:[1,2]}}).pretty()

**Output:**

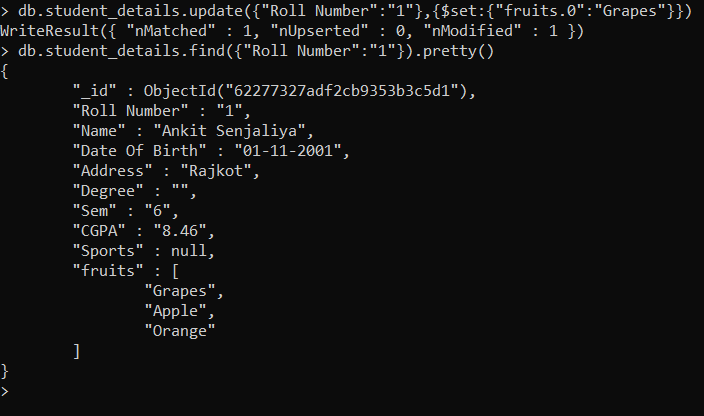


1. **To update the document with roll\_no:104 and replace the element present in the first index position of the fruits array with orange.**

**Syntax:** db.<Collection Name>.update({<Key>:<Value>},{$set:{<Key.Index>}:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1"},{$set:{"fruits.0":"Grapes"}})

**Output:**

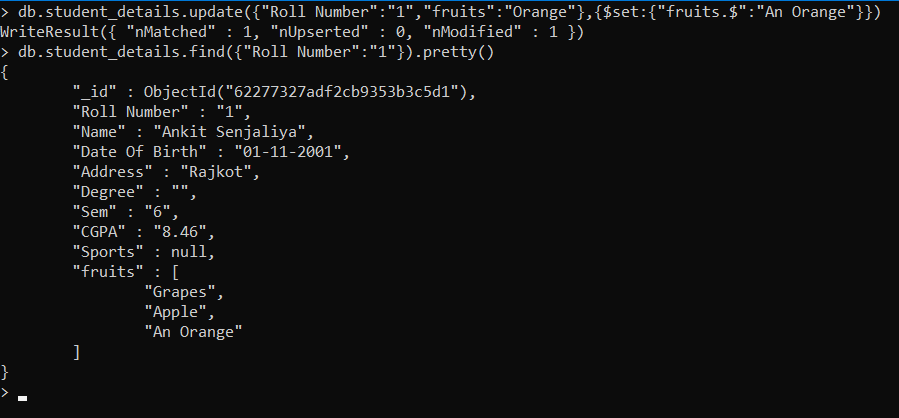


1. **To update the document with roll\_no:101 and replace the element present apple of the fruits array with “an apple”.**

**Syntax:** db.<Collection Name>.update({<Key>:<Value>},{$set:{<Key.$>}:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1","fruits":"Orange"},{$set:{"fruits.$":"An Orange"}})

**Output:**

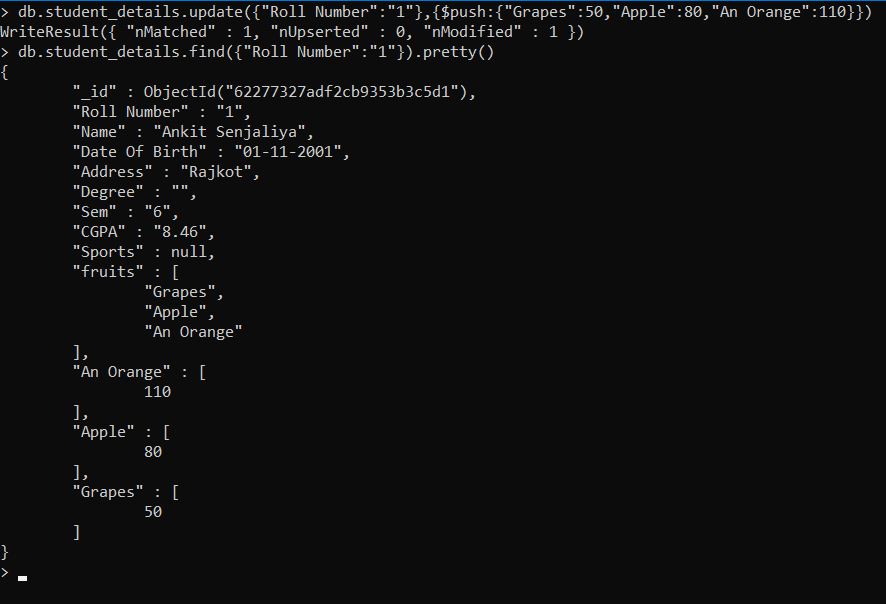


1. **To update the document with roll\_no:104 and push new key value pairs in the fruits array.**

**Syntax:** db.<Collection Name>.update({<Key>,<Value>},$push:{<Key>:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1"},{$push:{"Grapes":50,"Apple":80,"An Orange":110}})

**Output:**

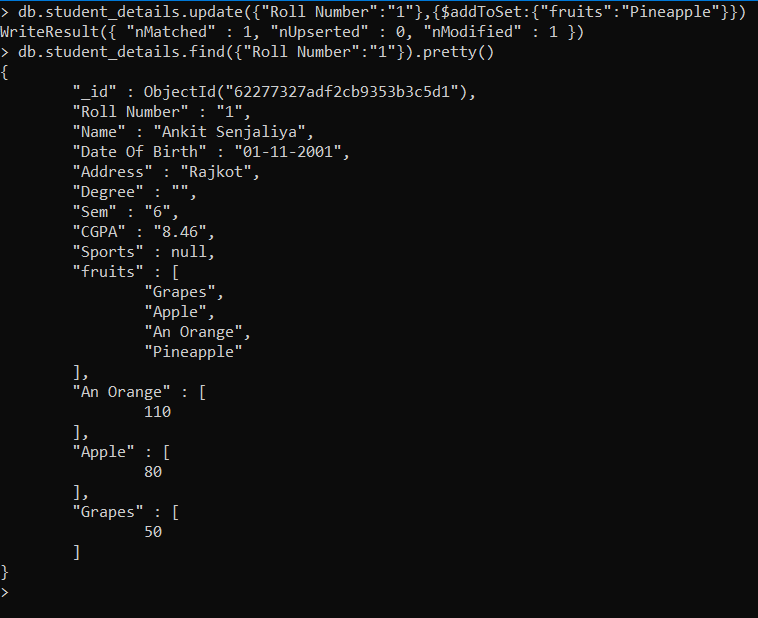


1. **To update the document with roll\_no:104by adding an element orange to the list of element in the array fruit**

**Syntax:** db.<Collection Name>.update({<Key>,<Value>},$addToSet:{<Key>:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1"},{$addToSet:{"fruits":"Pineapple"}})

**Output:**

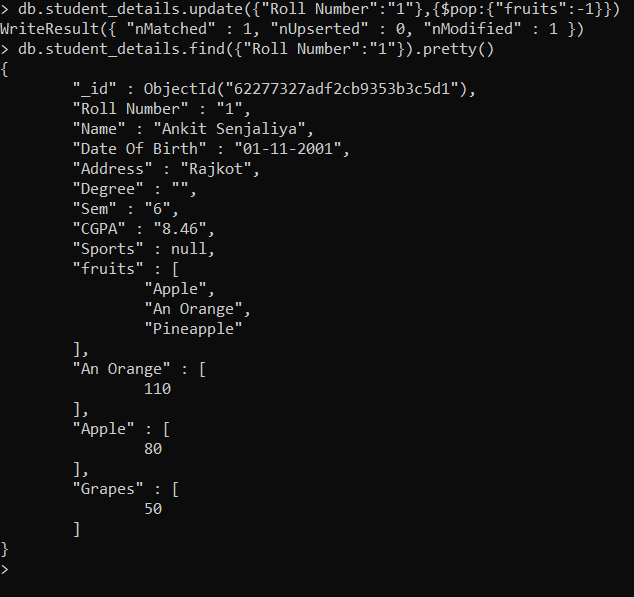


1. **To update the document with roll\_no:104 by popping an element from the list of element in the array fruit. The element the popped is the one from the end of the array.**

**Syntax:** db.<Collection Name>.update({<Key>,<Value>},$pop:{<Key>:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1"},{$pop:{"fruits":-1}})

**Output:**



1. **To update the document with roll\_no:104 by popping an element from the list of element in the array fruit. The element the popped is the one from the beginning of the array.**

**Syntax:** db.<Collection Name>.update({<Key>,<Value>},$pop:{<Key>:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1"},{$pop:{"fruits":1}})

**Output:**

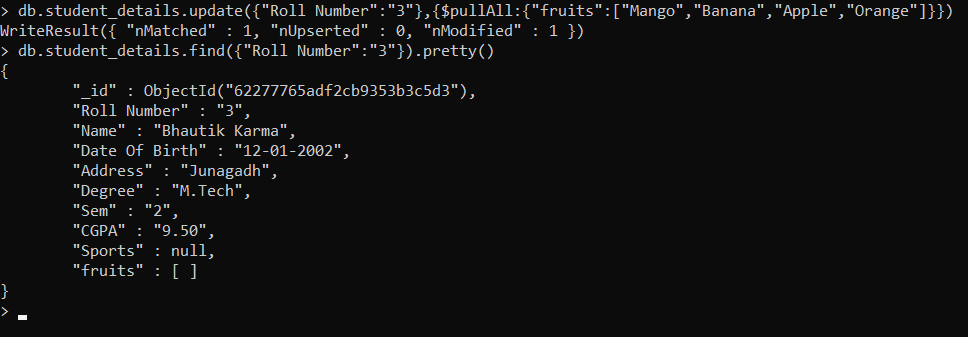


1. **To update the document with roll\_no:103 by popping two elements an element from the list of element in the array fruit.**

**Syntax:** db.<Collection Name>.update({<Key>,<Value>},$pullAll:{<Key>:[<Values>]})

**Command:** db.student\_details.update({"Roll Number":"3"},{$pullAll:{"fruits":["Mango","Banana","Apple","Orange"]}})

**Output:**



1. **To update the document having banana as an element in the array fruits and popped out element banana from those document**

**Syntax:** db.<Collection Name>.update({<Key>,<Value>},$pull:{<Key>:<Value>})

**Command:** db.student\_details.update({"Roll Number":"1"},{$pull:{"fruits":"Apple"}})

**Output:**



1. **To pull out an array element based on index position.**

**Syntax:**

db.<Collection Name>.update({<Key>,<Value>},$unset:{<Key>:null})

db.<Collection Name>.update({<Key>,<Value>},$pull:{<Key>:null})

**Command:** db.student\_details.update({"Roll Number":"1"},{$pull:{"fruits":null}})

**Output:**

