Practical 1
MongoDB Basics

1) write a MongoDB query to create and drop database.

```
https://docs.mongodb.com/mongodb-shell/install/
===============
---
The server generated these startup warnings when booting:
        2022-10-25T13:17:25.163+05:30: Access control is not enabled for the database. Read and write access to data and
 configuration is unrestricted
---
---
        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()
---
> show dbs
Employee  0.000GB
TYITDB    0.000GB
admin     0.000GB
config    0.000GB
local     0.000GB
test      0.000GB
> use TYITDB
switched to db TYITDB
> db.dropDatabase()
{ "ok" : 1 }
>
```

2) write a MongoDB query to create , display and drop collection.

```
> use TYITDB
switched to db TYITDB
> db.createCollection("Kartik")
{ "ok" : 1 }
> db.Kartik.drop()
true
>
```

3) write a MongoDB query to insert, query, update and delete a document.

```
> db.Kartik.insert({course:"TYBScIT",College:"MCC",rollno:229726})
WriteResult({ "nInserted" : 1 })
> db.Kartik.find().pretty()
{
        "_id" : ObjectId("635a49b61e1a4387977b4d37"),
        "course" : "TYBScIT",
        "College" : "MCC",
        "rollno" : 229726
}
> db.Kartik.update({},{$set:{College:"Mulund College Of Commerce"}},{})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Kartik.find().pretty()
{
        "_id" : ObjectId("635a49b61e1a4387977b4d37"),
        "course" : "TYBScIT",
        "College" : "Mulund College Of Commerce",
        "rollno" : 229726
}
> db.Kartik.remove({})
WriteResult({ "nRemoved" : 1 })
> db.Kartik.find().pretty()
>
```

Practical 2
Simple Queries with MongoDB

Q1) Insert and Display the Documents in the Employee Collection.

```
> db.employee.find().pretty()
{
        "_id" : ObjectId("630369517e6e1b7c113b70fc"),
        "EMPID" : 1,
        "ENAME" : "Kartik Sharma",
        "SAL" : 100000,
        "CITY" : "MULUND",
        "DNAME" : "Testing"
}
{
        "_id" : ObjectId("63036a197e6e1b7c113b70fd"),
        "EMPID" : 2,
        "ENAME" : "ELON MUSK",
        "SAL" : 10000000,
        "CITY" : "EUROP",
        "DNAME" : "Software Developer"
}
{
        "_id" : ObjectId("63036a557e6e1b7c113b70fe"),
        "EMPID" : 3,
        "ENAME" : "BRIL GATES",
        "SAL" : 100000,
        "CITY" : "THANE",
        "DNAME" : "Software Engineering"
}
{
        "_id" : ObjectId("63036b167e6e1b7c113b70ff"),
        "EMPID" : 4,
        "ENAME" : "Mukesh Ambani",
        "SAL" : 200000,
        "CITY" : "KURLA",
        "DNAME" : "Self Employee"
```

```
{
        "_id" : ObjectId("63036b5d7e6e1b7c113b7100"),
        "EMPID" : 5,
        "ENAME" : "Aakash Setty",
        "SAL" : 20000,
        "CITY" : "KURLA",
        "DNAME" : "Sales Man"
}
>
```

Q2) Display Details of employees who live in Mulund.

```
> db.employee.find({CITY:"MULUND"}).pretty()
{
        "_id" : ObjectId("630369517e6e1b7c113b70fc"),
        "EMPID" : 1,
        "ENAME" : "Kartik Sharma",
        "SAL" : 100000,
        "CITY" : "MULUND",
        "DNAME" : "Testing"
}
>
```

Q3) Display Details of employee staying in Thane and salary greater than 50000.

```
> db.employee.find({CITY:"THANE",SAL:{$gt:50000}}).pretty()
{
        "_id" : ObjectId("63036a557e6e1b7c113b70fe"),
        "EMPID" : 3,
        "ENAME" : "BRIL GATES",
        "SAL" : 100000,
        "CITY" : "THANE",
        "DNAME" : "Software Engineering"
}
>
```

Q4) Display details of the employee whose salary is greater than equal to 50000 but less than 100000.db

```
> db.employee.find({SAL:{$gte:50000},SAL:{$lt:100000}}).pretty()
{
        "_id" : ObjectId("63036b5d7e6e1b7c113b7100"),
        "EMPID" : 5,
        "ENAME" : "Aakash Setty",
        "SAL" : 20000,
        "CITY" : "KURLA",
        "DNAME" : "Sales Man"
}
>
```

Q5) Display Details of employee staying in either thane or Mulund and salary Greater than 50000.

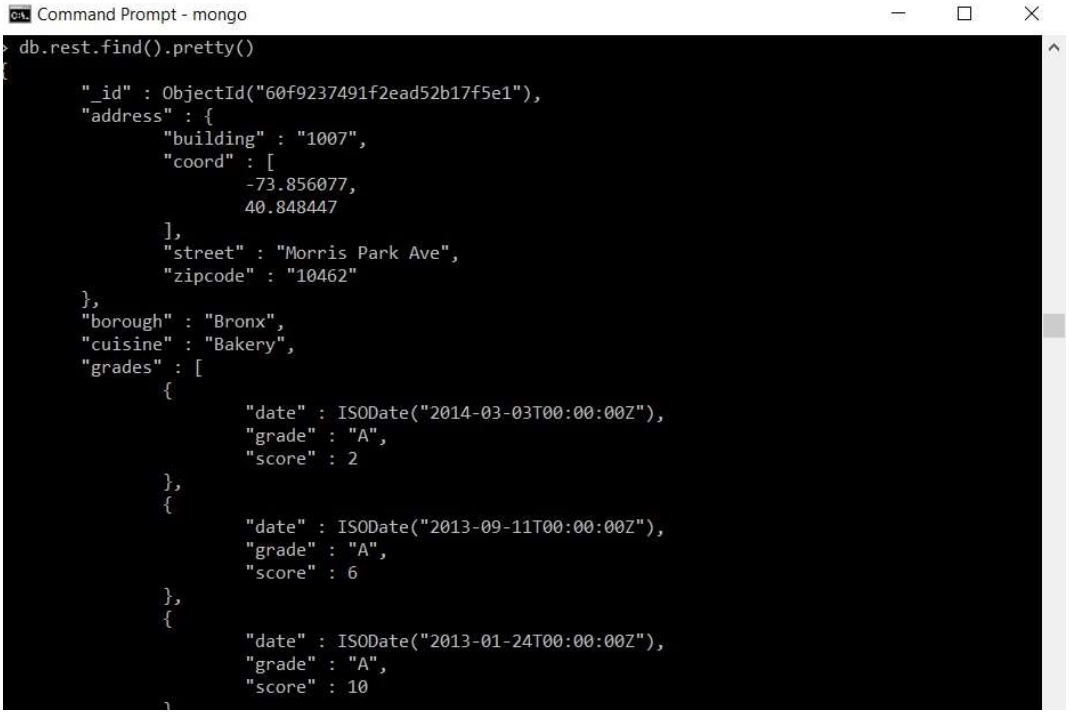
```
> db.employee.find({SAL:{$gt:50000},$or:[{CITY:"THANE"},{CITY:"MULUND"}]}).pretty()
{
        "_id" : ObjectId("630369517e6e1b7c113b70fc"),
        "EMPID" : 1,
        "ENAME" : "Kartik Sharma",
        "SAL" : 100000,
        "CITY" : "MULUND",
        "DNAME" : "Testing"
}
{
        "_id" : ObjectId("63036a557e6e1b7c113b70fe"),
        "EMPID" : 3,
        "ENAME" : "BRIL GATES",
        "SAL" : 100000,
        "CITY" : "THANE",
        "DNAME" : "Software Engineering"
}
```

Q6) Display Details of the employee whose salary is not equal to 50000 and working Either testing or Software Developer department.

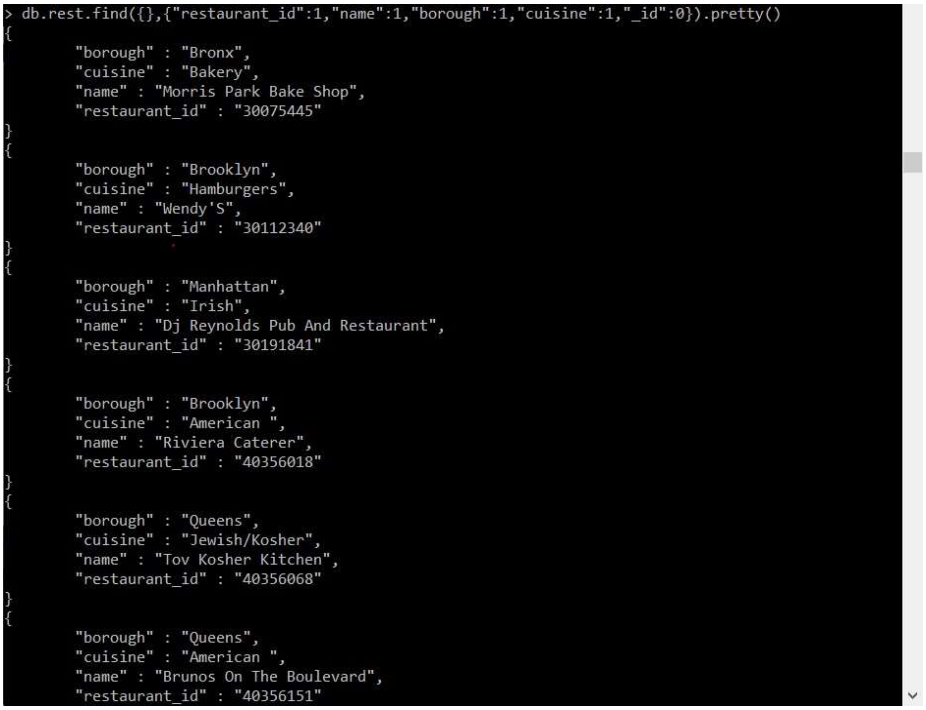
```
> db.employee.find({SAL:{$ne:50000},$or:[{DNAME:"Testing"},{DNAME:"Software Developer"}]}).pretty()
{
        "_id" : ObjectId("630369517e6e1b7c113b70fc"),
        "EMPID" : 1,
        "ENAME" : "Kartik Sharma",
        "SAL" : 100000,
        "CITY" : "MULUND",
        "DNAME" : "Testing"
}
{
        "_id" : ObjectId("63036a197e6e1b7c113b70fd"),
        "EMPID" : 2,
        "ENAME" : "ELON MUSK",
        "SAL" : 10000000,
        "CITY" : "EUROP",
        "DNAME" : "Software Developer"
}
>
```

Practical 2A – Basic Queries

1) Write a MongoDB query to display all the documents in the collection test.

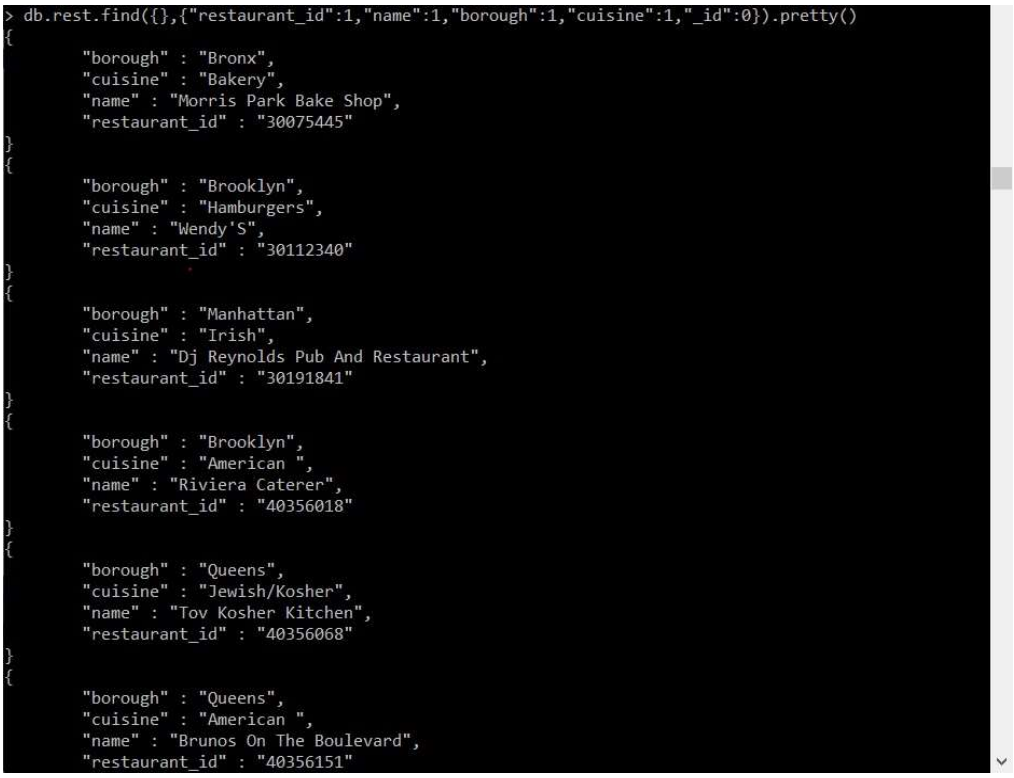


2) Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection rest.



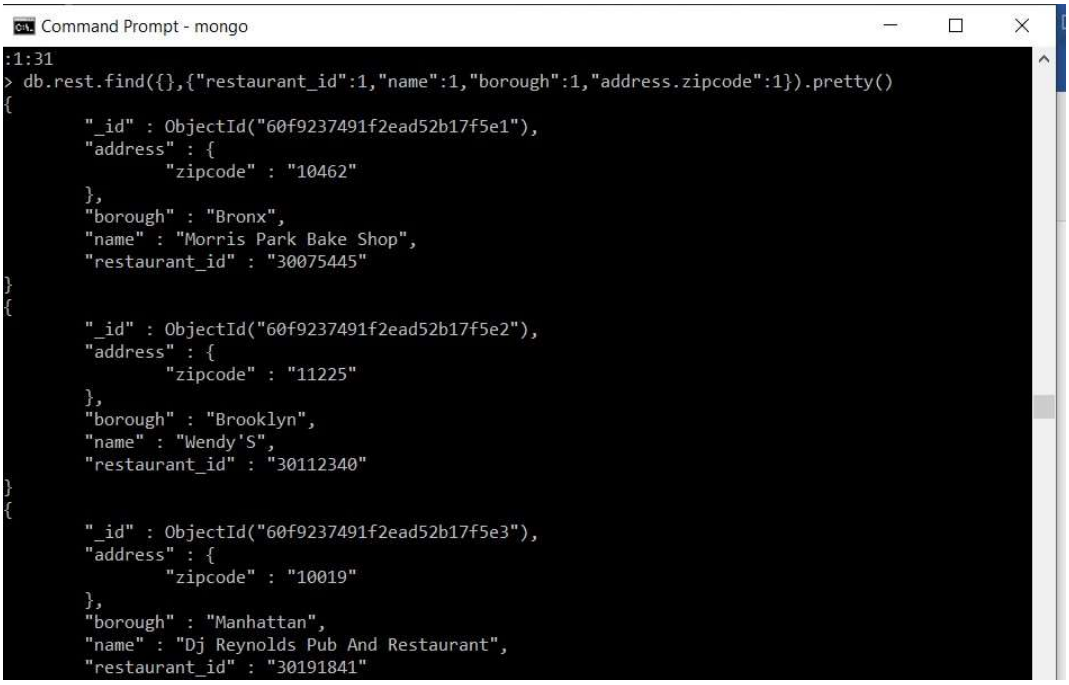
```
> db.rest.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine":1,"_id":0}).pretty()
{
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Hamburgers",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "borough" : "Queens",
        "cuisine" : "Jewish/Kosher",
        "name" : "Tov Kosher Kitchen",
        "restaurant_id" : "40356068"
}
{
        "borough" : "Queens",
        "cuisine" : "American ",
        "name" : "Brunos On The Boulevard",
        "restaurant_id" : "40356151"
```

3) Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine, but exclude the field _id for all the documents in the collection rest.

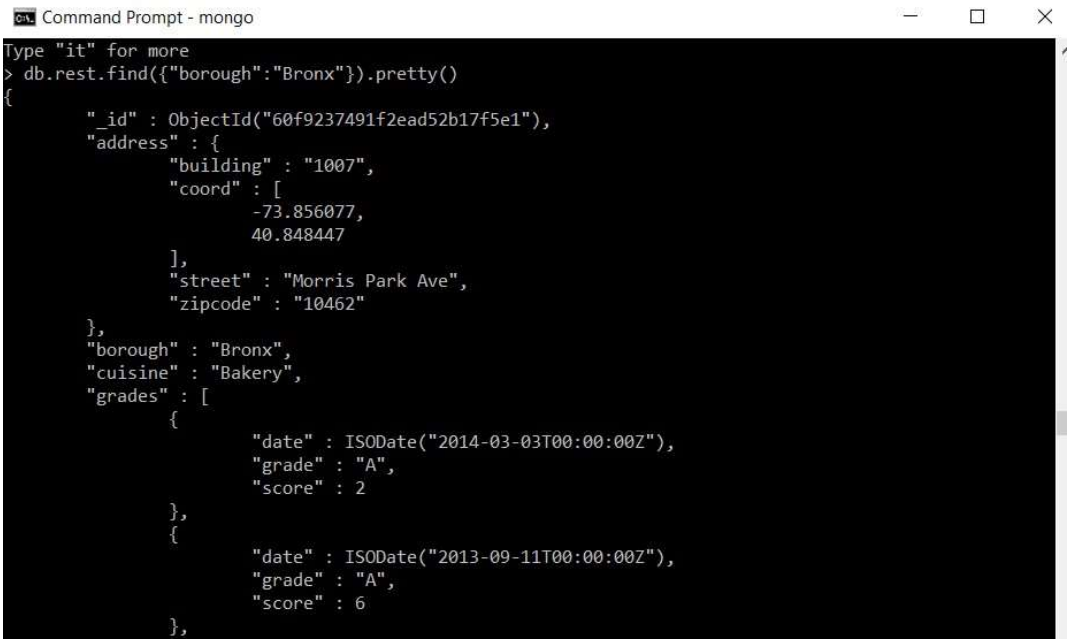


```
> db.rest.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine":1,"_id":0}).pretty()
{
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Hamburgers",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "borough" : "Queens",
        "cuisine" : "Jewish/Kosher",
        "name" : "Tov Kosher Kitchen",
        "restaurant_id" : "40356068"
}
{
        "borough" : "Queens",
        "cuisine" : "American ",
        "name" : "Brunos On The Boulevard",
        "restaurant_id" : "40356151"
```

4) Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collectionrest.



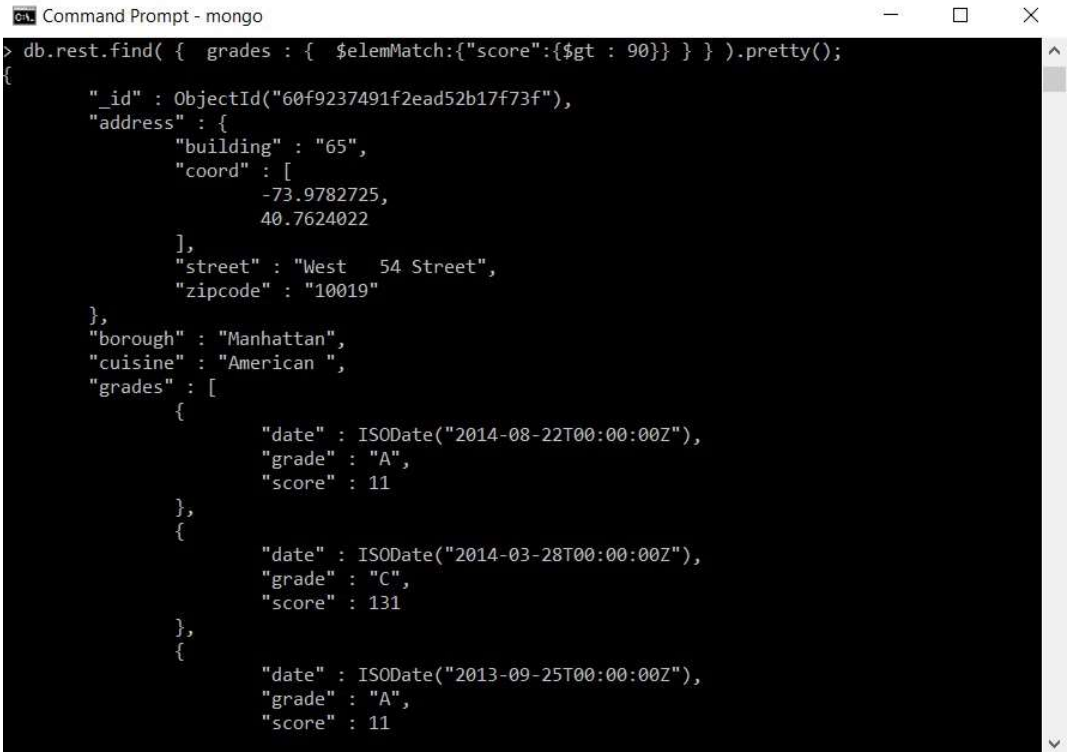
5) Write a MongoDB query to display all the rest which is in the borough is Bronx.



6) Write a MongoDB query to display the first 5 records which is in the borough Bronx.

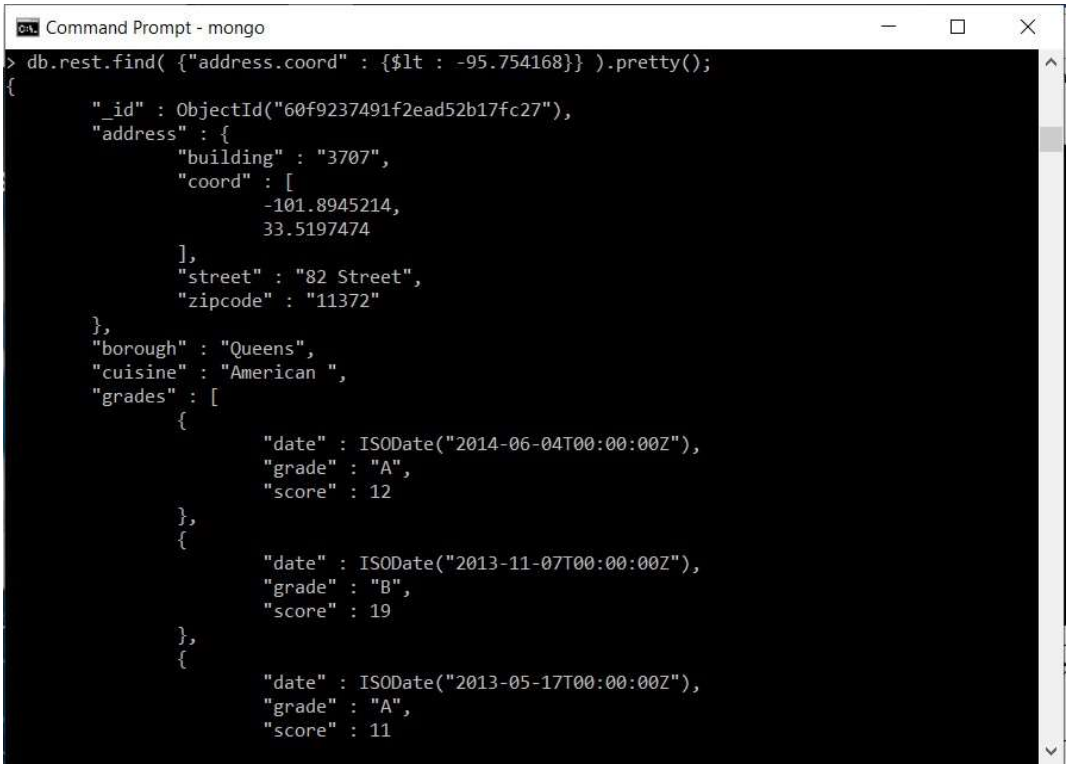
7) Write a MongoDB query to display the next 5 rest after skipping first 5 which are in the borough Bronx.

8) Write a MongoDB query to find the rest who achieved a score more than 90.

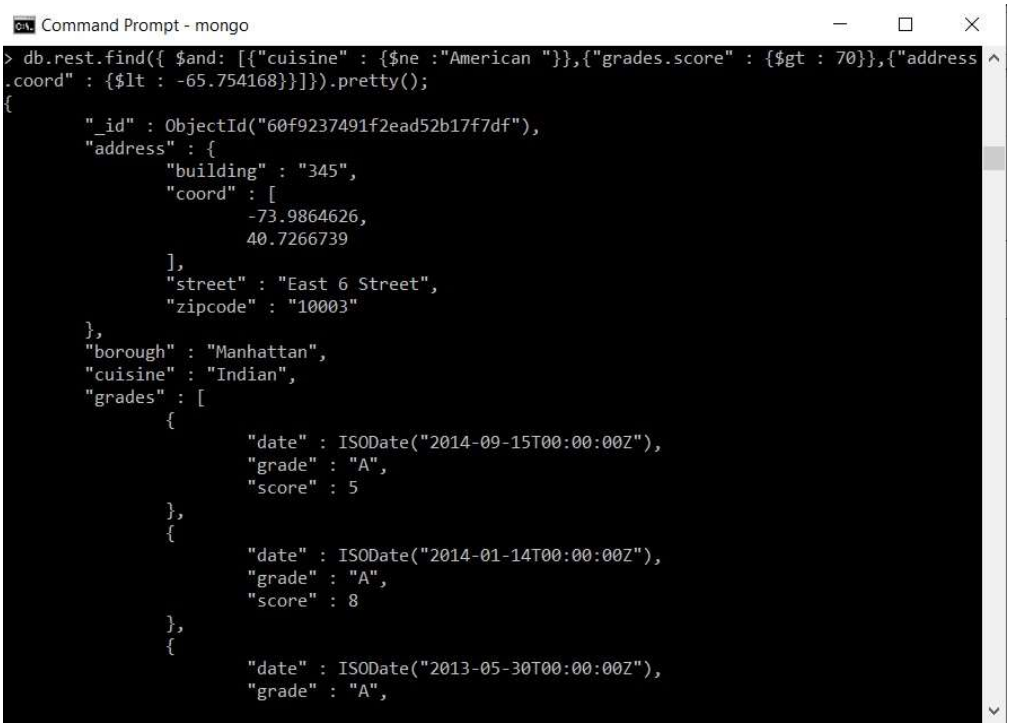


9) Write a MongoDB query to find the rest that achieved a score, more than 80 but less than 100.

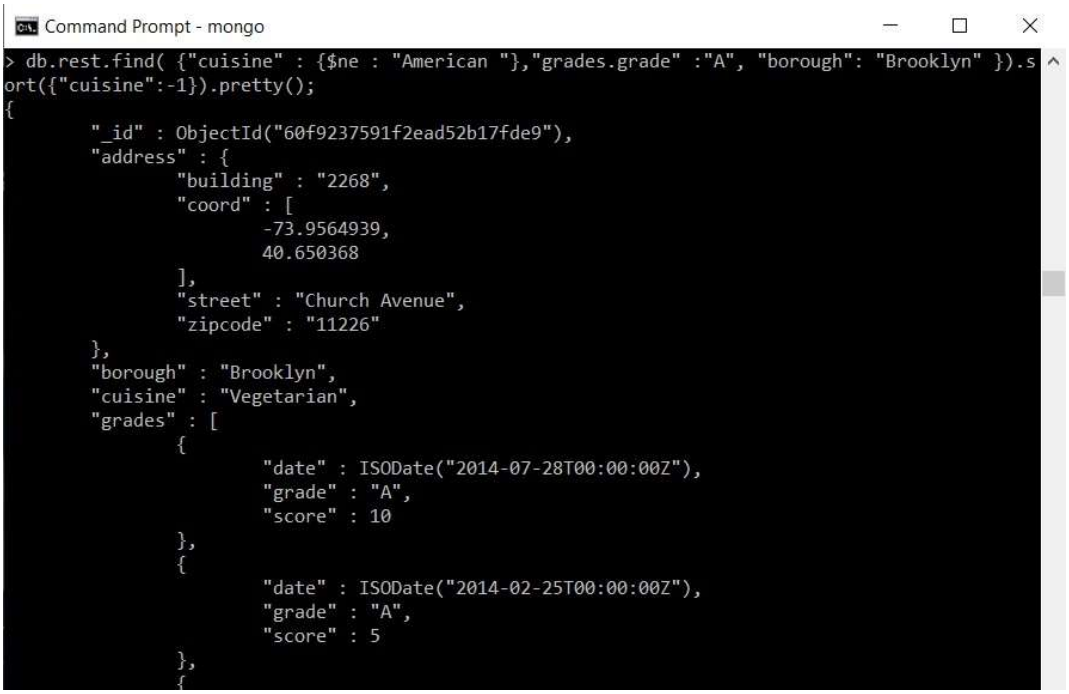
10) Write a MongoDB query to find the rest which locate in latitude value less than - 95.754168.



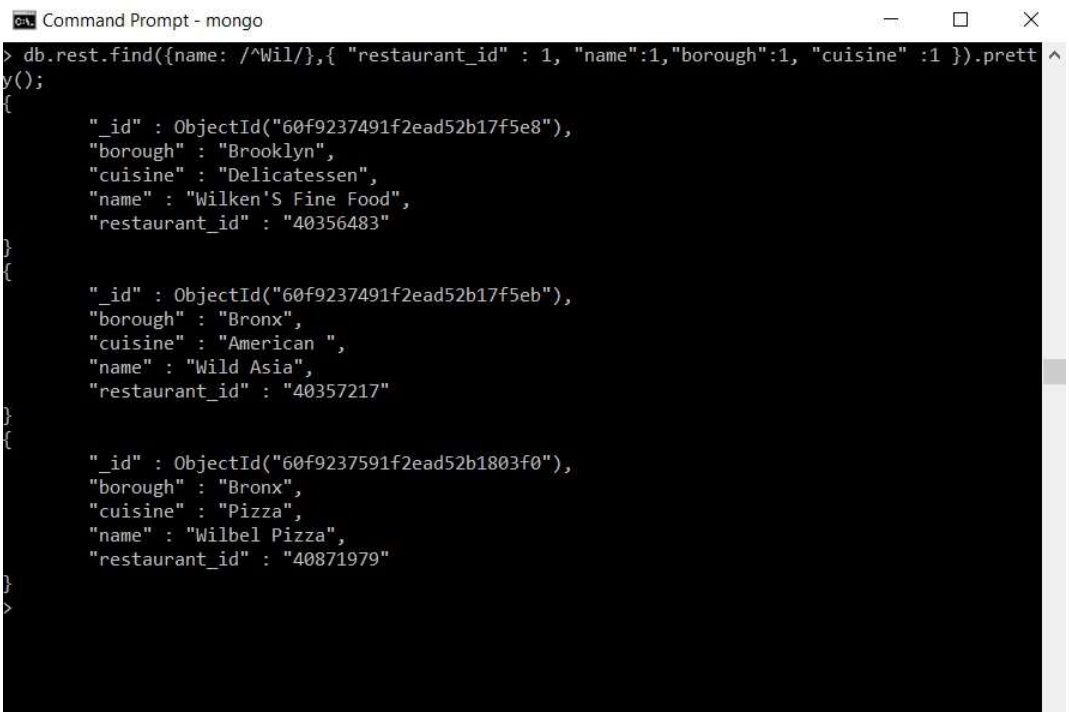
11) Write a MongoDB query to find the rest that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than 65.754168



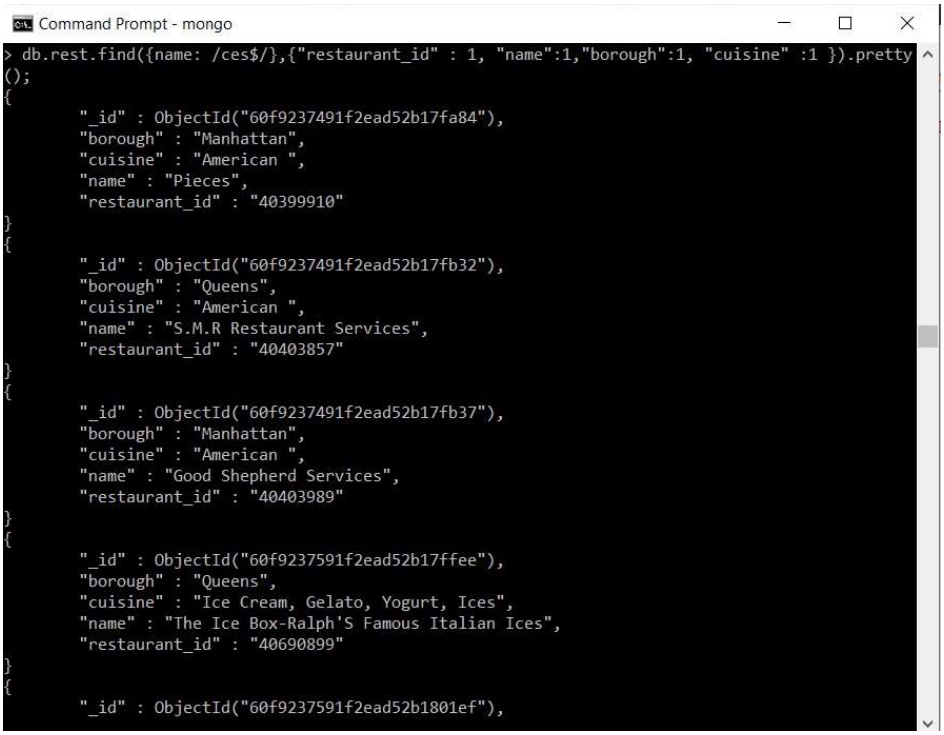
12) Write a MongoDB query to find the rest which do not prepare any cuisine of 'American ' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.



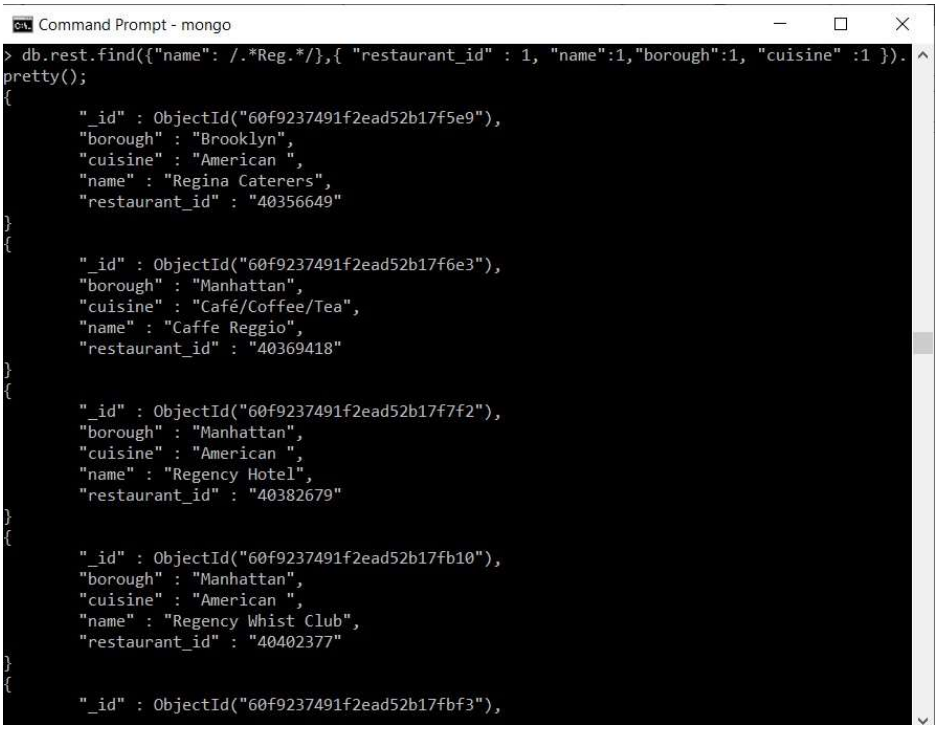
13) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which contain 'Wil' as first three letters for its name.



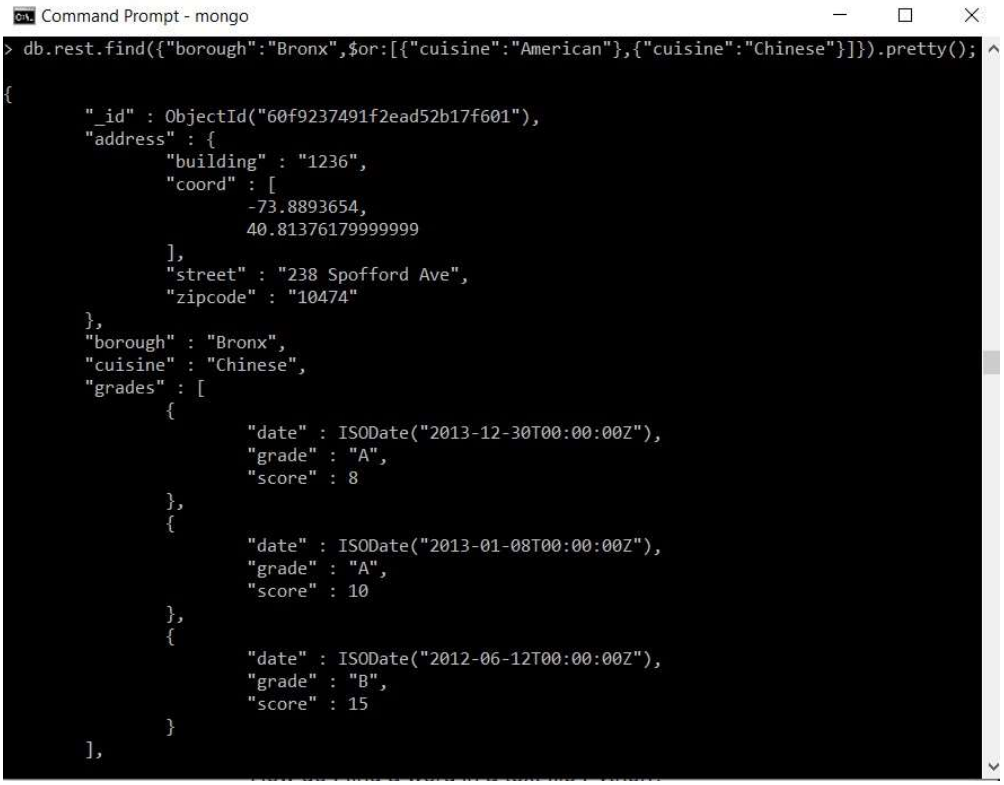
14) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which contain 'ces' as last three letters for its name.



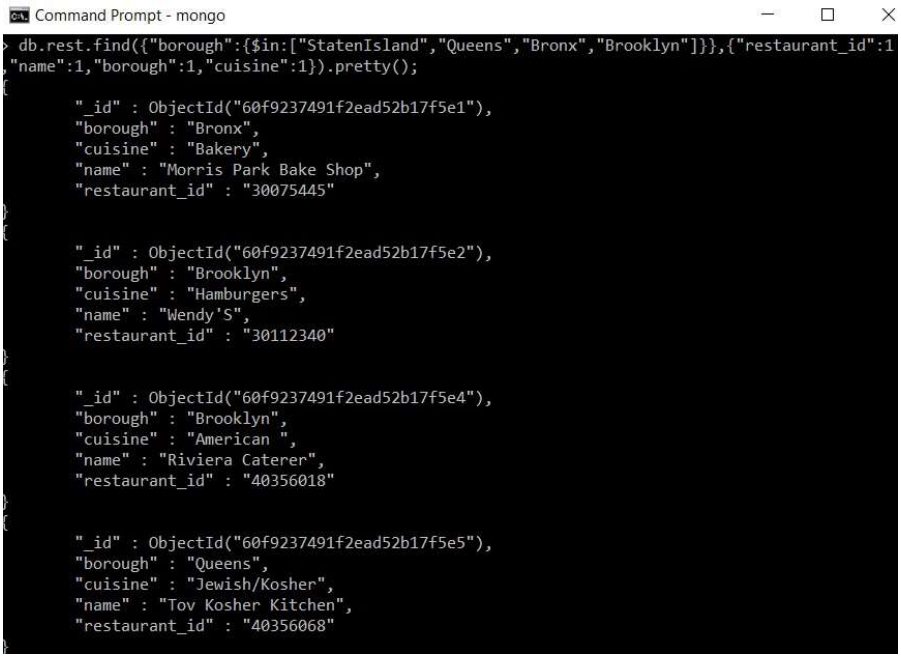
15) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which contain 'Reg' as three letters somewhere in its name.



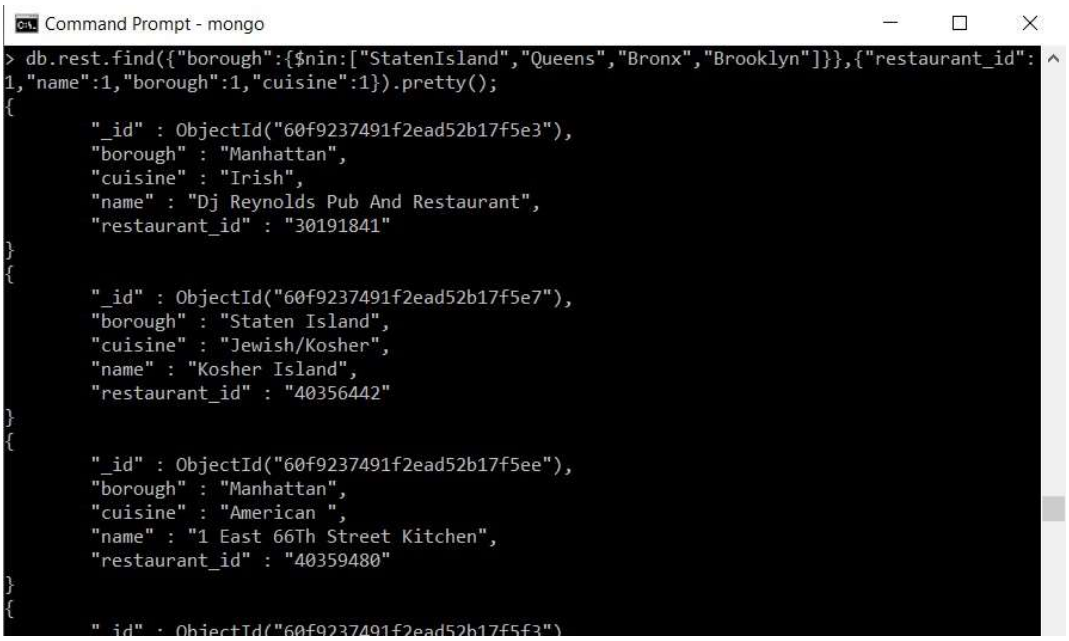
16) Write a MongoDB query to find the rest which belong to the borough Bronx and prepared either American or Chinese dish.



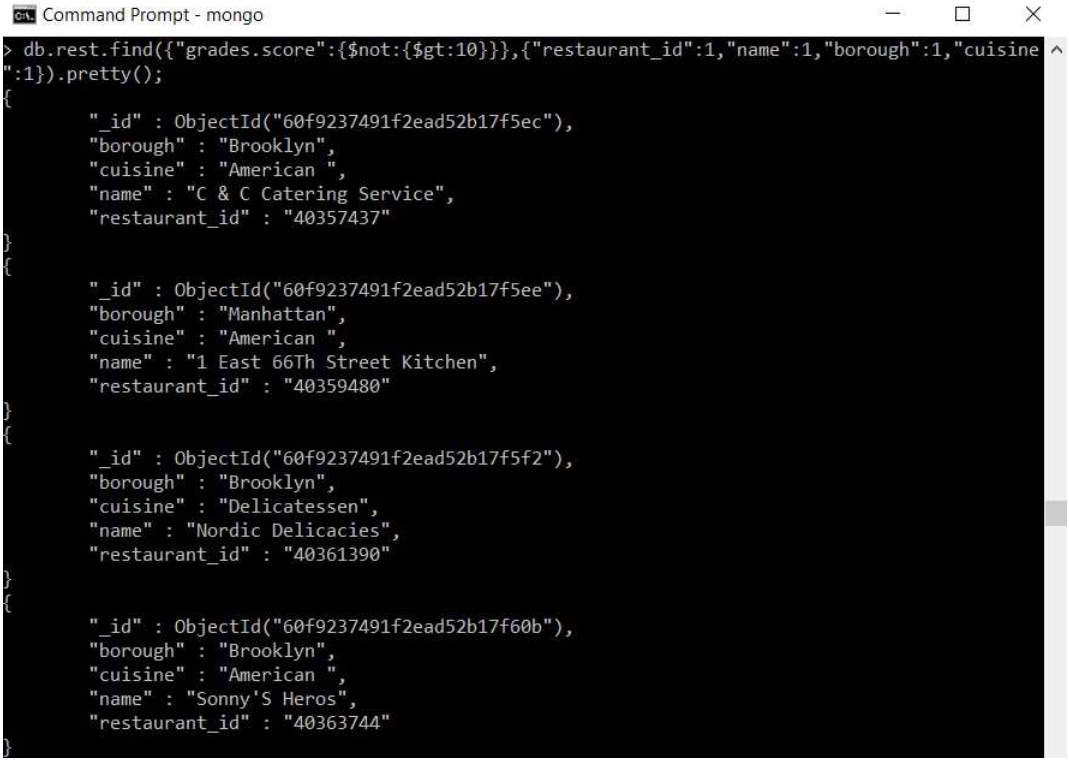
17) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which belong to the borough Staten Island or Queens or Bronxor Brooklyn .



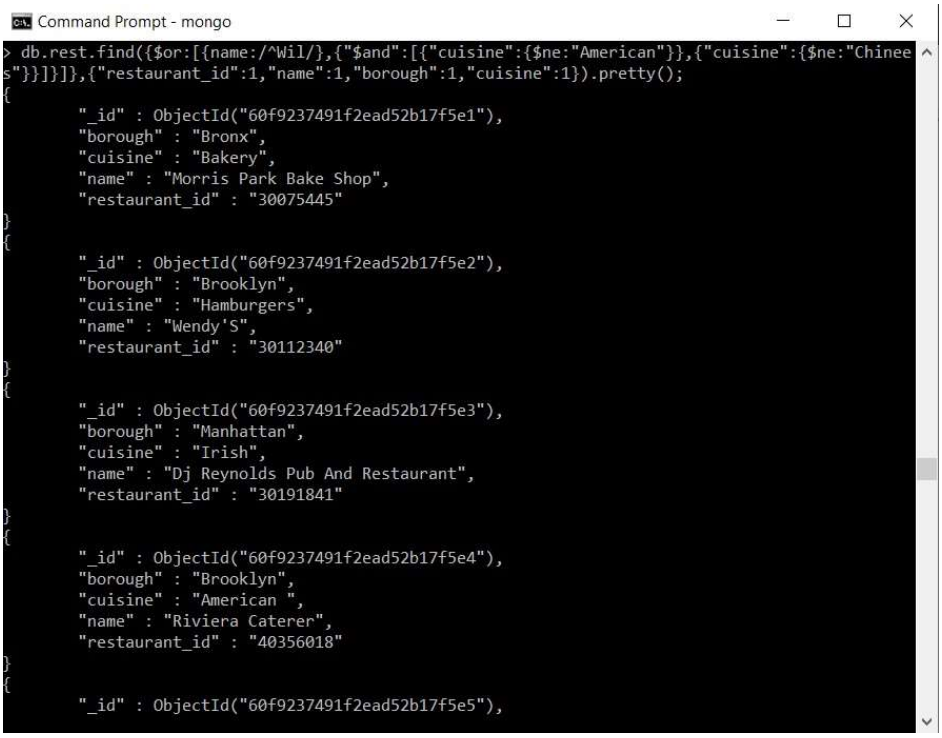
18) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.



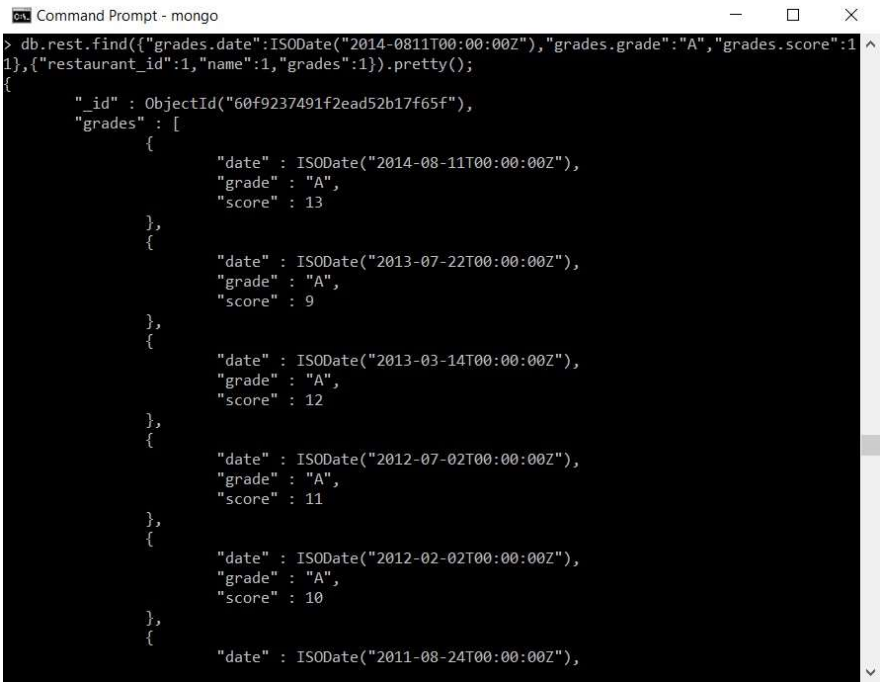
19) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which achieved a score which is not more than 10.



20) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those rest which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'



21) Write a MongoDB query to find the restaurant Id, name, and grades for those rest which achieved a grade of "A" and scored 11 on an ISODate "201408-11T00:00:00Z" among many of survey dates.



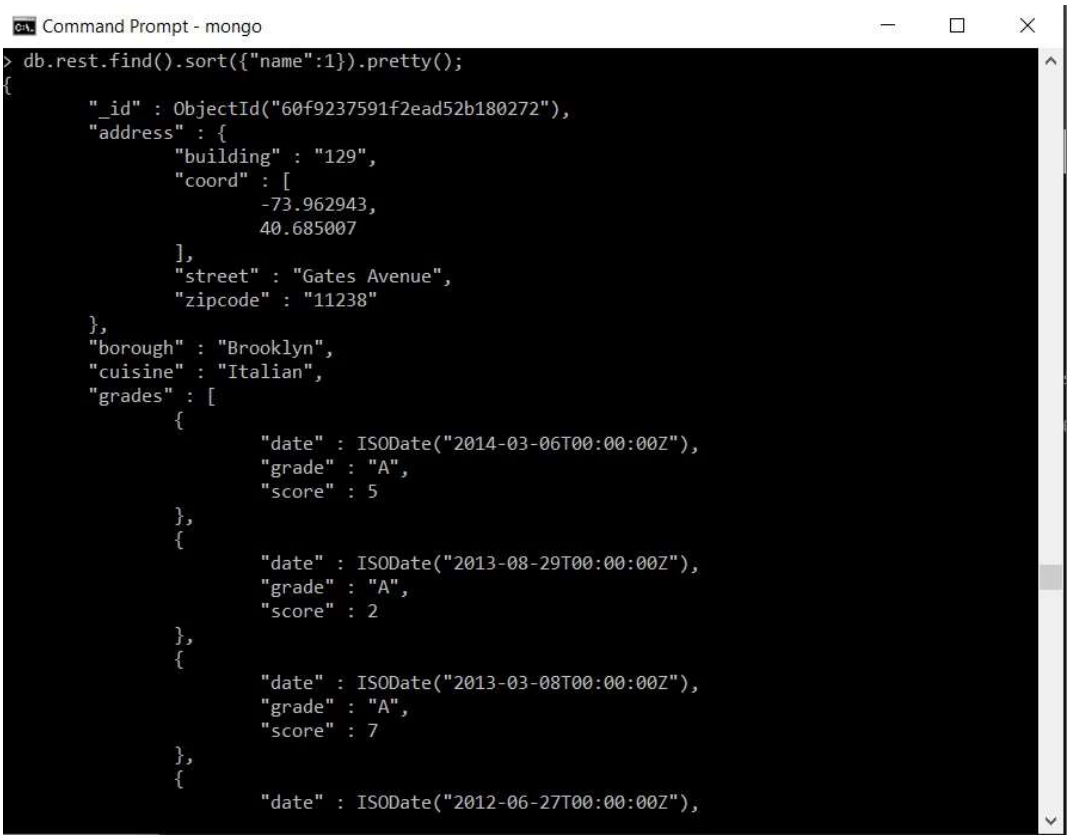
22) Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

```
> db.rest.find({"grades.1.date":ISODate("2014-0811T00:00:00Z"),"grades.1.grade":"A","grades.1.score":9},
{"restaurant_id":1,"name":1,"grades":1}).pretty()
{
        "_id" : ObjectId("60f9237491f2ead52b17fc0b"),
        "grades" : [
                {
                        "date" : ISODate("2015-01-12T00:00:00Z"),
                        "grade" : "A",
                        "score" : 10
                },
                {
                        "date" : ISODate("2014-08-11T00:00:00Z"),
                        "grade" : "A",
                        "score" : 9
                },
                {
                        "date" : ISODate("2014-01-14T00:00:00Z"),
                        "grade" : "A",
                        "score" : 13
                },
                {
                        "date" : ISODate("2013-02-07T00:00:00Z"),
                        "grade" : "A",
                        "score" : 10
                },
                {
                        "date" : ISODate("2012-04-30T00:00:00Z"),
                        "grade" : "A",
                        "score" : 11
                }
        ],
        "name" : "Club Macanudo (Cigar Bar)",
        "restaurant_id" : "40526406"
}
>
```

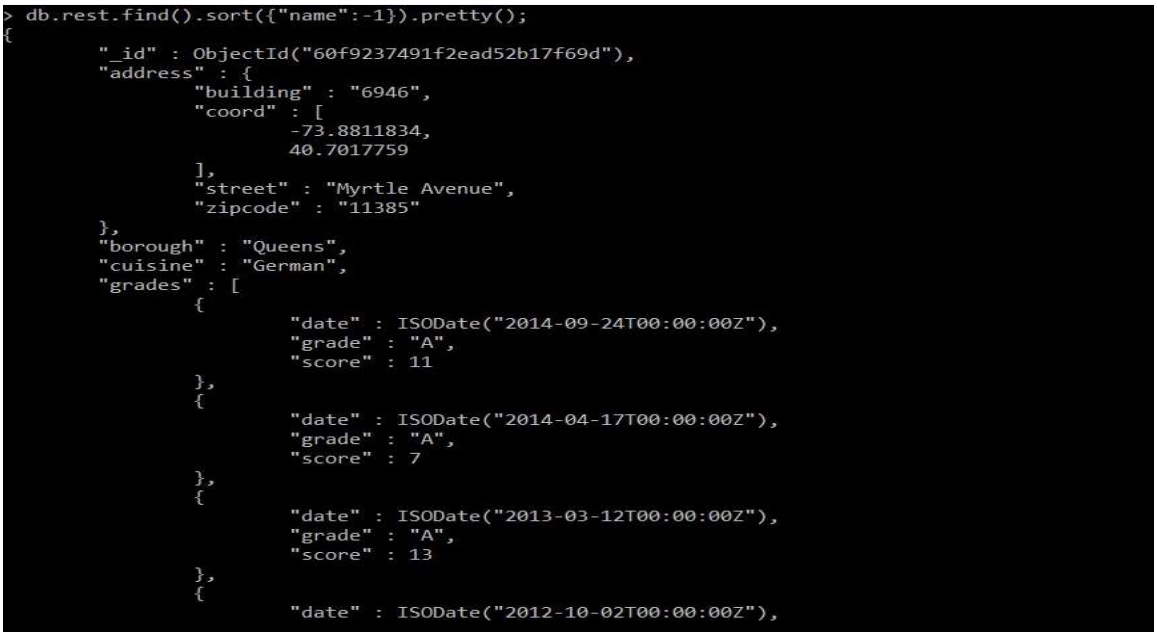
23) Write a MongoDB query to find the restaurant Id, name, address and geographical location for those rest where 2nd element of coord array contains a value which is more than 42 and upto 52.

```
Select Command Prompt - mongo
> db.rest.find({"address.coord.1":{$gt:42,$lte:52}},{"restaurant_id":1,"name":1,"address":1,"coo
rd":1}).pretty();
{
        "_id" : ObjectId("60f9237491f2ead52b17f881"),
        "address" : {
                "building" : "47",
                "coord" : [
                        -78.877224,
                        42.89546199999999
                ],
                "street" : "Broadway @ Trinity Pl",
                "zipcode" : "10006"
        },
        "name" : "T.G.I. Friday'S",
        "restaurant_id" : "40387990"
}
{
        "_id" : ObjectId("60f9237491f2ead52b17f8b4"),
        "address" : {
                "building" : "1",
                "coord" : [
                        -0.7119979,
                        51.6514664
                ],
                "street" : "Pennplaza E, Penn Sta",
                "zipcode" : "10001"
        },
        "name" : "T.G.I. Fridays",
        "restaurant_id" : "40388936"
}
{
        "_id" : ObjectId("60f9237491f2ead52b17fb07"),
```

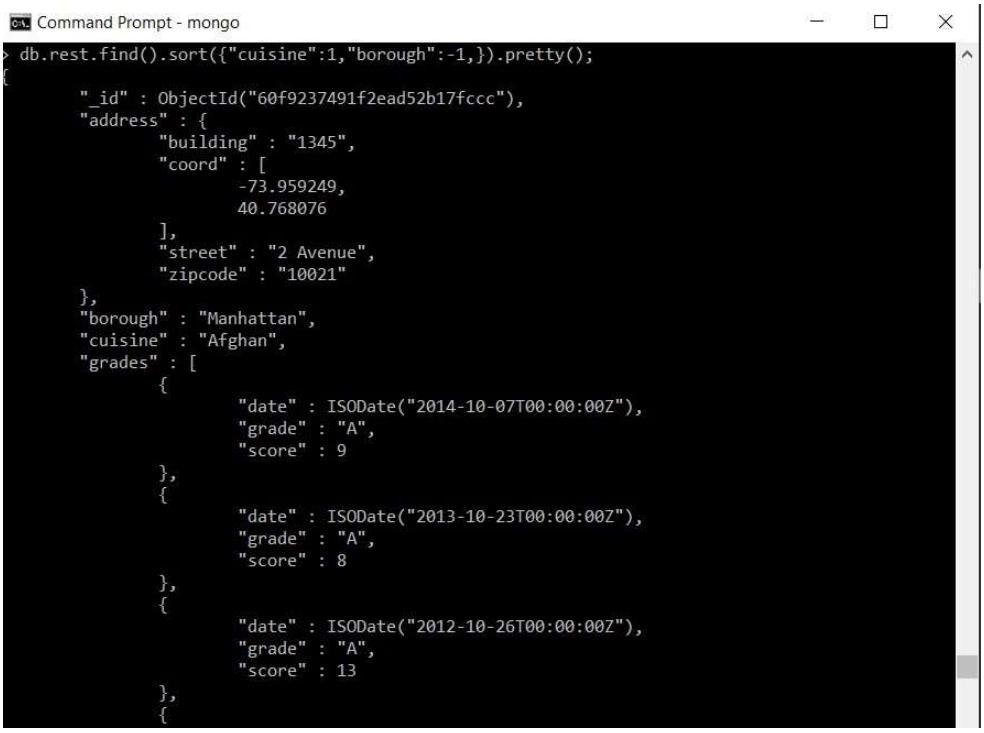
24) Write a MongoDB query to arrange the name of the rest in ascending order along with all the columns.



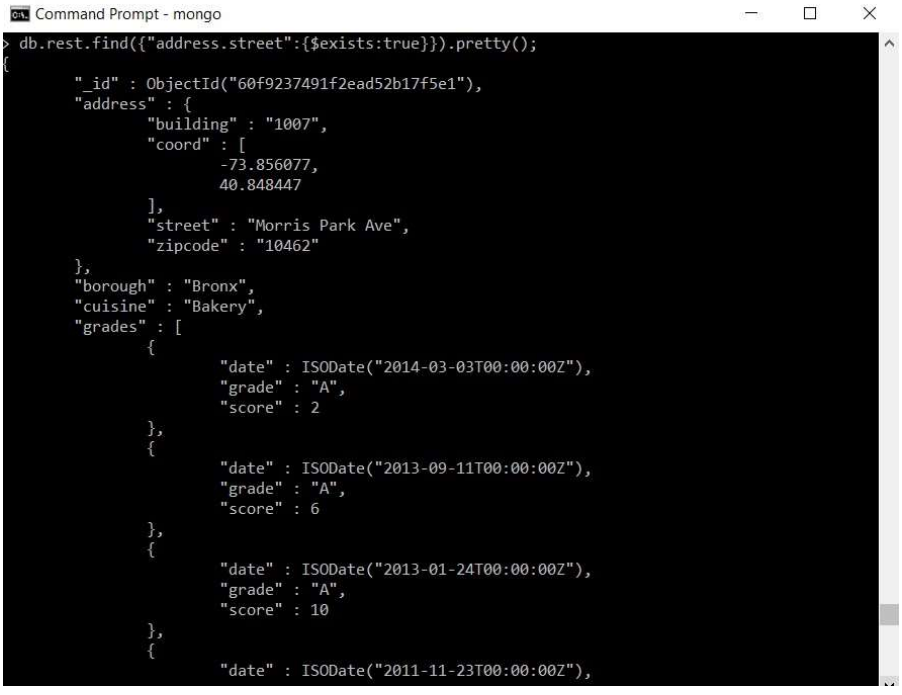
25) Write a MongoDB query to arrange the name of the rest in descending along with all the columns.



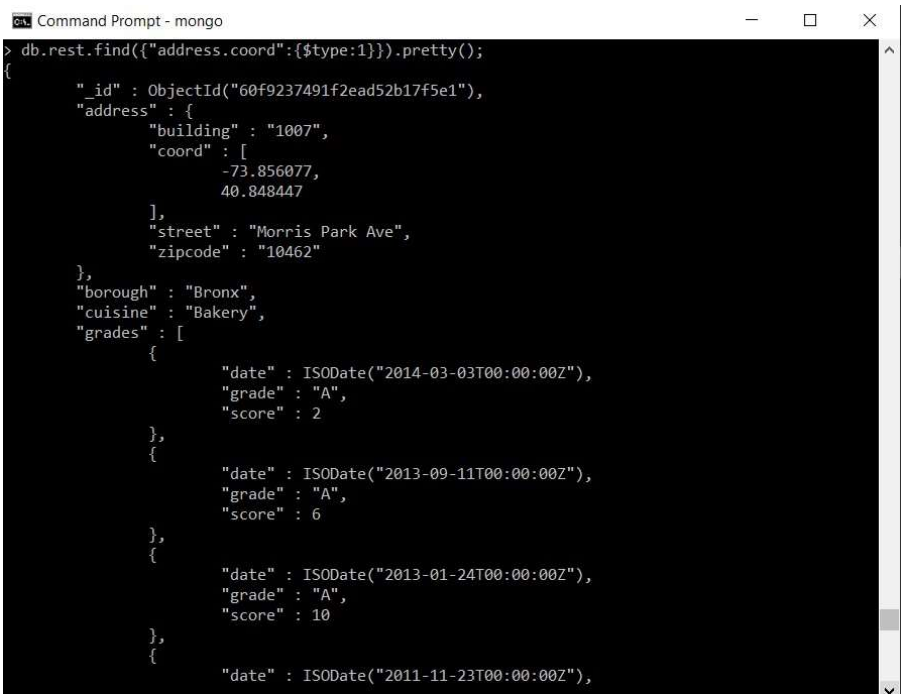
26) Write a MongoDB query to arranged the name of the cuisine in ascending order and borough should be in descending order .



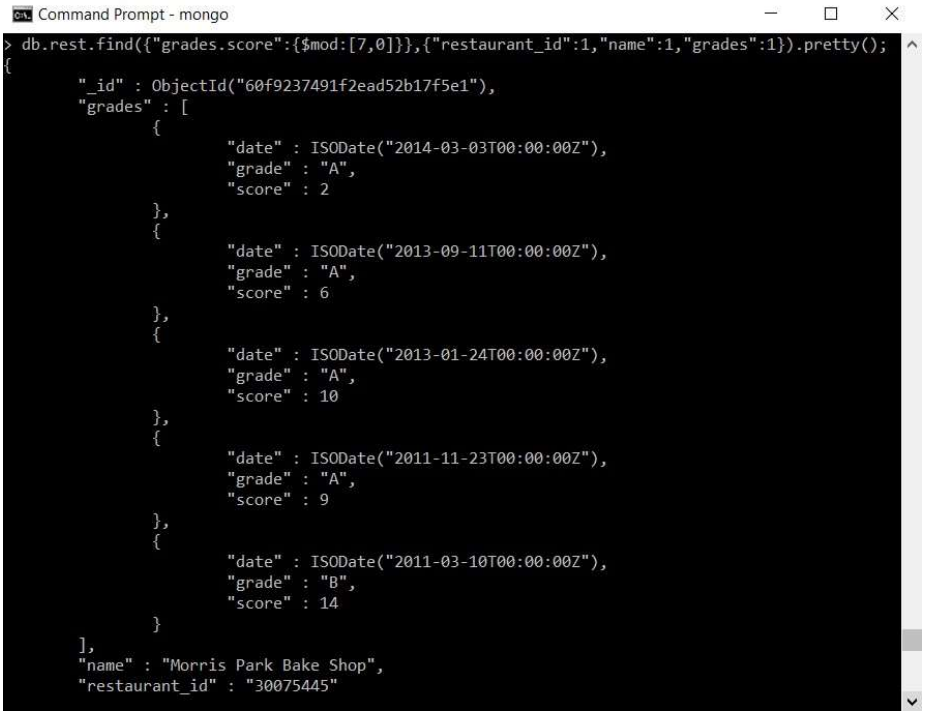
27) Write a MongoDB query to know whether all the addresses contains the street or not.



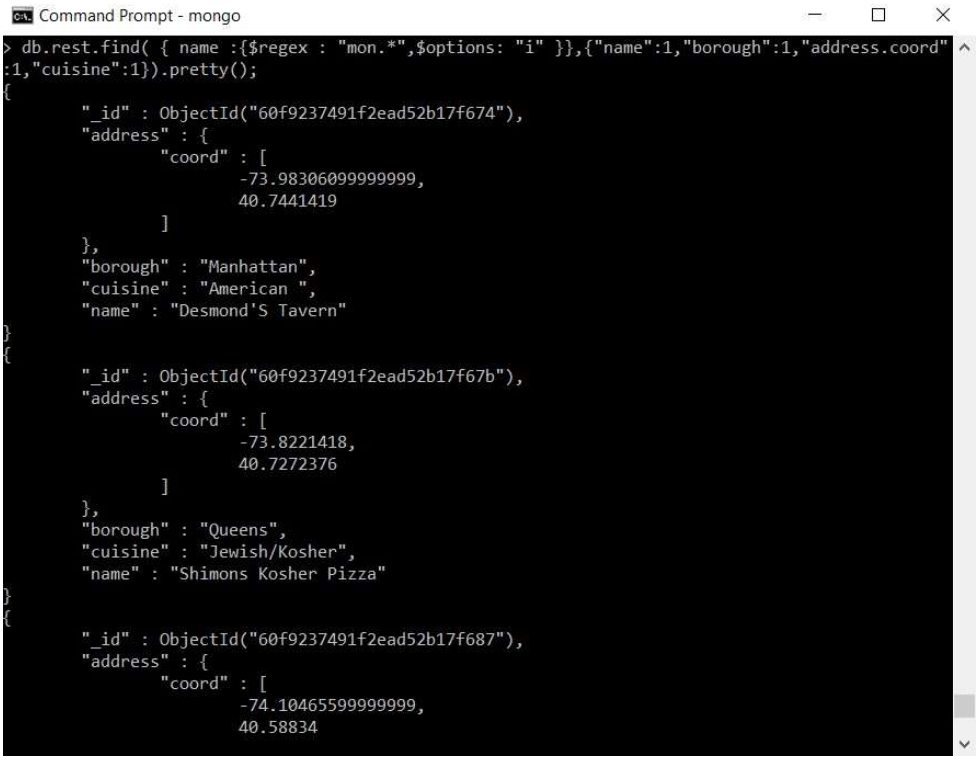
28) Write a MongoDB query which will select all documents in the rest collection where the coord field value is Double



29) Write a MongoDB query which will select the restaurant Id, name and grades for those rest which returns 0 as a remainder after dividing the score by 7.



30) Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those rest which contains 'mon' as three letters somewhere in its name.



31) Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those rest which contain 'Mad' as first three letters of its name

Practical -2B-Basic Queries

a. Consider a Collection users containing the following fields

{
id: ObjectID(),
FName: "First Name",
LName: "Last Name",
Age: 30,
Gender: "M",
Country: "Country"
}

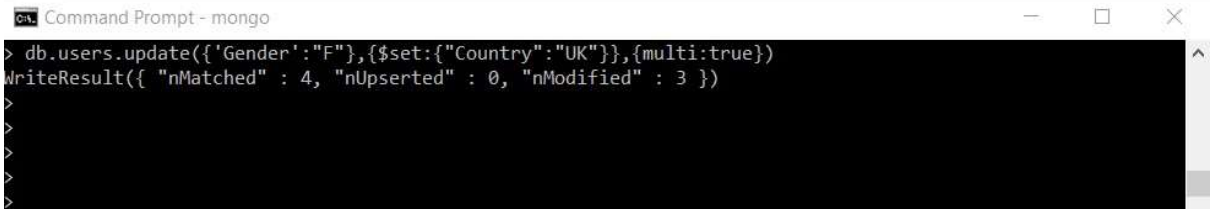
Where Gender value can be either "M" or "F" or "Other".
Country can be either "UK" or "India" or "USA".
Based on above information write the **MongoDB** query for the following.

i. Update the country to UK for all female users.
ii. Add the new field company to all the documents.
iii. Delete all the documents where Gender = 'M'.
iv. Find out a count of female users who stay in either India or USA.
v. Display the first name and age of all female employees.

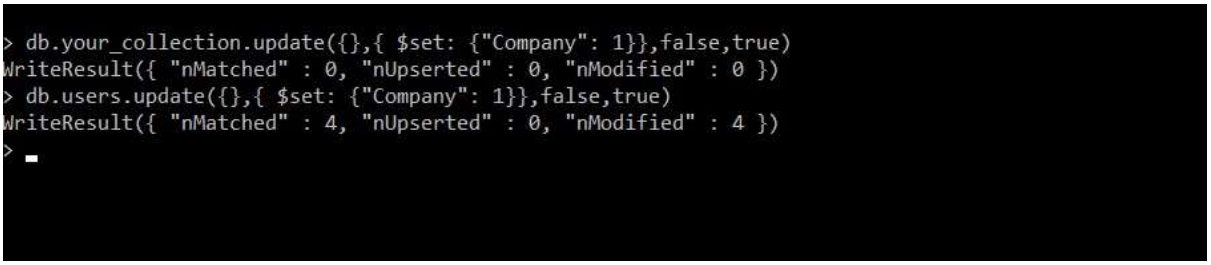
Inserted records

```
> use TYIT
switched to db TYIT
> db.users.insert({"FName":"Aryamaan","LName":"Phadnis","Age":19,"Gender":"M","Country":"India");
... }
2021-07-23T10:29:35.340+0530 E QUERY    [thread1] SyntaxError: missing } after property list @(shell):1:9
> db.users.insert({"FName":"Aryamaan","LName":"Phadnis","Age":19,"Gender":"M","Country":"India"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"John","LName":"Gacy","Age":41,"Gender":"M","Country":"USA"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Emily","LName":"Blunt","Age":38,"Gender":"F","Country":"UK"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Goldy","LName":"Jaiswal","Age":38,"Gender":"F","Country":"India"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Ritik","LName":"Jaiswal","Age":38,"Gender":"M","Country":"India"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Ronald","LName":"Donald","Age":40,"Gender":"M","Country":"India"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Donald","LName":"Trump","Age":52,"Gender":"M","Country":"USA"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Rakesh","LName":"Gill","Age":35,"Gender":"M","Country":"UK"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Lilly","LName":"Singh","Age":30,"Gender":"F","Country":"USA"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Sweety","LName":"Singh","Age":27,"Gender":"F","Country":"India"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Omkar","LName":"Parkar","Age":25,"Gender":"M","Country":"India"});
WriteResult({ "nInserted" : 1 })
> db.users.insert({"FName":"Swaroop","LName":"Dhamankar","Age":29,"Gender":"M","Country":"USA"});
WriteResult({ "nInserted" : 1 })
>
>
```

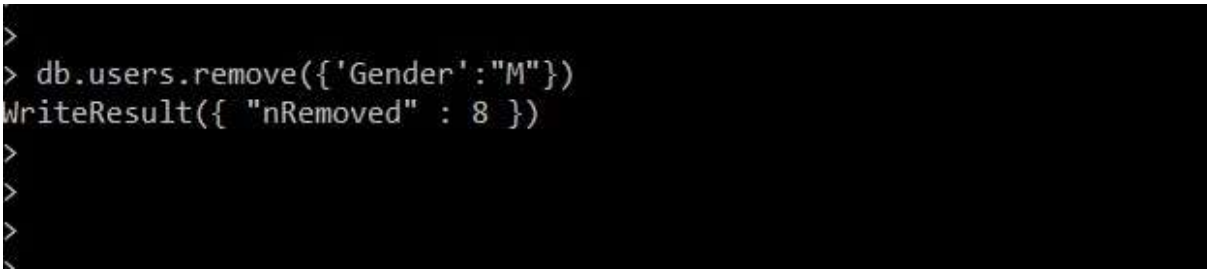
Q.1 Update the country to UK for all female users.



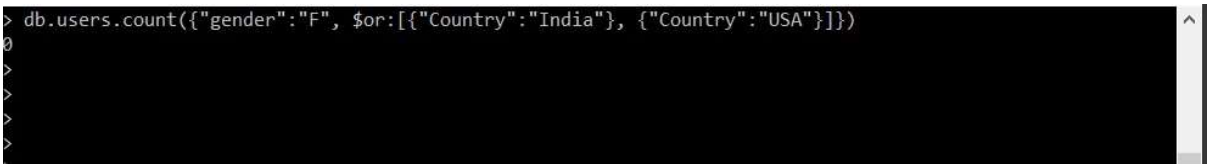
Q.2 Add the new field company to all documents.



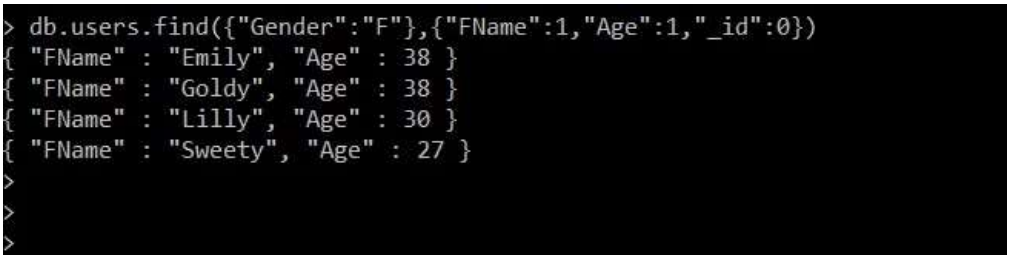
Q.3 Delete all documents with gender= “M”



Q.4 Find out a count of female users who stay in either India or USA



Q.5 Display the first name and age of all female employees.



Practical 3
Aggregate functions

Insert collection in the database and display all the records.

Command Prompt - mongo

```
> db.mycol.find().pretty()
{
        "_id" : ObjectId("60ee64f48d30381dbaf4ff7e"),
        "title" : "MongoDB Overview",
        "description" : "MongoDB is no sql database",
        "by_user" : "tutorials point",
        "url" : "http://www.tutorialspoint.com",
        "tags" : [
                "mongodb",
                "database",
                "NoSQL"
        ],
        "likes" : 100
}
{
        "_id" : ObjectId("60ee651c8d30381dbaf4ff7f"),
        "title" : "NoSQL Overview",
        "description" : "No sql database is very fast",
        "by_user" : "tutorials point",
        "url" : "http://www.tutorialspoint.com",
        "tags" : [
                "mongodb",
                "database",
                "NoSQL"
        ],
        "likes" : 10
}
{
        "_id" : ObjectId("60ee65768d30381dbaf4ff80"),
        "title" : "Neo4j Overview",
```

Q.1 Group by function to get count.

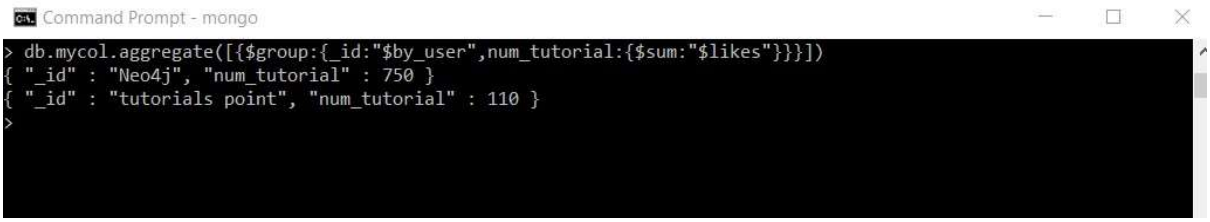
Command Prompt - mongo

```
> db.mycol.aggregate([{$group : {_id : "$by_user", num_tutorial : {$sum : 1}}}])
{ "_id" : "Neo4j", "num_tutorial" : 1 }
{ "_id" : "tutorials point", "num_tutorial" : 2 }
>
```

Q.2 Sum function.

$sum

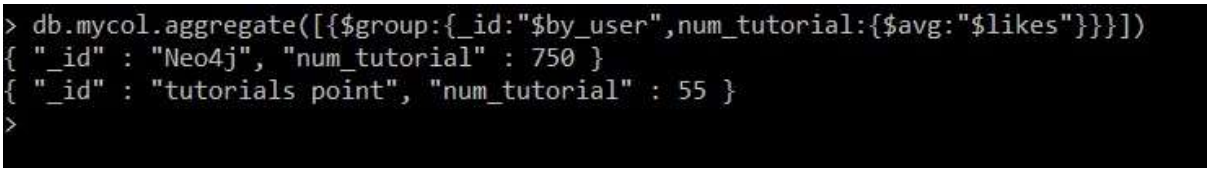
Sums up the defined value from all documents in the collection.



Q.3 Avg function.

$avg

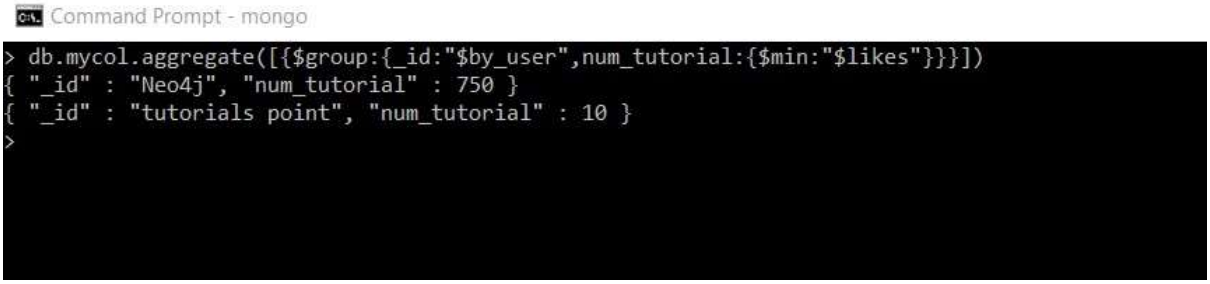
Calculates the average of all given values from all documents in the collection.



Q.4 Min function

$min

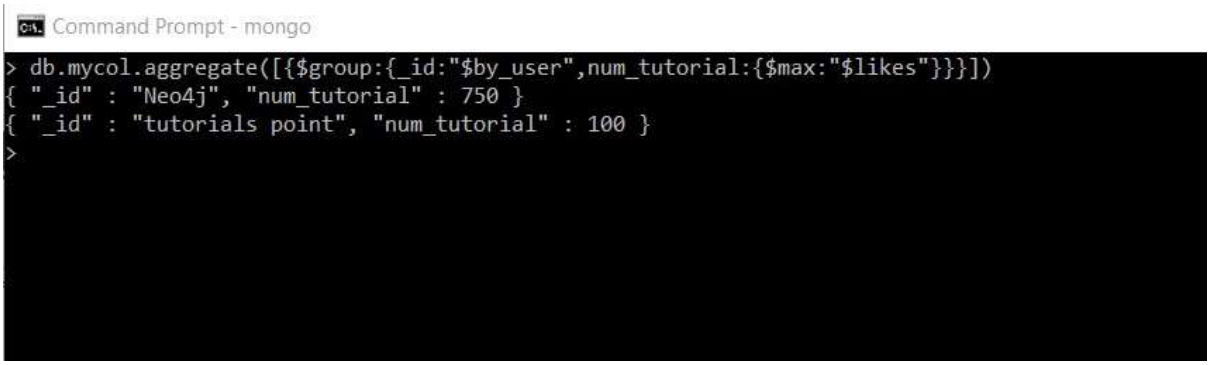
Gets the minimum of the corresponding values from all documents in the collection.



Q.5 Max function.

$max

Gets the maximum of the corresponding values from all documents in the collection.



Q.6 Push function

$push

Inserts the value to an array in the resulting document.

Q.7 addToSet function

$addToSet

Inserts the value to an array in the resulting document but does not create duplicates.

Q.8 First function

$first

Gets the first document from the source documents according to the grouping. Typically this makes only sense together with some previously applied “$sort”-stage.

Q.9 Last function

$last

Gets the last document from the source documents according to the grouping. Typically, this makes only sense together with some previously applied “$sort”-stage.

Practical 4
Replication in MongoDB

Step 1: Set the path MongoDB directory as:

C:\Program Files\MongoDB\Server\3.6\bin

Step 2: Follow the below steps:-

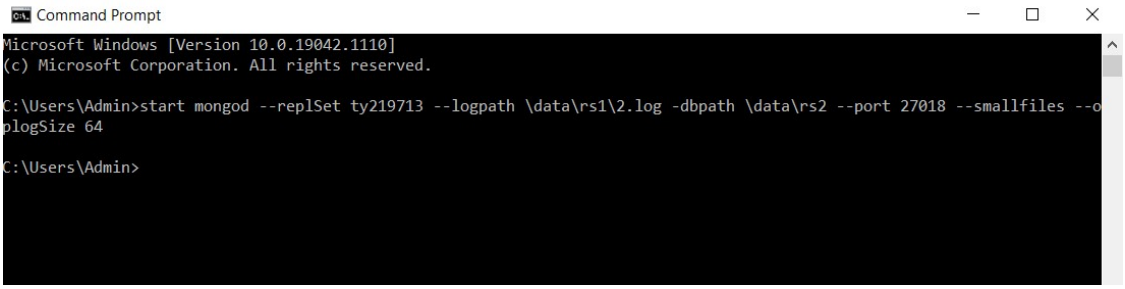
- Go to in "C" directory open the folder name as "data"
- Delete the "db" folder
- Create 3 new folder name as: rs1, rs2, rs3

Step 3:

- Open the CMD in admin mode
- For checking if MongoDB path set or not type command:

  >mongod
- For running server-1(27017) type the below command:

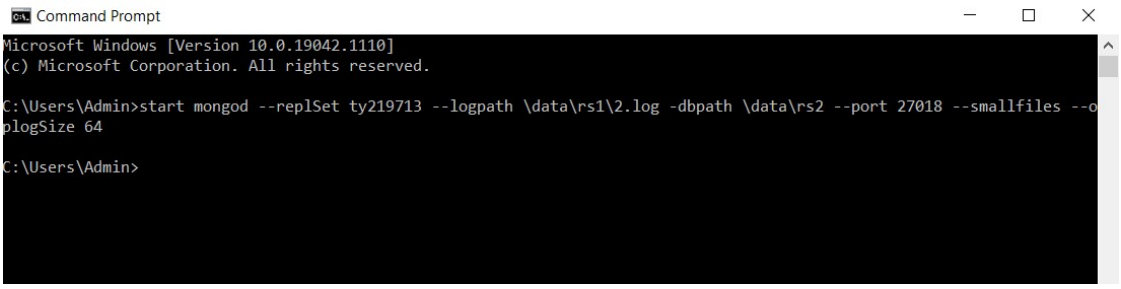
  start mongod --replSet ty219713 --logpath \data\rs1\1.log –dbpath \data\rs1 --port 27017 --smallfiles --oplogSize 64



Step 4:

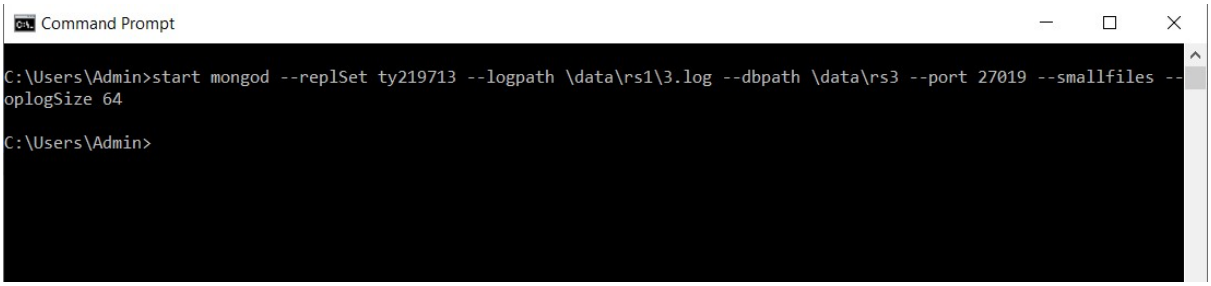
- Again open CMD in admin mode
- For running server-2 (27018) type below command:

  start mongod --replSet ty219713 --logpath \data\rs1\2.log –dbpath \data\rs2 --port 27018 --smallfiles --oplogSize 64



Step 5:

- Again open CMD in admin mode
- For running server-3 (27019) use below command:
- start mongod --replSet ty219713 --logpath \data\rs1\3.log –dbpath \data\rs3 --port 27019 --smallfiles --oplogSize 64



Server 1 port 27017:

Server 2 port 27018:

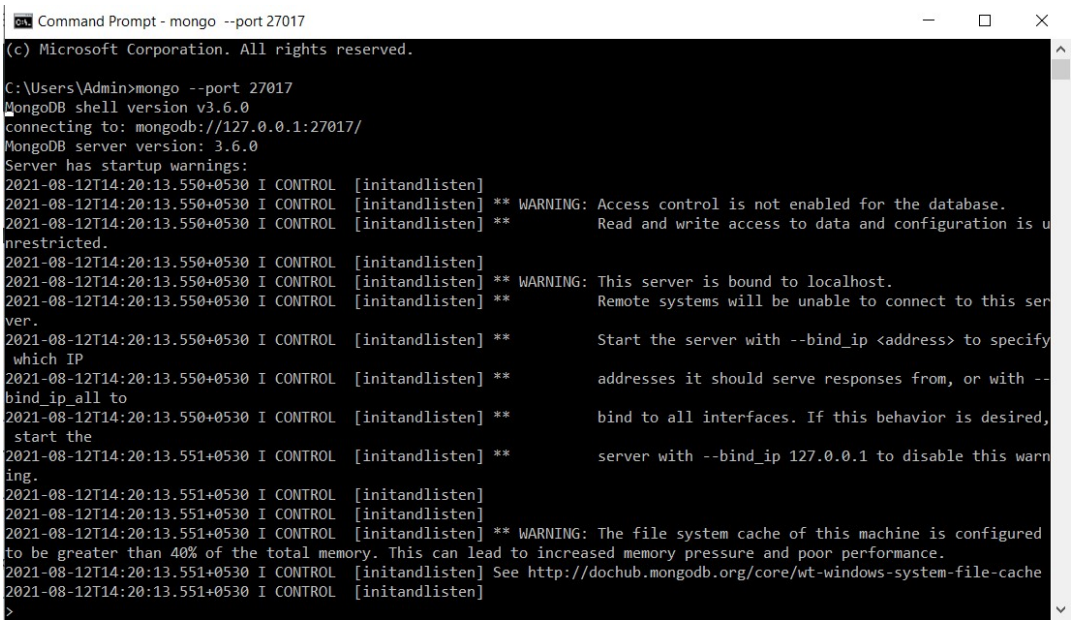
Server 3 port 27019:

Step 6:

• Switch to server-1 (27017) to connect the client to the server with the port 27017 used the following command:

>mongo –port 27017

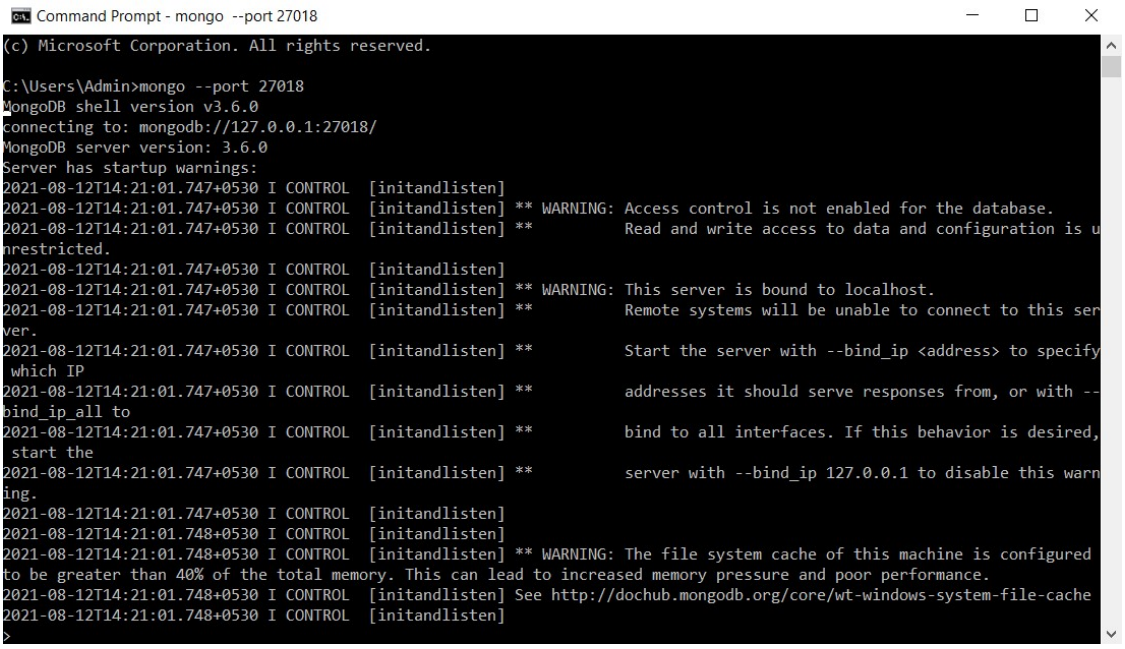
Starting client at Port 27017:



• Switch to server-2 (27018)

>mongo --port 27018

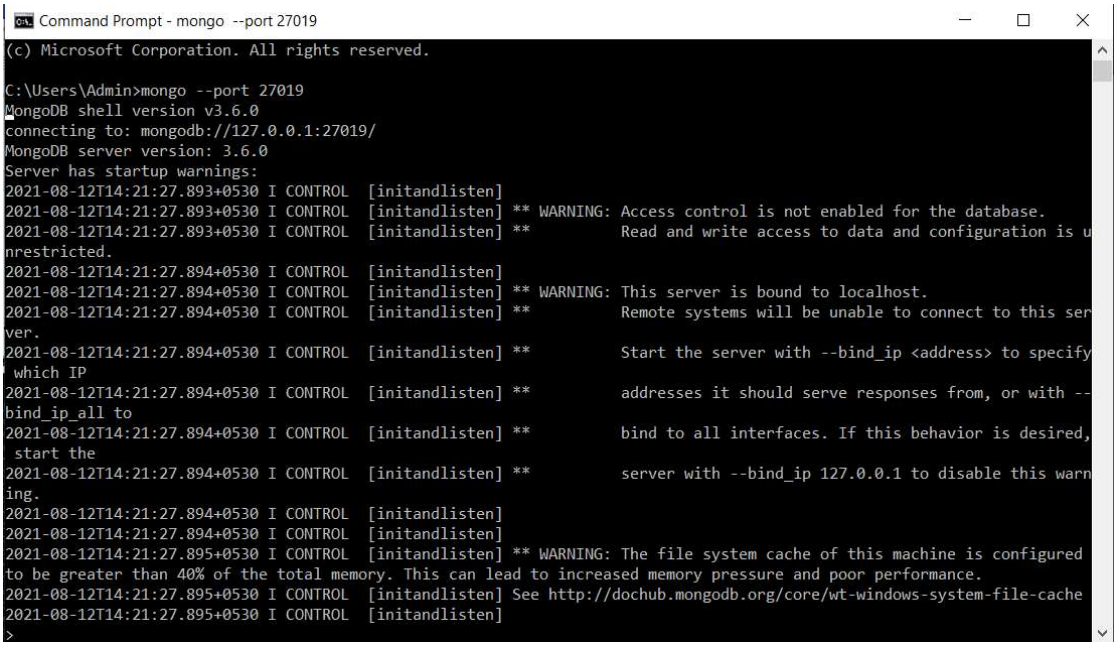
Starting client at Port 27017:



• Switch to server-3 (27019)

>mongo --port 27019

Starting client at Port 27019:



Step 7:

• Open server-1 (27017) and create configuration file which list the replica set name along with the server running on different port

```
config = {
...     _id:"ty219713",
...     members:[
...     {_id:0, host:"localhost:27017"},
...     {_id:1, host:"localhost:27018"},
...     {_id:2, host:"localhost:27019"}
...     ]
... }
```

• For initiating the Config file use the following command:

```
> rs.initiate(config)
```

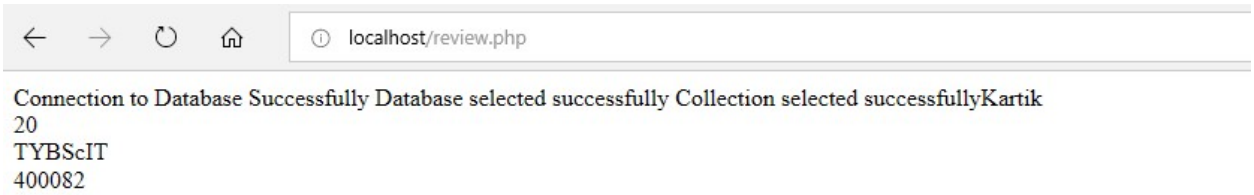
Configuring Primary Port 27017:

Step 8: For executing remaining server (2&3) use the following command in server-2 and server-3

> rs.slaveOk()

Configuring Secondary Port 27018:

Configuring Secondary Port 27019:



Command Prompt - mongo --port 27019

```
ty219713:SECONDARY> rs.slaveOk()
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
ty219713:SECONDARY>
```

Step 9: Go to server-1 to create on document and insert that:

>db.student.insert({"name":"Riddhi","class":"TYIT"})

Inserting Record in Primary Port 27017:

Command Prompt - mongo --port 27017

```
ty219713:PRIMARY> db.student.insert({"name":"Aryamaan","class":"TYIT"})
WriteResult({ "nInserted" : 1 })
ty219713:PRIMARY>
```

Step 10: Now login to any secondary server and issue the command:

> db.student.find().pretty()

Checking Record in secondary Port 27018:

Command Prompt - mongo --port 27018

```
ty219713:SECONDARY> db.student.find().pretty()
{
        "_id" : ObjectId("6115551cd42059e4ea47adba"),
        "name" : "Aryamaan",
        "class" : "TYIT"
}
ty219713:SECONDARY>
```

Checking Record in secondary Port 27019:

Command Prompt - mongo --port 27019

```
ty219713:SECONDARY> db.student.find().pretty()
{
        "_id" : ObjectId("6115551cd42059e4ea47adba"),
        "name" : "Aryamaan",
        "class" : "TYIT"
}
ty219713:SECONDARY>
```

Practical 5
Java And MongoDB

Q1) Insert the document in the MongoDB using Java.

Code:-

```
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;
import org.bson.Document;

class insert
{
	public static void main(String args[])
	{
		MongoClient mongo=new MongoClient("localhost",27017);
		System.out.println("Connected to the database successfully");
		MongoDatabase database=mongo.getDatabase("TYITDB");

		MongoCollection<Document>collection=database.getCollection("TYITCOL");
		System.out.println("Collection sampleCollection selected successfully");
		Document document=new Document();

		document.append("id",229726);
		document.append("name","Kartik Sharma");
		document.append("age",19);
		document.append("class","TYBscIT");
		document.append("college","Mulund cillege of Commerce");
		collection.insertOne(document);
		System.out.println("Document inserted successfully");
	}
}
```

Output:-

```
D:\NGT_Practical\javaMOD>java insert
Sep 15, 2022 9:56:16 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTimeout='30000 ms', maxWaitQueueSize=50
0}
Connected to the database successfully
Collection sampleCollection selected successfully
Sep 15, 2022 9:56:16 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by PrimaryServerSelector from cluster description ClusterDescription{type=UNKNOWN, connectionMode=SINGLE, all=[ServerDescription{addres
s=localhost:27017, type=UNKNOWN, state=CONNECTING}]}. Waiting for 30000 ms before timing out
Sep 15, 2022 9:56:16 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:204}] to localhost:27017
Sep 15, 2022 9:56:16 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=STANDALONE, state=CONNECTED, ok=true, v
ersion=ServerVersion{versionList=[5, 0, 9]}, minWireVersion=0, maxWireVersion=13, electionId=null, maxDocumentSize=16777216, roundTripTimeNanos=1579800}
Sep 15, 2022 9:56:16 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:205}] to localhost:27017
Document inserted successfully
```

```
> db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("632352289e51d473d0f430c3"),
        "id" : 229726,
        "name" : "Kartik Sharma",
        "age" : 19,
        "class" : "TYBScIT",
        "college" : "Mulund cillege of Commerce"
}
{
        "_id" : ObjectId("6325793418280e41a0b1331d"),
        "id" : 229727,
        "name" : "Tom",
        "age" : 34,
        "class" : "TYBscCS",
        "college" : "Mulund cillege of Commerce"
}
{
        "_id" : ObjectId("6325799518280e0560b5868f"),
        "id" : 229727,
        "name" : "Jerry",
        "age" : 23,
        "class" : "TYBSC",
        "college" : "Mulund cillege of Commerce"
}
```

Q2) Update the document in the MongoDB using Java.

Code:-

update - Notepad

File Edit View

```
import com.mongodb.client.MongoCollection;
import com.mongodb.BasicDBObject;
import com.mongodb.DB;
import com.mongodb.DBCollection;
import com.mongodb.MongoClient;
import com.mongodb.DBObject;
import com.mongodb.WriteResult;
import java.net.UnknownHostException;

class update
{
      public static void main(String args[])
      {
            MongoClient mongo=new MongoClient("localhost",27017);
            System.out.println("Connected to the database successfully");
            DB db=mongo.getDB("TYITDB");

            DBCollection col=db.getCollection("TYITCOL");
            DBObject query = new BasicDBObject("id",229726);
            DBObject update = new BasicDBObject();

            update.put("$set",new BasicDBObject("name","Shivay"));
            WriteResult result = col.update(query,update);
            mongo.close();
      }
}
```

Output:-

```
D:\NGT_Practical\javaMOD>javac update.java
Note: update.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\NGT_Practical\javaMOD>java update
Sep 17, 2022 1:19:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionT
imeout='30000 ms', maxWaitQueueSize=500}
Connected to the database successfully
Sep 17, 2022 1:19:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by PrimaryServerSelector from cluster description ClusterDescription{type=UNKNOWN, connectionMode
=SINGLE, all=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]}. Waiting for 30000 ms before
timing out
Sep 17, 2022 1:19:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:25}] to localhost:27017
Sep 17, 2022 1:19:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=S
TANDALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[5, 0, 9]}, minWireVersion=0, maxWireVersion=13,
electionId=null, maxDocumentSize=16777216, roundTripTimeNanos=1508700}
Sep 17, 2022 1:19:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:26}] to localhost:27017
Sep 17, 2022 1:19:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Closed connection [connectionId{localValue:2, serverValue:26}] to localhost:27017 because the pool has been closed
.
> db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("632352289e51d473d0f430c3"),
        "id" : 229726,
        "name" : "Shivay",
        "age" : 19,
        "class" : "TYBScIT",
        "college" : "Mulund cillege of Commerce"
}
{
        "_id" : ObjectId("6325793418280e41a0b1331d"),
        "id" : 229727,
        "name" : "Tom",
        "age" : 34,
        "class" : "TYBScIT",
        "college" : "Mulund cillege of Commerce"
}
{
        "_id" : ObjectId("6325799518280e0560b5868f"),
        "id" : 229727,
        "name" : "Jerry",
        "age" : 23,
        "class" : "TYBSC",
        "college" : "Mulund cillege of Commerce"
}
```

Q3) Delete the document in the MongoDB using Java.

Code:-

```
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;
import org.bson.Document;
import com.mongodb.client.model.Filters;

class delete
{
	public static void main(String args[])
	{
		MongoClient mongo=new MongoClient("localhost",27017);
		System.out.println("Connected to the database successfully");
		MongoDatabase database=mongo.getDatabase("TYITDB");

		MongoCollection <Document> collection=database.getCollection("TYITCOL");
		System.out.println("Collection sampleCollection selected successfully");

		collection.deleteOne(Filters.eq("id",229727));
		System.out.println("Document deleted successfully");
	}
}
```

Output:-

```
D:\NGT_Practical\javaMOD>javac delete.java

D:\NGT_Practical\javaMOD>java delete
Sep 28, 2022 8:58:49 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionT
imeout='30000 ms', maxWaitQueueSize=500}
Connected to the database successfully
Collection sampleCollection selected successfully
Sep 28, 2022 8:58:49 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by PrimaryServerSelector from cluster description ClusterDescription{type=UNKNOWN, connectionMode
=SINGLE, all=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]}. Waiting for 30000 ms before
timing out
Sep 28, 2022 8:58:49 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:1}] to localhost:27017
Sep 28, 2022 8:58:49 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=S
TANDALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[5, 0, 9]}, minWireVersion=0, maxWireVersion=13,
electionId=null, maxDocumentSize=16777216, roundTripTimeNanos=1060800}
Sep 28, 2022 8:58:49 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:2}] to localhost:27017
Document deleted successfully

D:\NGT_Practical\javaMOD>
```

```
> db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("632352289e51d473d0f430c3"),
        "id" : 229726,
        "name" : "Shivay",
        "age" : 19,
        "class" : "TYBScIT",
        "college" : "Mulund cillege of Commerce"
}
```

Q4) Retrieve the document in the MongoDB using Java.

Code:-

```
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;
import org.bson.Document;
import com.mongodb.client.FindIterable;
import java.util.Iterator;

class retrieve
{
	public static void main(String args[])
	{
		MongoClient mongo=new MongoClient("localhost",27017);
		System.out.println("Connected to the database successfully");
		MongoDatabase database=mongo.getDatabase("TYITDB");

		MongoCollection<Document>collection=database.getCollection("TYITCOL");
		FindIterable <Document> iter=collection.find();
		int i=1;
		Iterator it=iter.iterator();
		while(it.hasNext())
		{
			System.out.println(it.next());
			i++;
		}
	}
}
```

Output:-

```
D:\NGT_Practical\javaMOD>javac retrieve.java

D:\NGT_Practical\javaMOD>java retrieve
Sep 17, 2022 1:29:21 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionT
imeout='30000 ms', maxWaitQueueSize=500}
Connected to the database successfully
Sep 17, 2022 1:29:21 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by ReadPreferenceServerSelector{readPreference=primary} from cluster description ClusterDescripti
on{type=UNKNOWN, connectionMode=SINGLE, all=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]
}. Waiting for 30000 ms before timing out
Sep 17, 2022 1:29:21 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:31}] to localhost:27017
Sep 17, 2022 1:29:21 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=S
TANDALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[5, 0, 9]}, minWireVersion=0, maxWireVersion=13,
electionId=null, maxDocumentSize=16777216, roundTripTimeNanos=3480701}
Sep 17, 2022 1:29:21 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:32}] to localhost:27017
Document{{_id=632352289e51d473d0f430c3, id=229726, name=Shivay, age=19, class=TYBScIT, college=Mulund cillege of Commerc
e}}
```

Practical 6
PHP and MongoDB

1) Insert the document of MongoDB using PHP.

Code:-

```php
<?php
$m=new MongoClient();
echo "\nConnection to Database Successfully";
$db=$m->KARTIKDB;
echo "\nDatabase selected successfully";
$col=$db->KARTIKCOL;
echo "\nCollection selected successfully";

$doc=array(
"name"=>"Kartik",
"age"=>20,
"dept"=>"TYBScIT",
"pin"=>400082
);
$col->insert($doc);
echo "\nDocument inserted successfully";
?>
```

Output:-

Connection to Database Successfully Database selected successfully Collection selected successfully Document inserted successfully

```
MongoDB Enterprise > db.KARTIKCOL.find().pretty()
{
        "_id" : ObjectId("634d43c7d7271a8813000029"),
        "name" : "Shivay",
        "age" : 20,
        "dept" : "TYIT",
        "pin" : 229601
}
{
        "_id" : ObjectId("634d4497d7271a881300002a"),
        "name" : "Kartik",
        "age" : 20,
        "dept" : "TYBScIT",
        "pin" : 400082
}
MongoDB Enterprise >
```

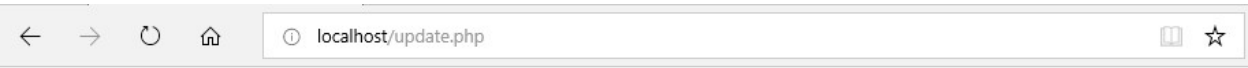
2) Update the document of MongoDB using PHP.

Code:-

```php
<?php
$m=new MongoClient();
echo "\nConnection to Database Successfully";
$db=$m->KARTIKDB;
echo "\nDatabase selected successfully";
$col=$db->KARTIKCOL;
echo "\nCollection selected successfully";

$col->update(array("name"=>"Shivay"),array('$set'=>array("name"=>"Shalini")));
echo "\n document update successfully";
?>
```

Output:-



Connection to Database Successfully Database selected successfully Collection selected successfully document update successfully

```
MongoDB Enterprise > db.KARTIKCOL.find().pretty()
{
        "_id" : ObjectId("634d43c7d7271a8813000029"),
        "name" : "Shalini",
        "age" : 20,
        "dept" : "TYIT",
        "pin" : 229601
}
{
        "_id" : ObjectId("634d4497d7271a881300002a"),
        "name" : "Kartik",
        "age" : 20,
        "dept" : "TYBScIT",
        "pin" : 400082
}
MongoDB Enterprise >
```

3) Delete the document of MongoDB using PHP.

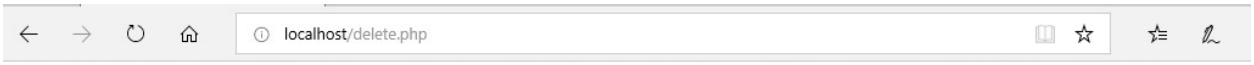
Code:-

```php
<?php
$m=new MongoClient();
echo "\nConnection to Database Successfully";
$db=$m->KARTIKDB;
echo "\nDatabase selected successfully";
$col=$db->KARTIKCOL;
echo "\nCollection selected successfully";

$col->remove(array("name"=>"Shalini"));
echo "\n Document deleted successfully";

$cursor=$col->find();
echo "update document";
foreach($cursor as $document){
    echo $document["Shalini"] . "\n";
    }
?>
```

Output:-



Connection to Database Successfully Database selected successfully Collection selected successfully Document deleted successfullyupdate document
Notice: Undefined index: Shalini in **C:\xampp\htdocs\delete.php** on line **15**

```
MongoDB Enterprise > db.KARTIKCOL.find().pretty()
{
        "_id" : ObjectId("634d4497d7271a881300002a"),
        "name" : "Kartik",
        "age" : 20,
        "dept" : "TYBScIT",
        "pin" : 400082
}
MongoDB Enterprise >
```

4) Review the document of MongoDB using PHP.

Code:-

```php
<?php
$m=new MongoClient();
echo "\nConnection to Database Successfully";
$db=$m->KARTIKDB;
echo "\nDatabase selected successfully";
$col=$db->KARTIKCOL;
echo "\nCollection selected successfully";

$cursor=$col->find();
foreach($cursor as $doc)
{
echo $doc["name"]."<br/>";
echo $doc["age"]."<br/>";
echo $doc["dept"]."<br/>";
echo $doc["pin"]."<br/>";
}
?>
```

Output:-

Connection to Database Successfully Database selected successfully Collection selected successfullyKartik
20
TYBScIT
400082

Practical 7
Python and MongoDB

Q1) Insert the document in the MongoDB using Python.

Code:-

229726_insertMOD.py - D:/NGT_Practical/229726_insertMOD.py (3.9.0)

File Edit Format Run Options Window Help

```
from pymongo import MongoClient
client=MongoClient('localhost',27017)
db=client.TYITDB
def insert():
    try:
        name1=input("Enter the name:")
        Age1=input("Enter the age:")
        db.TYITCOL.insert_one({
            "name":name1,
            "age":Age1
        })
        print("Data inserted successfully")
    except Exception:
        print(str(e))
insert()
```

Output:-

```
================= RESTART: D:/NGT_Practical/229726_insertMOD.py ================
Enter the name:Kartik
Enter the age:20
Data inserted successfully
>>>
================= RESTART: D:/NGT_Practical/229726_insertMOD.py ================
Enter the name:Tom
Enter the age:21
Data inserted successfully
>>>
================= RESTART: D:/NGT_Practical/229726_insertMOD.py ================
Enter the name:Jerry
Enter the age:22
Data inserted successfully
```

```
> db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("63208a5ba295d4c2d37e5626"),
        "name" : "Kartik",
        "age" : "20"
}
{
        "_id" : ObjectId("63208a66860793d45955d36f"),
        "name" : "Tom",
        "age" : "21"
}
{
        "_id" : ObjectId("63208a70694b9f16718b5323"),
        "name" : "Jerry",
        "age" : "22"
}
```

Q2) Update the document in the MongoDB using Python.

Code:-

229726_updateMOD.py - D:/NGT_Practical/229726_updateMOD.py (3.9.0)

File Edit Format Run Options Window Help

```
from pymongo import MongoClient
client=MongoClient('localhost',27017)
db=client.TYITDB
def update():
    try:
        name1=input("Enter the name:")
        Age1=int(input("Enter the age:"))
        db.TYITCOL.update_one(
            {"name":name1},{"$set":{"age":Age1}}
        )
        print("\nRecord is updated successfully\n")
    except Exception:
        print(str(e))
update()
```

Output:-

```
==================== RESTART: D:\NGT_Practical\updateMOD.py ====================
Enter the name:Kartik
Enter the age:45

Record is updated successfully

>>>
==================== RESTART: D:\NGT_Practical\updateMOD.py ====================
Enter the name:Tom
Enter the age:45

Record is updated successfully
```

```
> db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("63208a5ba295d4c2d37e5626"),
        "name" : "Kartik",
        "age" : 45
}
{ "_id" : ObjectId("63208a66860793d45955d36f"), "name" : "Tom", "age" : 45 }
{
        "_id" : ObjectId("63208a70694b9f16718b5323"),
        "name" : "Jerry",
        "age" : "22"
}
```

Q3) Delete the document in the MongoDB using Python.

Code:-

229726_delateMOD.py - D:/NGT_Practical/229726_delateMOD.py (3.9.0)

File Edit Format Run Options Window Help

```
from pymongo import MongoClient
client=MongoClient('localhost',27017)
db=client.TYITDB
def delete():
    try:
        name1=input("Enter the name:")
        db.TYITCOL.delete_one(
            {"name":name1})
        print("\nData removed successfully\n")
    except Exception:
        print(str(e))
delete()
```

Output:-

```
================= RESTART: D:/NGT_Practical/229726_delateMOD.py ================
Enter the name:Tom

Data removed successfully

>>>
================= RESTART: D:/NGT_Practical/229726_delateMOD.py ================
Enter the name:Jerry

Data removed successfully
```

```
> db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("63208a5ba295d4c2d37e5626"),
        "name" : "Kartik",
        "age" : 45
}
```

Q4) Fetching the document in the MongoDB using Python.

Code:-

229726_fetchMOD.py - D:\NGT_Practical\229726_fetchMOD.py (3.9.0)

File Edit Format Run Options Window Help

```
from pymongo import MongoClient
client=MongoClient('localhost',27017)
db=client.TYITDB
def read():
    try:
        res=db.TYITCOL.find()
        print("\nData updated successfully\n")
        for one in res:
            print(one)
    except Exception as e:
        print(str(e))
read()
```

Output:-

```
================== RESTART: D:\NGT_Practical\229726_fetchMOD.py ==============

Data updated successfully

{'_id': ObjectId('63208a5ba295d4c2d37e5626'), 'name': 'Kartik', 'age': 45}
>>>
```

Practical 8
Program On Basic jQuery

Q1) Write a jQuery to check the content of the element on button click.

Code:-

229726_content.html

D: > NGT_Practical > 229726_content.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    document.write("Hello World!")
                })
            })
        </script>
    </head>
    <body>
        <p>Well Come to beautiful world</p>
        <button>Click Me</button>
    </body>
</html>
```

Output:-

Well Come to beautiful world

Click Me

Hello World!

Q2) Write a jQuery to select element by class name, id and element name.

Code:-

229726_content.html

D: > NGT_Practical > 229726_id.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("class1").css("background","yellow")
                $("#id1").css("background","red")
                $("h2").css("background","green")

            })
        </script>
    </head>
    <body>
        <p class="class1">Hello!</p>
        <p id="id1">Student</p>
        <h2>Well Come</h2>
    </body>
</html>
```

Output:-

Q3) Write a jQuery to show the use of click(), hover(), on(), trigger(), off() event.

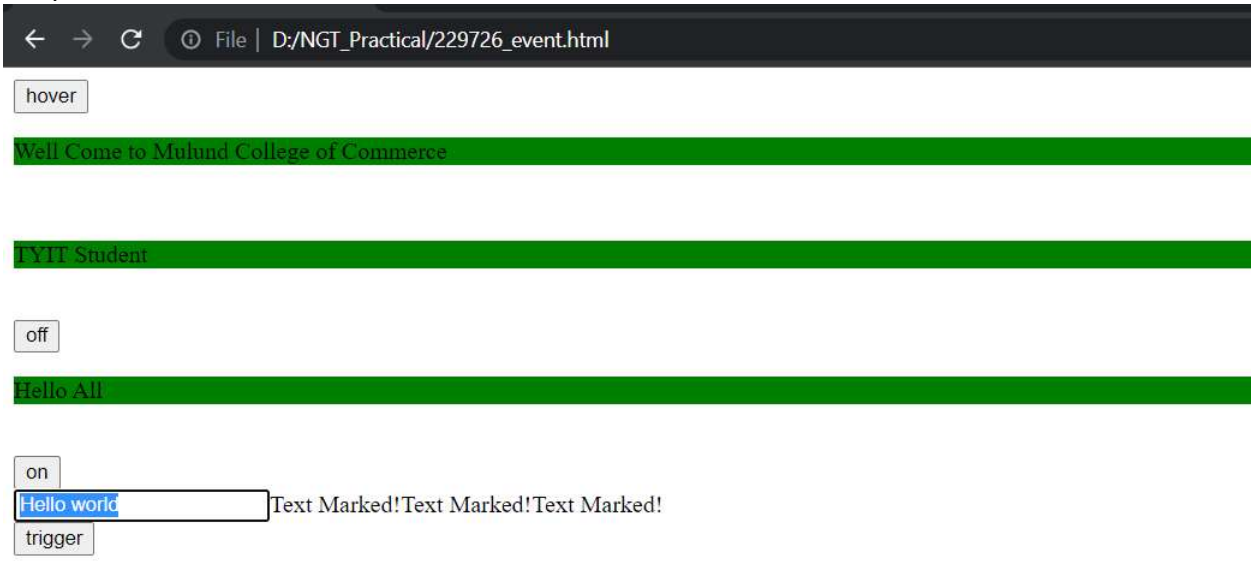
Code:-

229726_event.html

D: > NGT_Practical > 229726_event.html > html > body

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("#b1").hover(function(){
                    document.write("Hello World!")
                });
                $("p").on("click",function(){
                    $(this).css("background","green")
                });
                $("#b2").click(function(){
                    $("p").off("click")
                });
                $("#b3").click(function(){
                    $("#t1").hide()
                });
                $("input").select(function(){
                    $("input").after("Text Marked!")
                });
                $("#b4").click(function(){
                    $("input").trigger("select")
                })
            })
        </script>
    </head>
    <body>
        <button id="b1">hover</button><br>
        <p>Well Come to Mulund College of Commerce</p><br>
        <p>TYIT Student</p><br>
        <button id="b2">off</button><br>
        <p id="t1">Hello All</p><br>
        <button id="b3">on</button><br>
        <input type="text" value="Hello world"><br>
        <button id="b4">trigger</button>
    </body>
</html>
```

Output:-



Q4) Write a jQuery to create animated hide effect.

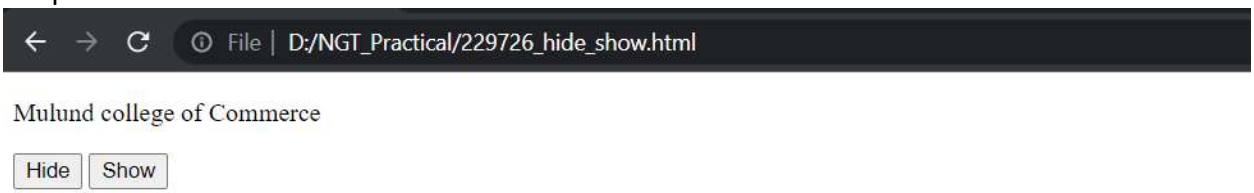
Code:-

229726_event.html | 229726_content.html

D: > NGT_Practical > 229726_hide_show.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("#b1").click(function(){
                    $("p").hide();
                });
                $("#b2").click(function(){
                    $("p").show();
                })
            })
        </script>
    </head>
    <body>
        <p>Mulund college of Commerce</p>
        <button id="b1">Hide</button>
        <button id="b2">Show</button>
    </body>
</html>
```

Output:-



Q5) Write a jQuery to create a simple toggle effects.

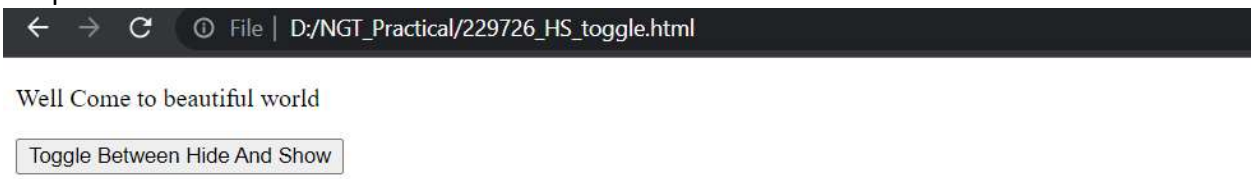
Code:-

229726_content.html

D: > NGT_Practical > 229726_HS_toggle.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $("p").toggle()
                })
            })
        </script>
    </head>
    <body>
        <p>Well Come to beautiful world</p>
        <button>Toggle Between Hide And Show</button>
    </body>
</html>
```

Output:-



Q6) Write a jQuery to create fade-in and fade-out effects.

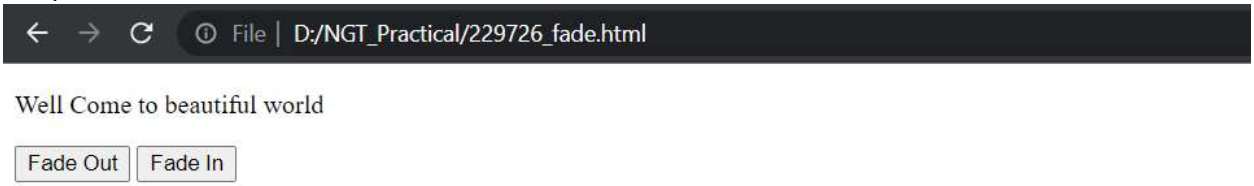
Code:-

229726_content.html | 229726_fade.html

D: > NGT_Practical > 229726_fade.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $(".class1").click(function(){
                    $("p").fadeOut(3000)
                });
                $(".class2").click(function(){
                    $("p").fadeIn(5000)
                })
            })
        </script>
    </head>
    <body>
        <p>Well Come to beautiful world</p>
        <button class="class1">Fade Out</button>
        <button class="class2">Fade In</button>
    </body>
</html>
```

Output:-



Q7) Write a jQuery to create fade toggle effects.

Code:-

229726_content.html

D: > NGT_Practical > 229726_fade_toggle.html > html

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <title>Content Of Element</title>
5           <script src="jquery-3.6.1.min.js"></script>
6           <script>
7               $(document).ready(function(){
8                   $("button").click(function(){
9                       $("p").fadeToggle()
10                  })
11              })
12          </script>
13      </head>
14      <body>
15          <p>Well Come to beautiful world</p>
16          <button>Fade Toggle</button>
17      </body>
18  </html>
```

Output:-

Q8) Write a jQuery to create a slide-up and slide-down effects.

Code:-

229726_content.html | 229726_fade.html | 229726_slide.html

D: > NGT_Practical > 229726_slide.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("#b1").click(function(){
                    $("p").slideUp(5000)
                });
                $("#b2").click(function(){
                    $("p").slideDown(3000)
                })
            })
        </script>
    </head>
    <body>
        <p>Well Come to beautiful world</p>
        <button id="b1">Slide Up</button>
        <button id="b2">Slide Down</button>
    </body>
</html>
```

Output:-

Q9) Write a jQuery to create slide toggle effects.

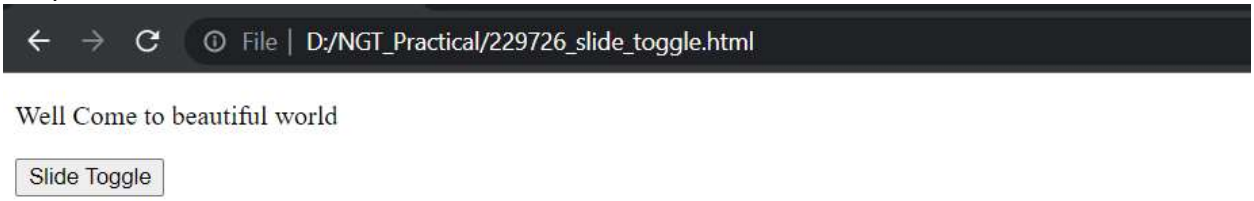
Code:-

229726_fade_toggle.html | 229726_slide_toggle.html

D: > NGT_Practical > 229726_slide_toggle.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Content Of Element</title>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $("p").slideToggle(3000)
                })
            })
        </script>
    </head>
    <body>
        <p>Well Come to beautiful world</p>
        <button>Slide Toggle</button>
    </body>
</html>
```

Output:-


File | D:/NGT_Practical/229726_slide_toggle.html
Well Come to beautiful world
Slide Toggle

Practical 9
jQuery Animation

Q1) Write a jQuery to create Animation effects.

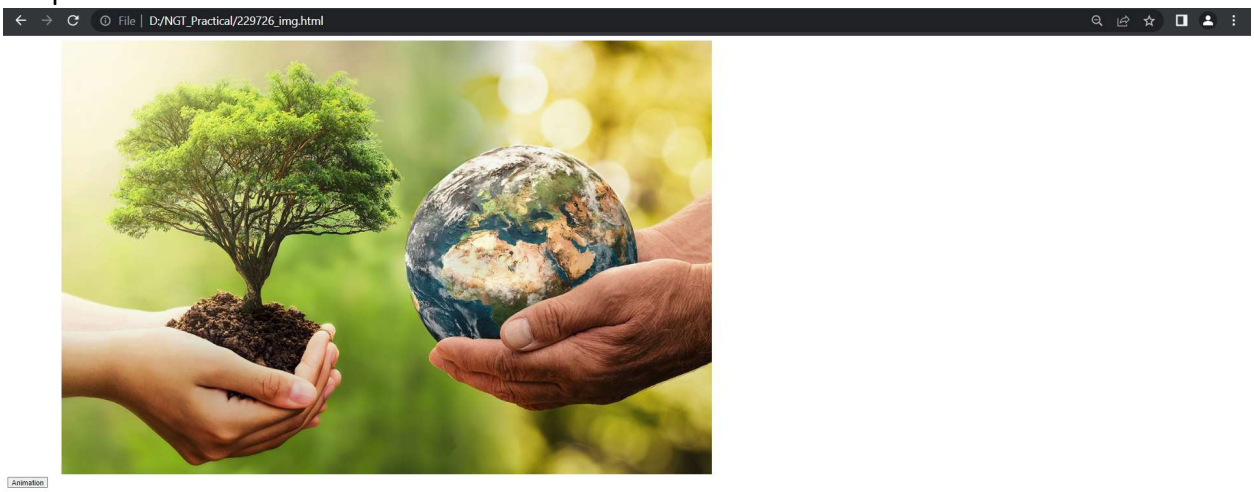
Code:-

229726_fade_toggle.html | 229726_img.html

D: > NGT_Practical > 229726_img.html > html

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script src="jquery-3.6.1.min.js"></script>
5          <style>
6              img{
7              position:relative;
8          }
9          </style>
10         <script>
11             $(document).ready(function(){
12                 $("button").click(function(){
13                     $("img").animate({left:100});
14                 })
15             })
16         </script>
17     </head>
18     <body>
19         <img src="C:\Users\Lenovo\Desktop\Earth.jpg" alt="Earth">
20         <br><button>Animation</button>
21     </body>
22 </html>
```

Output:-



Q2) Write a jQuery animate multiple CSS properties.

Code:-

D: > NGT_Practical > 229726_milti_css.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <script src="jquery-3.6.1.min.js"></script>
        <style type="text/css">
            .box{
                width:100px;;
                height:100px;
                background:black;
                margin-top:30px;
                border-style:solid;
                border-color:green;
                border-width:14px;
            }
        </style>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $(".box").animate({
                        width:"300px"
                        height:"300px"
                        margin-left:"150px"
                        border-width:"10px"
                        opacity:0.5
                    })
                })
            })
        </script>
    </head>
    <body>
        <button>Animation</button>
        <div class="box"></div>
    </body>
</html>
```

Output:-

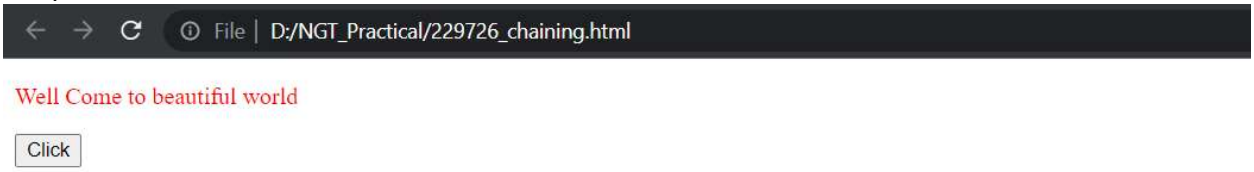
Q3) Write a jQuery to perform method of chaining.

Code:-

D: > NGT_Practical > 229726_chaining.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $("#p1").css("color","red").slideUp(2000).slideDown(2000)
                })
            })
        </script>
    </head>
    <body>
        <p id="p1">Well Come to beautiful world</p>
        <button>Click</button>
    </body>
</html>
```

Output:-



Q4) Write a jQuery effects method with callback function.

Code:-

D: > NGT_Practical > 229726_callback.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $("p").hide("slow",function(){
                        alert("The paragraph is now hide")
                    })
                })
            })
        </script>
    </head>
    <body>
        <p>This is a simple Paragraph.</p>
        <button>Click</button>
    </body>
</html>
```

Output:-

Q5) Write a jQuery to get and set text contents of the elements.

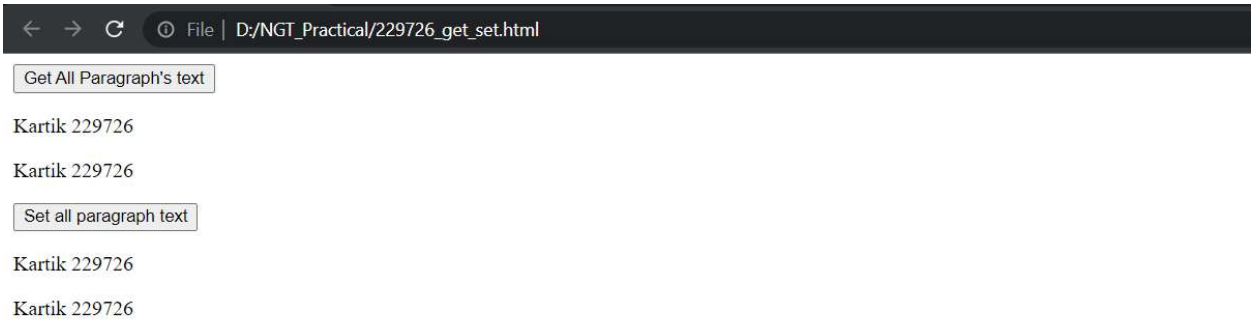
Code:-

D: > NGT_Practical > 229726_get_set.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $(".b1").click(function(){
                    var str=$("p").text();
                    alert(str);

                });
                $(".b2").click(function(){
                    $("p").text("Kartik 229726")
                })
            })
        </script>
    </head>
    <body>
        <button class="b1">Get All Paragraph's text</button>
        <p>This is a paragraph.</p>
        <p>This is an another paragraph.</p>
        <button class="b2">Set all paragraph text</button>
        <p>This is a test paragraph.</p>
        <p>This is an another test paragraph.</p>
    </body>
</html>
```

Output:-



Q6) Write a jQuery to get and set HTML content of the elements.

Code:-

D: > NGT_Practical > 229726_get_set_HTML.html > html

```
<!DOCTYPE html>
<html>
    <head>
        <script src="jquery-3.6.1.min.js"></script>
        <script>
            $(document).ready(function(){
                $(".b1").click(function(){
                    var str=$("p").html()
                    alert(str)

                });
                $(".b2").click(function(){
                    $("body").html("<p> <b>Kartik 229726</b> </p>")
                })
            })
        </script>
    </head>
    <body>
        <button class="b1">Get paragraph HTML content</button>
        <p>The quick <b>brown for</b> jumps over the lazy dog.</p>
        <button class="b2">Write Message</button>
    </body>
</html>
```

Output:-

Kartik 229726

Practical 10
JSON

Create a JSON code.

D: > NGT_Practical > 229726_create.html > ...

```
<!DOCTYPE html>
<html>
<head>
    <title> Creating JSON </title>

</head>
<body>
<script type="text/javascript">
        var data={
                    "Bname":"Kartik",
                    "publisher":"Rolls-Royce Motor Cars",
                    "author":"Motor Cars",
                    "price":"6.22 Cr"
                };
        document.writeln(JSON.stringify(data)+"<br/>");

        document.writeln(JSON.stringify(data,["Bname","author","price"])+"<br/>");

        document.writeln(JSON.stringify(data,["Bname","author","price"],5));

    </script>
</body>
</html>
```

Output:-

{"Bname":"Kartik","publisher":"Rolls-Royce Motor Cars","author":"Motor Cars","price":"6.22 Cr"}
{"Bname":"Kartik","author":"Motor Cars","price":"6.22 Cr"}
{ "Bname": "Kartik", "author": "Motor Cars", "price": "6.22 Cr" }

Parsing JSON

1.Parsing.html

D: > NGT_Practical > 229726_Parsing1.html > ...

```
<html>
<head>
<title>parsing</title>
</head>
<body>
<script type="text/javascript">
    var data='{"name":"Kartik","sem":5,"type":"programming"}';

    var json_obj=JSON.parse(data,function(name,value)
                {
                    return value;
                }
            );
    document.writeln(json_obj.name);    for(key in json_obj)
    {
    document.write("</br>"+key+ ":" +json_obj[key]);
    }
    var json_obj=JSON.parse(data,function(name,value)
                {
                    if(name=="sem")
                    {
                    return undefined;
                    }
                    else
                    {
                    return value;
                    }
                }               );
    for(key in json_obj)
    {
    document.write("</br>"+key+ ":" +json_obj[key]);
    }
</script>
</body>
</html>
```

Output:-

Kartik
name:Kartik
sem:5
type:programming
name:Kartik
type:programming

2.Parsing.html

D: > NGT_Practical > 229726_Parsing2.html > ...

```
<!DOCTYPE html>
<html>
<body>

<script type="text/javascript">  var data1='{"Bname":"Kartik","publisher":"Rolls-Royce Motor Cars"}';  var docx=JSON.parse(data1);

for(key in docx)
{
document.writeln(key+" :"+docx[key]+"<br/>");
}

</script>
</body>
</html>
```

Output:-

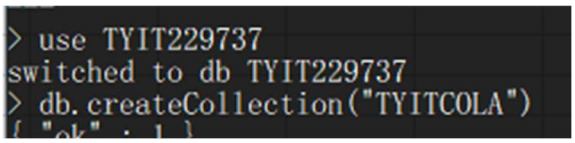
Bname :Kartik
publisher :Rolls-Royce Motor Cars

Practical 11

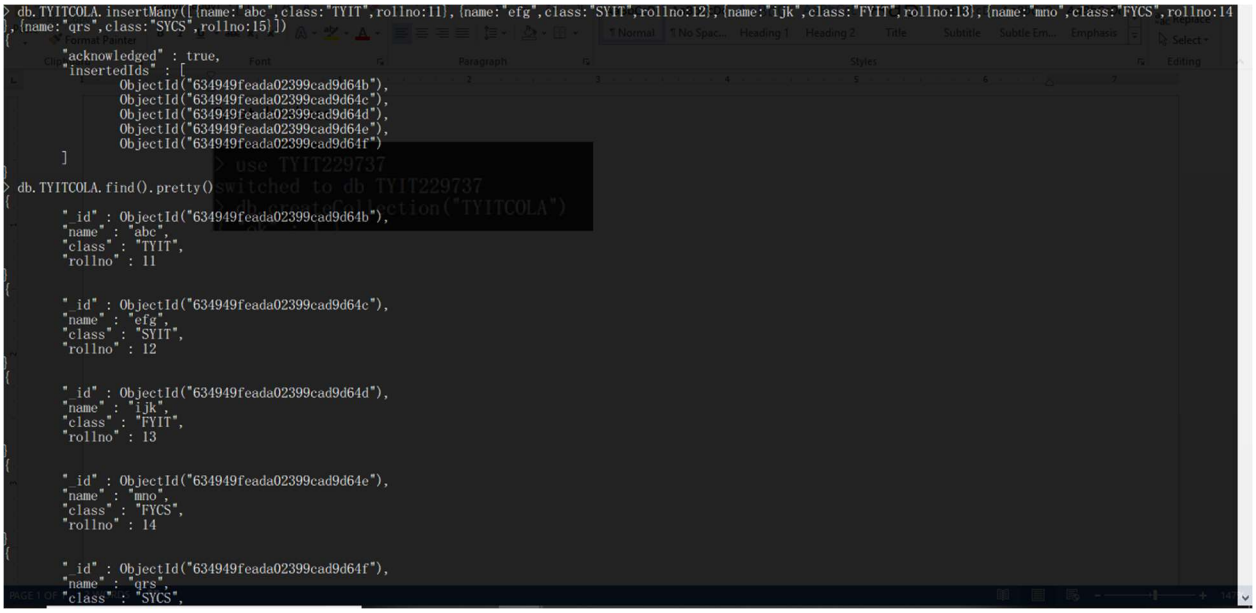
Create a JSON file and import it to MongoDB

11 Exporting data from mongodb to csv file

Creat database



Enter value init

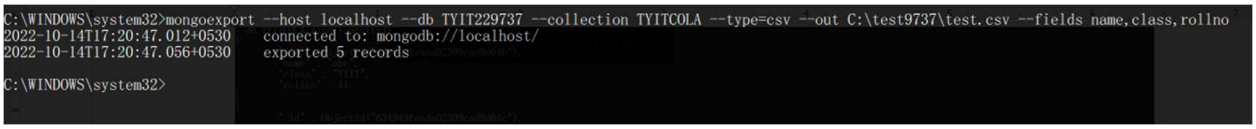


Download mongo DATABASE Tool and extract

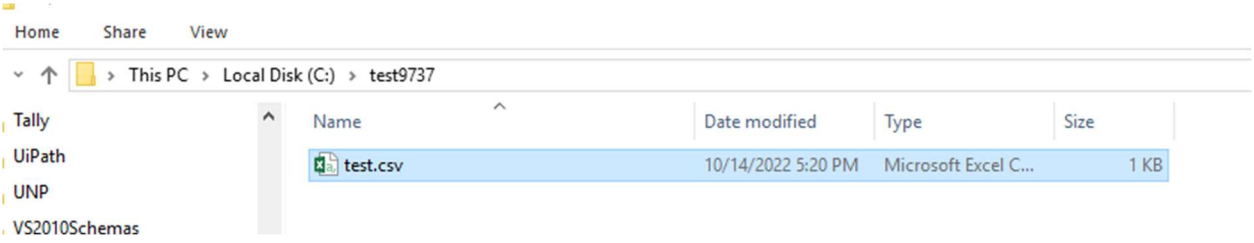
https://www.mongodb.com/try/download/database-tools

After extract copy the bin file values and paste them in mongodb bin file

Then open new cmd and write



It will create a new file in that particular location



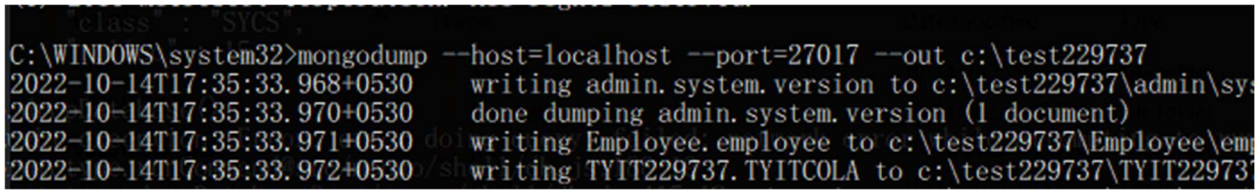
Clipboard Font

A1 na

	A	B	C	D
1	name	class	rollno	
2	abc	TYIT	11	
3	efg	SYIT	12	
4	ijk	FYIT	13	
5	mno	FYCS	14	
6	qrs	SYCS	15	
7				
8				
9				
10				

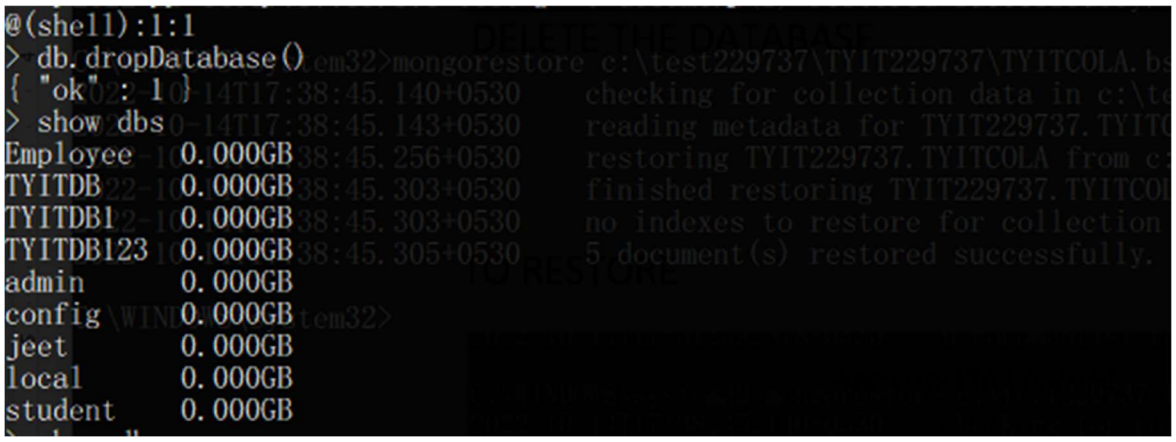
##

Backup DATABASE

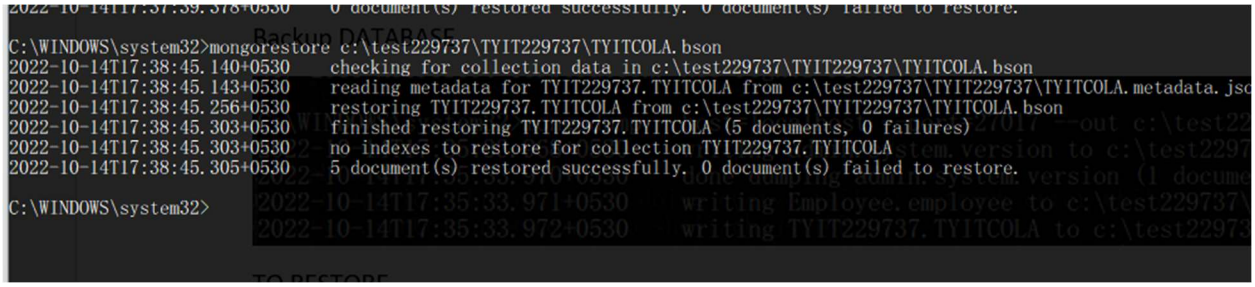


Step 1 :: Type the following command to create baackup of follingdatabase

Step 2 :: Drop the existing database from the mongoshell



Step 3::lets us now us e the backup to copy of drop database through cmd



Step 4:; after running the restore command the database will get restore in the mongodb through the mongo shell we can see the drop database again

```
local      0.000GB
student    0.000GB
> show dbs
Employee    0.000GB
TYIT229737  0.000GB
TYITDB      0.000GB
TYITDB1     0.000GB
TYITDB123   0.000GB
admin       0.000GB
config      0.000GB
jeet        0.000GB
local       0.000GB
student     0.000GB
>
```

Import the json data to the mongodb

Step 1::type following command in cmd to import the json file to the mongodb

```
C:\WINDOWS\system32>mongoimport --db test123 --collection TYITCOLA --file D:\restaurants.json
2022-10-14T18:07:43.205+0530    connected to: mongodb://localhost/
2022-10-14T18:07:43.599+0530    3772 document(s) imported successfully. 0 document(s) failed to import.
```

Mongodb will automatically create database as per the name given in the command after db keyword the same command will automatically create the collection in the database and command will import the json file in the created databasefrom the specified file path after –file keyword

Step 2::we can check same database and imported file in that database

```
To permanently disable this reminder, run the foll
> show dbs
Employee    0.000GB
TYIT229737  0.000GB
TYITDB      0.000GB
TYITDB1     0.000GB
TYITDB123   0.000GB
admin       0.000GB
config      0.000GB
jeet        0.000GB
local       0.000GB
student     0.000GB
test123     0.001GB
> use test123
switched to db test123
> db.TYITCOLA.find().pretty()
{
        "_id" : ObjectId("6349581748f3a6088be84a14"),
        "address" : {
                "building" : "351",
```

Backup of the database

Step 1: Type the following command to create the backup of

```
C:\Windows\system32>mongodump --host=localhost --port=27017 --out c:\test
2022-10-14T17:40:49.740+0530    writing admin.system.version to c:\test\admin\system.version.bson
2022-10-14T17:40:49.746+0530    done dumping admin.system.version (1 document)
2022-10-14T17:40:49.747+0530    writing TYIT229721.TYITCOL to c:\test\TYIT229721\TYITCOL.bson
2022-10-14T17:40:49.755+0530    done dumping TYIT229721.TYITCOL (5 documents)
```

Step 2: Let us delete existing database from the mongodb through the mongo shell, for this we will use the following command

db.dropDatabase()

```
MongoDB Enterprise > db.dropDatabase()
{ "dropped" : "TYIT229721", "ok" : 1 }
MongoDB Enterprise >
```

Step 3: Let us now use the backup copy of the dropped database from the following command through cmd (admin mode).

mongorestore c:\test\TYIT229721\TYITCOL.bson

```
C:\Windows\system32>mongorestore c:\test\TYIT229721\TYITCOL.bson
2022-10-14T17:50:13.843+0530    checking for collection data in c:\test\TYIT229721\TYITCOL.bson
2022-10-14T17:50:13.844+0530    reading metadata for TYIT229721.TYITCOL from c:\test\TYIT229721\TYITCOL.metadata.json
2022-10-14T17:50:13.861+0530    restoring TYIT229721.TYITCOL from c:\test\TYIT229721\TYITCOL.bson
2022-10-14T17:50:13.903+0530    finished restoring TYIT229721.TYITCOL (5 documents, 0 failures)
2022-10-14T17:50:13.903+0530    no indexes to restore for collection TYIT229721.TYITCOL
2022-10-14T17:50:13.903+0530    5 document(s) restored successfully. 0 document(s) failed to restore.
```

Step 4: After running the restore command the database will get restores in the mongodb through the mongo shell we can see the dropped database again.

show.dbs

```
MongoDB Enterprise > show dbs
TYIT229721  0.000GB
admin       0.000GB
config      0.000GB
local       0.000GB
```

Importing the json data to the mongoDB

[3772 restaurant records]

Step 1: Type the following command in CMD to import the JSON file to the MongoDB.

```
C:\Windows\system32>mongoimport --db test229721 --collection TYITCOL --file C:\Users\NGT\Desktop\Softwares\restaurants.json
2022-10-14T18:07:16.325+0530    connected to: mongodb://localhost/
2022-10-14T18:07:16.919+0530    3772 document(s) imported successfully. 0 document(s) failed to import.
```

MongoDB will automatically create the database as per the name given in the command after "– db" and the same command will automatically create the collection in the database and the command will import the JSON file in the created database from the specified file path after the "—file" keyword.

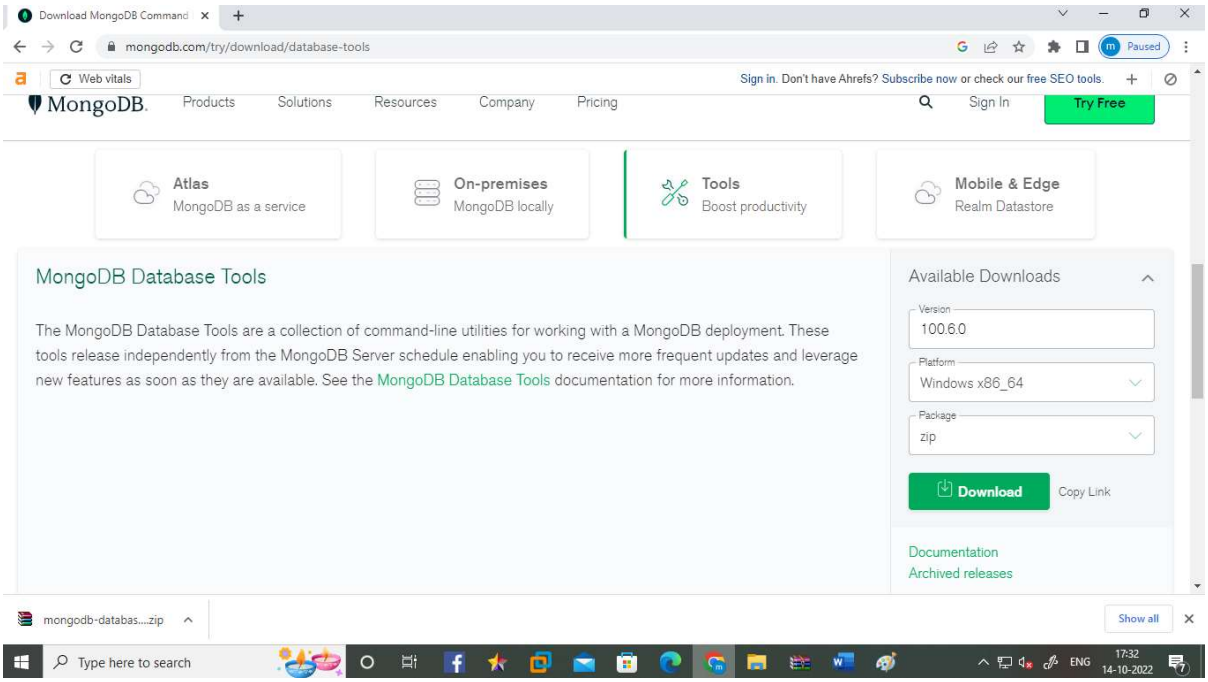
Step 2: We can check the same database and the imported file in the mongo shell.

```
MongoDB Enterprise > show dbs
TYIT229721  0.000GB
admin       0.000GB
config      0.000GB
local       0.000GB
test229721  0.001GB
MongoDB Enterprise > use test229721
switched to db test229721
MongoDB Enterprise > show collections
TYITCOL
MongoDB Enterprise > db.TYITCOL.find().pretty()
{
        "_id" : ObjectId("634957fc26f7592d655fa767"),
        "address" : {
                "building" : "1007",
                "coord" : [
                        -73.856077,
                        40.848447
                ],
                "street" : "Morris Park Ave",
                "zipcode" : "10462"
        },
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "grades" : [
                {
                        "date" : ISODate("2014-03-03T00:00:00Z"),
                        "grade" : "A",
                        "score" : 2
```

Exporting the data from mongoDB to CSV

Step 1: Open Google and search for mongodb database tool.

Step 2: Download the tool files in the zip format.



Step 3: Extract the zip file and copy all the files from the bin folder.

Step 4: Paste the copied file to the build folder of mongodb folder.

Step 5: Now, run the new cmd in the admin mode and type the following command.