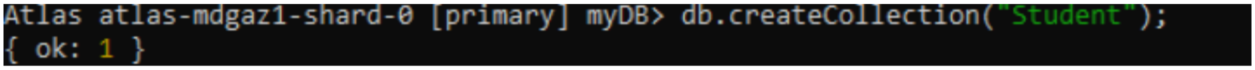
WEEK-1

I. Perform the following DB operations using MongoDB.
1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.

ANS-db.createCollection("Student");



```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.createCollection("Student");
{ ok: 1 }
```

2. Insert appropriate values

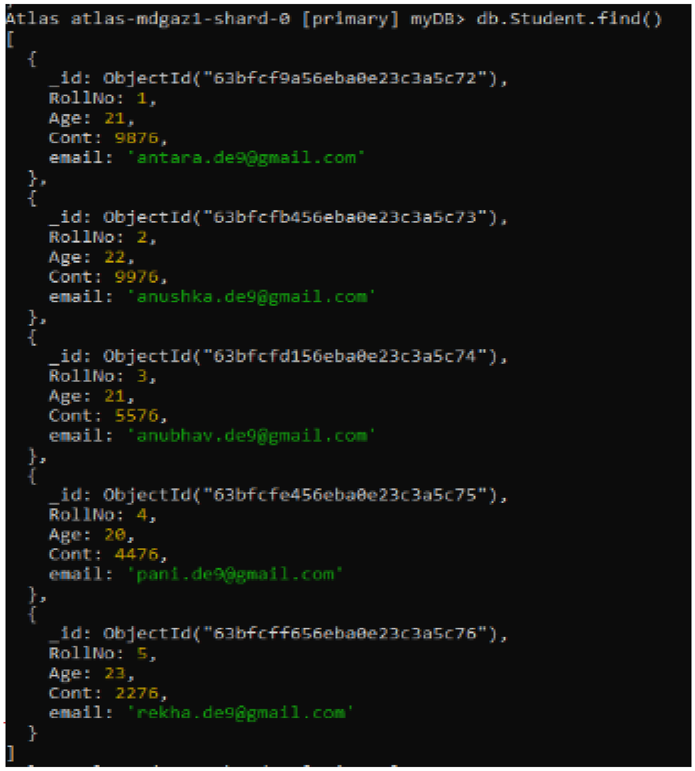
ANS-db.Student.insert({RollNo:1,Age:21,Cont:9876,email:"antara.de9@gmail.com"});

db.Student.insert({RollNo:2,Age:22,Cont:9976,email:"anushka.de9@gmail.com"});

db.Student.insert({RollNo:3,Age:21,Cont:5576,email:"anubhav.de9@gmail.com"});

db.Student.insert({RollNo:4,Age:20,Cont:4476,email:"pani.de9@gmail.com"});

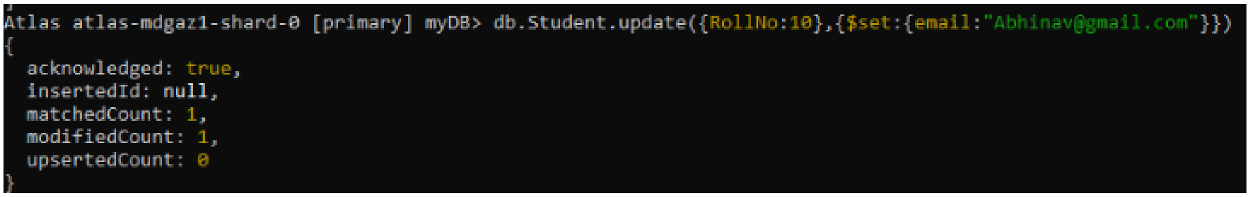
db.Student.insert({RollNo:10,Age:23,Cont:2276,email:"rekha.de9@gmail.com"});
db.Student.find()



```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.Student.find()
[
  {
    _id: ObjectId("63bfcf9a56eba0e23c3a5c72"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'antara.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcfb456eba0e23c3a5c73"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'anushka.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcfd156eba0e23c3a5c74"),
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcfe456eba0e23c3a5c75"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcff656eba0e23c3a5c76"),
    RollNo: 5,
    Age: 23,
    Cont: 2276,
    email: 'rekha.de9@gmail.com'
  }
]
```

3. Write query to update Email-Id of a student with rollno 10.

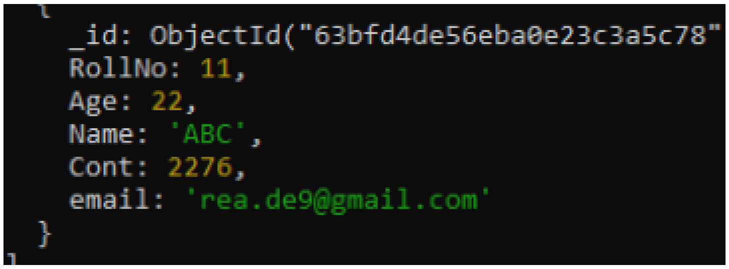
ANS-db.Student.update({RollNo:10},{$set:{email:"Abhinav@gmail.com"}})



```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.Student.update({RollNo:10},{$set:{email:"Abhinav@gmail.com"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

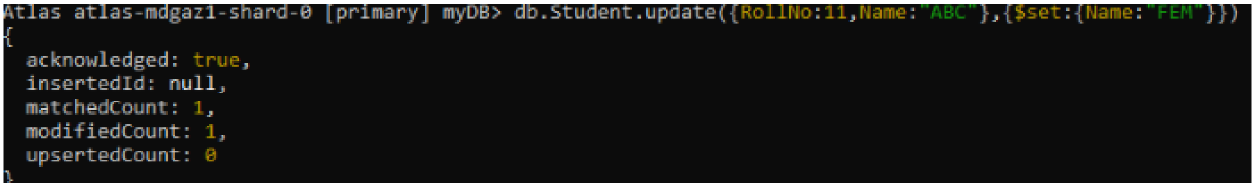
4. . Replace the student name from “ABC” to “FEM” of rollno 11

ANS-db.Student.insert({RollNo:11,Age:22,Name:"ABC",Cont:2276,email:"rea.de9@gmail.com"});

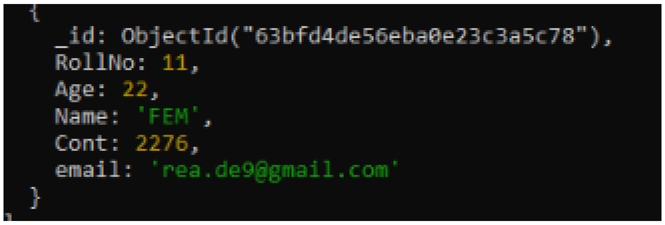


```
  {
    _id: ObjectId("63bfd4de56eba0e23c3a5c78"
    RollNo: 11,
    Age: 22,
    Name: 'ABC',
    Cont: 2276,
    email: 'rea.de9@gmail.com'
  }
]
```

db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})



```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```



```
  {
    _id: ObjectId("63bfd4de56eba0e23c3a5c78"),
    RollNo: 11,
    Age: 22,
    Name: 'FEM',
    Cont: 2276,
    email: 'rea.de9@gmail.com'
  }
]
```

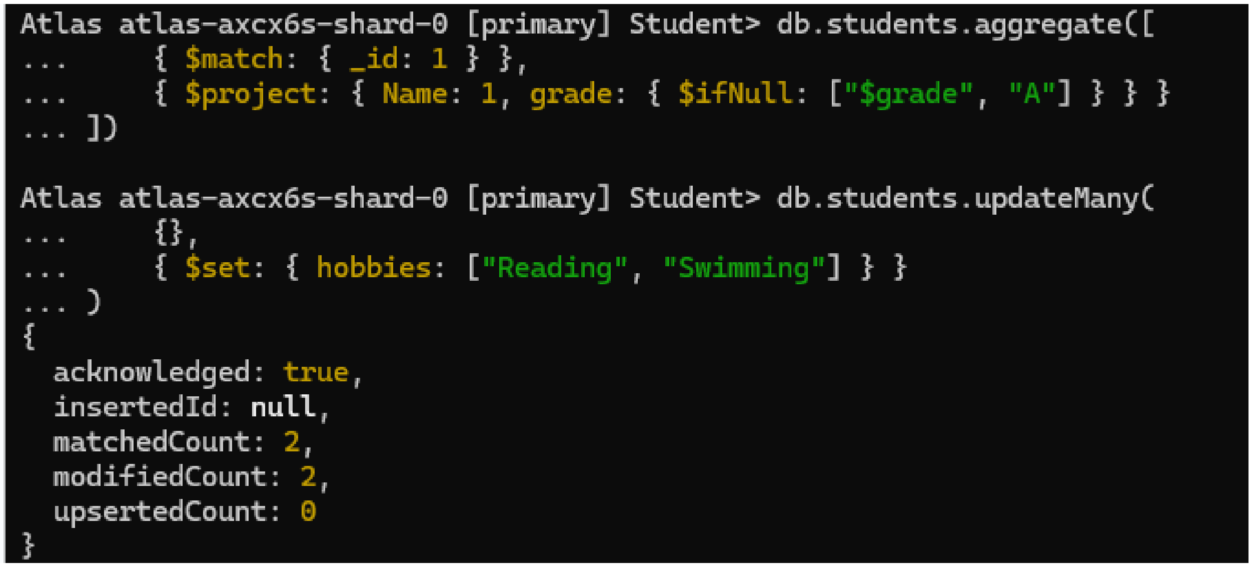
5. Display Student Name and grade(Add if grade is not present)where the _id column is 1.

```
db.students.aggregate([
  { $match: { _id: 1 } },
  { $project: { Name: 1, grade: { $ifNull: ["$grade", "A"] } } }
])
```

6. Update to add hobbies

```
db.students.updateMany(
  {},
  { $set: { hobbies: ["Reading", "Swimming"] } }
```

)



```
Atlas atlas-axcx6s-shard-0 [primary] Student> db.students.aggregate([
...     { $match: { _id: 1 } },
...     { $project: { Name: 1, grade: { $ifNull: ["$grade", "A"] } } }
... ])

Atlas atlas-axcx6s-shard-0 [primary] Student> db.students.updateMany(
...     {},
...     { $set: { hobbies: ["Reading", "Swimming"] } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

7. Find documents where hobbies is set neither to Chess nor to Skating

db.students.find({ hobbies: { $nin: ["Chess", "Skating"] } })

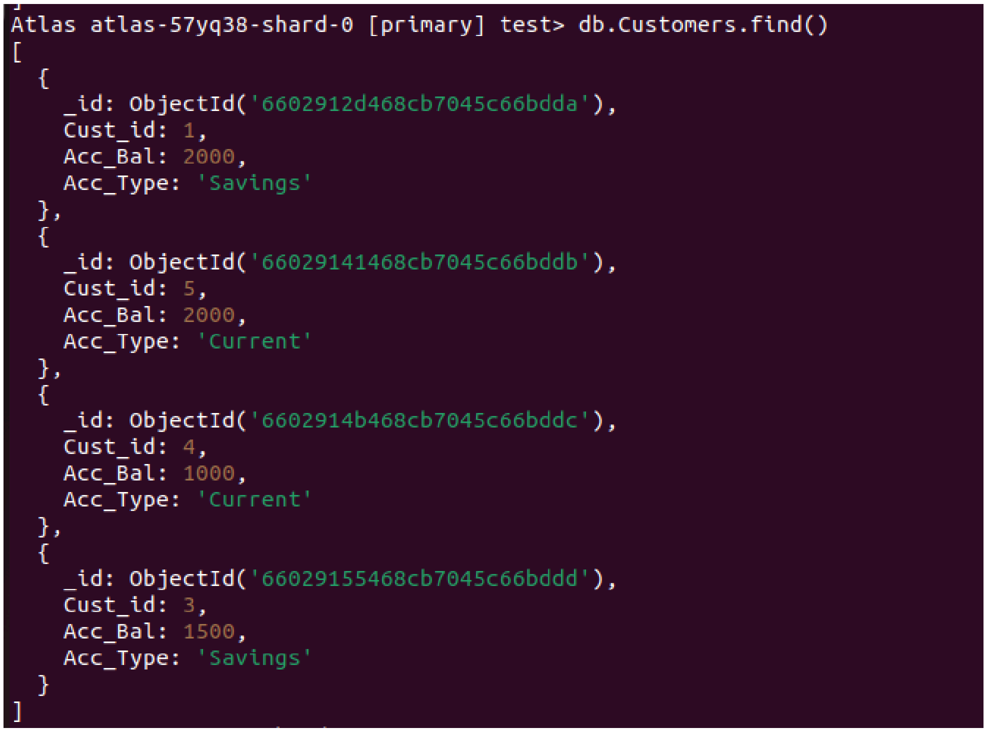
8.Find documents whose name begins with A

db.students.find({ Name: /^A/ })

II. Perform the following DB operations using MongoDB.

1. Create a collection by name Customers with the following attributes.
Cust_id, Acc_Bal, Acc_Type
ANS-1.db.createCollection(“Customer”);

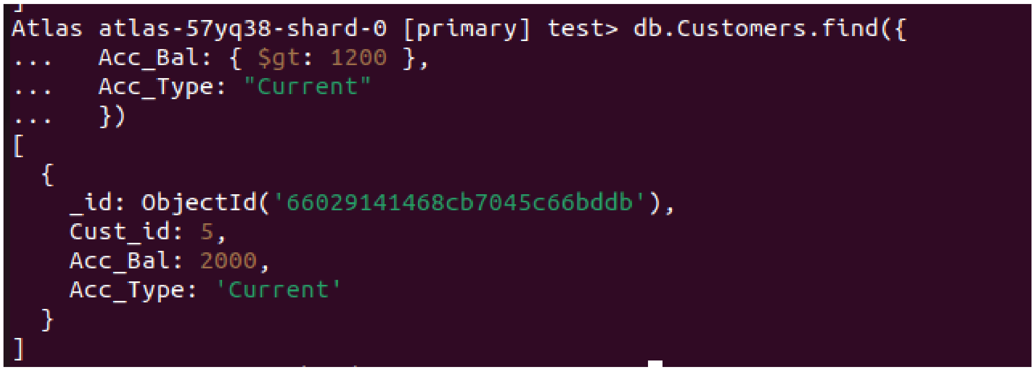
2. Insert at least 5 values into the table
ANS-2.db.Customers.insert({Cust_id:1,Acc_Bal:2000,Acc_Type:"Savings"});
db.Cusotmers.insert({Cust_id:2,Acc_Bal:3000,Acc_Type:"Savings"});
db.Customers.insert({Cust_id:3,Acc_Bal:1500,Acc_Type:"Savings"});
db.Customers.insert({Cust_id:4,Acc_Bal:1000,Acc_Type:"Current"});
db.Customers.insert({Cust_id:5,Acc_Bal:2000,Acc_Type:"Current"});



```
Atlas atlas-57yq38-shard-0 [primary] test> db.Customers.find()
[
  {
    _id: ObjectId('6602912d468cb7045c66bdda'),
    Cust_id: 1,
    Acc_Bal: 2000,
    Acc_Type: 'Savings'
  },
  {
    _id: ObjectId('66029141468cb7045c66bddb'),
    Cust_id: 5,
    Acc_Bal: 2000,
    Acc_Type: 'Current'
  },
  {
    _id: ObjectId('6602914b468cb7045c66bddc'),
    Cust_id: 4,
    Acc_Bal: 1000,
    Acc_Type: 'Current'
  },
  {
    _id: ObjectId('66029155468cb7045c66bddd'),
    Cust_id: 3,
    Acc_Bal: 1500,
    Acc_Type: 'Savings'
  }
]
```

3. Write a query to display those records whose total account balance is greater than 1200 of account type ‘Z’ for each customer_id.

ANS-3. db.Customers.find({
 Acc_Bal: { $gt: 1200 },
 Acc_Type: "Current"
 })



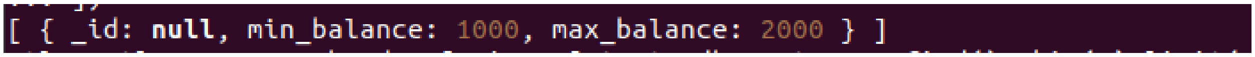
```
Atlas atlas-57yq38-shard-0 [primary] test> db.Customers.find({
...   Acc_Bal: { $gt: 1200 },
...   Acc_Type: "Current"
...   })
[
  {
    _id: ObjectId('66029141468cb7045c66bddb'),
    Cust_id: 5,
    Acc_Bal: 2000,
    Acc_Type: 'Current'
  }
]
```

4. Determine Minimum and Maximum account balance for each customer_i

ANS-4 db.Customers.aggregate([{ $group: { _id: "$Cust_id", min_balance: { $min: "$Acc_Bal" }, max_balance: { $max: "$Acc_Bal" } } }])

5. Sort the documents based on Customer ID in ascending order and Account Balance in descending order

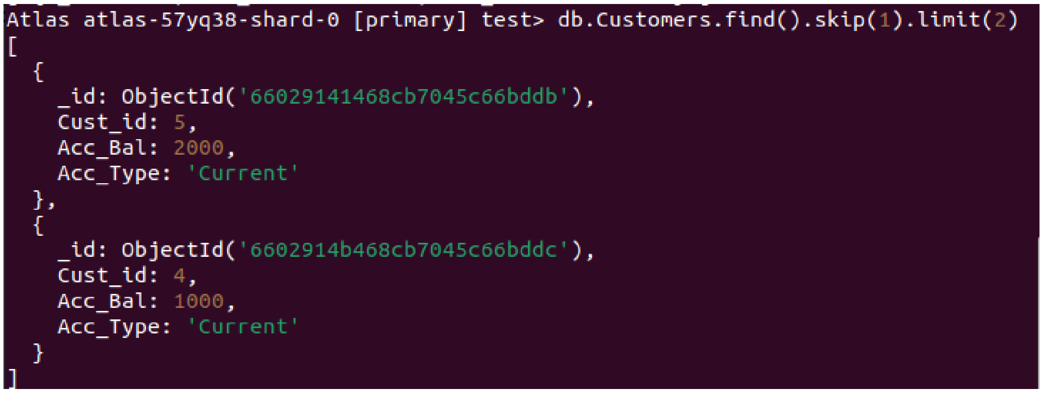
```
db.Customers.aggregate([
 {
        $match: {
        Acc_Type: "Current"
        }
 },
 {
        $group: {
        _id: "$customer_id",
        min_balance: { $min: "$Acc_Bal" },
        max_balance: { $max: "$Acc_Bal" }
        }
 },
 {
        $sort: {
        "_id": 1,
        "max_balance": -1
        }
 }
])
```



```
[ { _id: null, min_balance: 1000, max_balance: 2000 } ]
```

6.Display only 2nd and 3rd records from the collection

```
db.Customers.find().skip(1).limit(2)
```



```
Atlas atlas-57yq38-shard-0 [primary] test> db.Customers.find().skip(1).limit(2)
[
  {
    _id: ObjectId('66029141468cb7045c66bddb'),
    Cust_id: 5,
    Acc_Bal: 2000,
    Acc_Type: 'Current'
  },
  {
    _id: ObjectId('6602914b468cb7045c66bddc'),
    Cust_id: 4,
    Acc_Bal: 1000,
    Acc_Type: 'Current'
  }
]
```