Program 1

1. Create a database “Student” and a collection “Student_Details” with the following attributes: USN, Age, Contact, Email and Name

```
Atlas atlas-q9t5lc-shard-0 [primary] test> use Student
switched to db Student

Atlas atlas-q9t5lc-shard-0 [primary] Student> db.createCollection('Student_Details')
{ ok: 1 }

Atlas atlas-q9t5lc-shard-0 [primary] Student> show collections
Student_Details
```

2. Insert appropriate values

```
Atlas atlas-q9t5lc-shard-0 [primary] Student> db.Student_Details.insertMany([
{USN:1,Name:"Asmi",Contact:"9845874655",Age:20,Email:"asmi@gmail.com"},
... {USN:2,Name:"Anand",Contact:"9641894562",Age:20,Email:"anand@gmail.com"},
... {USN:3,Name:"Abhay",Contact:"9845632222",Age:20,Email:"abhay@gmail.com"},
...{USN:4,Name:"Sanam",Contact:"8554712554",Age:21,Email:"sanam@gmail.com"},
... {USN:5,Name:"Manoj",Contact:"9877784200",Age:21,Email:"manoj@gmail.com"},
...{USN:6,Name:"Prakriti",Contact:"9663654102",Age:21,Email:"prakriti@gmail.com"},
... {USN:7,Name:"Zoya",Contact:"8796654122",Age:20,Email:"zoya@gmail.com"}]
... )


{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("660a7e337ac89cf8a39d8c22"),
    '1': ObjectId("660a7e337ac89cf8a39d8c23"),
    '2': ObjectId("660a7e337ac89cf8a39d8c24"),
    '3': ObjectId("660a7e337ac89cf8a39d8c25"),
    '4': ObjectId("660a7e337ac89cf8a39d8c26"),
    '5': ObjectId("660a7e337ac89cf8a39d8c27"),
    '6': ObjectId("660a7e337ac89cf8a39d8c28")
  }
}
```

3. Write a query to update the Email of a student with USN 1.

```
Atlas atlas-q9t5lc-shard-0 [primary] Student>
db.Student_Details.updateOne({USN:2},{$set:{Email:"ashana@gmail.com"}})

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

4. Replace the student name from “Asmi” to “Ashana” of USN 1.

```
Atlas atlas-q9t5lc-shard-0 [primary] Student> db.Student_Details.updateOne({USN:1},{$set:
{Name:"Ashana"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}

Atlas atlas-q9t5lc-shard-0 [primary] Student> db.Student_Details.find()
[
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c22"),
    USN: 1,
    Name: 'Ashana',
    Contact: '9845874655',
    Age: 20,
    Email: 'ashana@gmail.com'
  },
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c23"),
    USN: 2,
    Name: 'Anand',
    Contact: '9641894562',
    Age: 20,
    Email: 'anand@gmail.com'
  },
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c24"),
    USN: 3,
    Name: 'Abhay',
    Contact: '9845632222',
    Age: 20,
    Email: 'abhay@gmail.com'
  },
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c25"),
    USN: 4,
    Name: 'Sanam',
    Contact: '8554712554',
    Age: 21,
    Email: 'sanam@gmail.com'
  },
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c26"),
    USN: 5,
    Name: 'Manoj',
    Contact: '9877784200',
    Age: 21,
    Email: 'manoj@gmail.com'
  },
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c27"),
    USN: 6,
    Name: 'Prakriti',
    Contact: '9663654102',
    Age: 21,
    Email: 'prakriti@gmail.com'
  },
  {
    _id: ObjectId("660a7e337ac89cf8a39d8c28"),
    USN: 7,
    Name: 'Zoya',
    Contact: '8796654122',
    Age: 20,
    Email: 'zoya@gmail.com'
  }]
```

Program 2

1. Create a collection by name Customers with the following attributes: Cust_id, Acc_Bal, Acc_Type

```
Atlas atlas-q9t5lc-shard-0 [primary] Student> use Bank
switched to db Bank
```

```
Atlas atlas-q9t5lc-shard-0 [primary] Student>db.createCollection(‘Customers’)
{ ok: 1 }
```

2. Insert at least 5 values into the table

```
Atlas atlas-q9t5lc-shard-0 [primary] Bank>
db.Customers.insertMany([{CustID:455,AccBal:4500,AccType:'Z'},
{CustID:489,AccBal:2000,AccType:'Y'}, {CustID:854,AccBal:1200,AccType:'X'},
{CustID:522,AccBal:1500,AccType:'Z'},
{CustID:230,AccBal:2500,AccType:'Z'}])

{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("660a83347ac89cf8a39d8c29"),
    '1': ObjectId("660a83347ac89cf8a39d8c2a"),
    '2': ObjectId("660a83347ac89cf8a39d8c2b"),
    '3': ObjectId("660a83347ac89cf8a39d8c2c"),
    '4': ObjectId("660a83347ac89cf8a39d8c2d")
  }
}
```

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

```
Atlas atlas-q9t5lc-shard-0 [primary] Bank> db.Customers.aggregate([ { $match: {
AccType: "Z" } }, { $group: { _id: "$CustID", totalBalance: { $sum: "$AccBal" }
} }, { $match: { totalBalance: { $gt: 1200 } } }] )
```

```
[
  { _id: 455, totalBalance: 4500 },
  { _id: 522, totalBalance: 1500 },
  { _id: 230, totalBalance: 2500 }
]
```

4. Determine Minimum and Maximum account balance for each CustID.

```
Atlas atlas-q9t5lc-shard-0 [primary] Bank> db.Customers.aggregate([ { $group: { _id: "CustID", minBalance: { $min: "$AccBal" }, maxBalance: { $max: "$AccBal" } } }] )
```

```
[ { _id: 'CustID', minBalance: 1200, maxBalance: 4500 } ]
```