Q1. Make a schema first and then insert 6 documents. • Roll_no=[1,2,3,4,5,] • Name=["Ram","Alex","John","Bob","Mukesh",”Danny”] • Age=[20,19,40,55,30,28] • Salary=[546.7,333.4,666.7,678.4,245.6,546.3] • City=["A","B","C","D","E",”F”] • Phone_no=[123,456,122,444,567,892].

Ans:- {

roll_no: Number,

name: String,

age: Number,

salary: Number,

city: String,

phone_no: Number

}

```
test> use collegeDB
switched to db collegeDB
collegeDB> db.student.insertMany([
...   { roll_no: 1, name: "Ram",    age: 20, salary: 546.7, city: "A", phone
_no: 123 },
...   { roll_no: 2, name: "Alex",   age: 19, salary: 333.4, city: "B", phone
_no: 456 },
...   { roll_no: 3, name: "John",   age: 40, salary: 666.7, city: "C", phone
_no: 122 },
...   { roll_no: 4, name: "Bob",    age: 55, salary: 678.4, city: "D", phone
_no: 444 },
...   { roll_no: 5, name: "Mukesh", age: 30, salary: 245.6, city: "E", phone
_no: 567 },
...   { roll_no: 6, name: "Danny",  age: 28, salary: 546.3, city: "F", phone
_no: 892 }
... ])
...
```

Q2. Write a query to update the name of RAM to SAM.

```
collegeDB> db.student.updateOne({name: "Ram"}, {$set: {name:"Sam"} })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Q3. Write a query to display only the cities present in that collection.

```
collegeDB> db.student.distinct("city")
[ 'A', 'B', 'C', 'D', 'E', 'F' ]
```

Q4. Write a query to update the salary by 10%.

```
collegeDB> db.student.updateMany({}, { $mul : {salary:1.1}})
```

Q5. Write a query to display all the documents in ascending and descending order of age.

```
collegeDB> db.student.find().sort({age: -1})
```

Q6. Write a query to display all the documents with City → A,B,C

```
collegeDB> db.student.find({city : {$in :["A","B","C"]}})
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2621'),
    roll_no: 1,
    name: 'Sam',
    age: 20,
    salary: 601.3700000000001,
    city: 'A',
    phone_no: 123
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2622'),
    roll_no: 2,
    name: 'Alex',
    age: 19,
    salary: 366.74,
    city: 'B',
    phone_no: 456
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2623'),
    roll_no: 3,
    name: 'John',
    age: 40,
    salary: 733.3700000000001,
    city: 'C',
    phone_no: 122
  }
]
```

Q7. Write a query to display only two documents from the entire collection.

```
collegeDB> db.student.find().limit(2)
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2621'),
    roll_no: 1,
    name: 'Sam',
    age: 20,
    salary: 601.3700000000001,
    city: 'A',
    phone_no: 123
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2622'),
    roll_no: 2,
    name: 'Alex',
    age: 19,
    salary: 366.74,
    city: 'B',
    phone_no: 456
  }
]
```

Q8. Write a query to delete a document with ROLL_NO:5.

```
collegeDB> db.student.deleteOne({roll_no: 5})
```

Q9. Write a query to display all the documents with AGE greater than 20.

```
collegeDB> db.student.find({age: {$gt: 20}})
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2623'),
    roll_no: 3,
    name: 'John',
    age: 40,
    salary: 733.3700000000001,
    city: 'C',
    phone_no: 122
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2624'),
    roll_no: 4,
    name: 'Bob',
    age: 55,
    salary: 746.24,
    city: 'D',
    phone_no: 444
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2626'),
    roll_no: 6,
    name: 'Danny',
    age: 28,
    salary: 600.93,
    city: 'F',
    phone_no: 892
  }
]
```

Q10. Write a query to display all the documents with AGE less than 20

```
collegeDB> db.student.find({ age: { $lt: 20 } })
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2622'),
    roll_no: 2,
    name: 'Alex',
    age: 19,
    salary: 366.74,
    city: 'B',
    phone_no: 456
  }
]
collegeDB> db.student.find({ age: 20 })
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2621'),
    roll_no: 1,
    name: 'Sam',
    age: 20,
    salary: 601.3700000000001,
    city: 'A',
    phone_no: 123
  }
]
collegeDB> db.student.find({ age: { $ne: 20 } })
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2622'),
    roll_no: 2,
    name: 'Alex',
    age: 19,
    salary: 366.74,
    city: 'B',
    phone_no: 456
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2623'),
    roll_no: 3,
    name: 'John',
    age: 40,
    salary: 733.3700000000001,
```

Q11. Write a query to display all the documents with AGE equals to 20.

```
collegeDB> db.student.find({ age: 20 })
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2621'),
    roll_no: 1,
    name: 'Sam',
    age: 20,
    salary: 601.3700000000001,
    city: 'A',
    phone_no: 123
  }
]
```

Q12. Write a query to display all the documents with AGE not equals to 20.

```
collegeDB> db.student.find({ age: { $ne: 20 } })
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2622'),
    roll_no: 2,
    name: 'Alex',
    age: 19,
    salary: 366.74,
    city: 'B',
    phone_no: 456
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2623'),
    roll_no: 3,
    name: 'John',
    age: 40,
    salary: 733.3700000000001,
    city: 'C',
    phone_no: 122
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2624'),
    roll_no: 4,
    name: 'Bob',
    age: 55,
    salary: 746.24,
    city: 'D',
    phone_no: 444
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2626'),
    roll_no: 6,
    name: 'Danny',
    age: 28,
    salary: 600.93,
    city: 'F',
    phone_no: 892
  }
```

Q13. Write a query to display all the documents where AGE is greater than equals to 30.

```
collegeDB> db.student.find({age:{$gte: 30}})
[
  {
    _id: ObjectId('697436c2643dcc21bf1e2623'),
    roll_no: 3,
    name: 'John',
    age: 40,
    salary: 733.3700000000001,
    city: 'C',
    phone_no: 122
  },
  {
    _id: ObjectId('697436c2643dcc21bf1e2624'),
    roll_no: 4,
    name: 'Bob',
    age: 55,
    salary: 746.24,
    city: 'D',
    phone_no: 444
  }
]
```