Sr.	Solution

Lab-3 [MongoDB]

Create Database with Name: **BANK_INFO**
Insert below data into the Collection.

Deposite

ACTNO	CNAME	BNAME	AMOUNT	ADATE
101	ANIL	VRCE	1000.00	1-3-95
102	SUNIL	AJNI	5000.00	4-1-96
103	MEHUL	KAROLBAGH	3500.00	17-11-95
104	MADHURI	CHANDI	1200.00	17-12-95
105	PRMOD	M.G. ROAD	3000.00	27-3-96
106	SANDIP	ANDHERI	2000.00	31-3-96
107	SHIVANI	VIRAR	1000.00	5-9-95
108	KRANTI	NEHRU PLACE	5000.00	2-7-95

From the above given collection perform the following queries:

1. Retrieve/Display every document of your collections.
   db.Deposite.find()

2. Retrieve/Display every document of your collection. (Use option pretty)
   db.Deposite.find().pretty()

3. Display only one documents of your collection. (Use findone)
   db.Deposite.findOne()

4. Display documents whose Account Number is 101.
   db.Deposite.find({ACTNO:101})
   db.Deposite.find({ ACTNO :{$eq:101}})
   OR
   db.Deposite.find({ACTNO:"101"})
   db.Deposite.find({ ACTNO :{$eq:"101"}})

   db.Deposite.find({ ACTNO :{$ne:"101"}}) | **Used For Not Equal**
   db.Deposite.find({ ACTNO :{$gt:"101"}}) | **Used for greater than**
   db.Deposite.find({ ACTNO :{$gte:"101"}}) | **Used for greater than equal**
   db.Deposite.find({ ACTNO :{$lt:"101"}}) | **Used for less than**
   db.Deposite.find({ ACTNO :{$lte:"101"}}) | **Used for less than equal**
   db.Deposite.find({ ACTNO :{$in:[101,105,107]}}) | **IN** Operator
   db.Deposite.find({ ACTNO :{$nin:[101,105,107]}}) |**NOT IN** Operator
   db.Deposite.find().count() | **For Total Records**

5. Display documents whose Account Number is less than 103.
   db.Deposite.find({ ACTNO :{$lt:103}})

6. Display documents whose Account Number is greater than 102 and Customer Name is Arjun.
   db.Deposite.find({$and:[{ACTNO:{$gt:102}},{CName:"Arjun"}]})

OR

db.Deposite.find({$or:[{ACTNO:{$gt:102}},{CName:"Arjun"}]})

7. Display documents whose Account Number is 105 or 108 using IN.
   db.Deposite.find({ ACTNO :{$in:[105,108]}})

8. Display documents whose Account Number is not greater than 105.
   db.Deposite.find({ ACTNO: { $not: { $gt: 105 } } })

9. Display documents with CNAME, BNAME and AMOUNT fields.
   db.Deposite.find({},{CNAME:1,BNAME:1,AMOUNT:1}

10. Display Nagpur city branch's documents with CNAME, BNAME and AMOUNT fields.
    db.Deposite.find({BNAME: "Nagpur"},{_id: 0, CNAME: 1, BNAME: 1, AMOUNT: 1})

11. Display every document of your collection on ascending order by CNAME and descending order by AMOUNT.
    db.Deposite .find().sort({CNAME: 1, AMOUNT: -1})

12. Display only two documents of your collection. (Use LIMIT | Use Customer Collection)
    db.Deposite.find().limit(2)

13. Display from 3rd documents of your collection. (Use SKIP | Use Borrow Collection)
    db.Deposite.find().limit(1).skip(2)

    OR
    db.Deposite.find().skip(2)

14. Display the count of documents in your collection. (Use Deposite Collection)
    db.Deposite.find().count()

15. Display the documents whose name starts with S in your collection.
    db.Deposite.find({CNAME: /^S/i})

16. Display the documents whose name starts with S or M in your collection.
    db.Deposite.find({CNAME: /^[S,M]/i})

17. Display the documents whose name starts with A and having 5 characters in your collection.
    db.Deposite.find({CNAME: /^A..../})

18. Display the documents whose name starts with A to M in your collection.
    db.Deposite.find({CNAME: /^[A-M]/})

19. Display the sum of amount in your collection. (Use Deposite Collection)
    db.Deposite.aggregate([{ $group: { _id: null , total: { $sum: "$AMOUNT" }} }])

20. Display branch wise sum of amount in your collection. (Use Deposite Collection)
    db.Deposite.aggregate([{ $group: { _id: "$BNAME" , total: { $sum: "$AMOUNT" }} }])

21. Update name of Anil to Arjun and also Branch Name to "DPS".
    db.Deposite.updateMany({CNAME: "ANIL"},{$set: {CNAME:"ARJUN", BNAME: "DPS"}})

22. Delete the document whose Branch Name is DPS.
    db.Deposite.drop({BNAME: "DPS"})

23. Delete the collection deposite.
    db.Deposite.drop()

24. Drop BANK_INFO database.
    db.dropDatabase()

Lab-4 [MongoDB]

Create Database with Name: **Employee_Info**
Create following Collection under Employee_Info database.

Employee					
EID	**EName**	**Gender**	**JoiningDate**	**Salary**	**City**
1	Nick	Male	01-JAN-13	4000	London
2	Julian	Female	01-OCT-14	3000	New York
3	Roy	Male	01-JUN-16	3500	London
4	Tom	Male	*NULL*	4500	London
5	Jerry	Male	01-FEB-13	2800	Sydney
6	Philip	Male	01-JAN-15	7000	New York
7	Sara	Female	01-AUG-17	4800	Sydney
8	Emily	Female	01-JAN-15	5500	New York
9	Michael	Male	*NULL*	6500	London
10	John	Male	01-JAN-15	8800	London

From the above given table perform the following queries:

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

```
db.Employee.find()
```

2. Write a MongoDB query to display the fields EID, Name, Gender, and salary for all the documents in the collection employee.

```
db.Employee.find({},{EID:true,EName:true,Gender:true,Salary:true})
```

3. Write a MongoDB query to display the fields EID, Name, Gender, and City, but exclude the field _id for all the documents in the collection employee.

```
db.Employee.find({},{EID:true,EName:true,Gender:true,Salary:true,_id:
false})
```

4. Write a MongoDB query to display the fields salary, but exclude the field _id for all the documents in the collection employee.

```
db.Employee.find({},{Salary:true, _id:false})
```

5. Write a MongoDB query to display all the Employees which are in the city London.

```
db.Employee.find({City: "London"})
```

6. Write a MongoDB query to display the first 5 EID which are in the city Sydney.

```
db.Employee.find({},{EID:true},{City: "Sydney"}).limit(5);
```

7. Write a MongoDB query to display the next 2 Employees after skipping the first 2 which are in the city New York.

```
db.Employee.find({City: "New York"}).skip(2).limit(2);
```

8. Write a MongoDB query to display the count of documents in your collection.

```
db.Employee.find().count()
```

9. Write a MongoDB query to display the sum of salary in your collection.

```
db.Employee.aggregate([{ $group: {  id: null , total: { $sum:
"$Salary"}}}])
```

10. Write a MongoDB query to display the documents whose employee name starts with S or M in your collection.

```
db.Employee.find({EName: /^[S,M]/i})
```

11. Write a MongoDB query to find the employee Id, name, city, and salary for those employees which contain 'Phi' as the first three letters of their name.

```
db.Employee.find(
{EName: /^Phi/},
{
EID : true,
EName:true,City:true,
Salary :true
}
);
```

12. Write a MongoDB query to find the employee Id, name, city, and gender for those employees which contain 'ael' as the last three letters of their name.

```
db.Employee.find(
{EName: /ael$/},
{
```

```
EID : true,
EName:true,City:true,
Gender :true
}
);
```

13. Write a MongoDB query to find the name, joining date, and city for those employees which contain 'dne' as three letters somewhere in their city name.

```
db.Employee.find(
{City: /.dne./},
{
EName:true,JoiningDate:true,
City :true
}
);
```

14. Write a MongoDB query to find the employee Id, name, city, and joining date for those employees which do not belong to the city London or Sydney.

```
db.Employee.find({City
:{$nin:["London","Sydney"]}},{EID:true,EName:true,City:true,JoiningDa
te:true})
```

15. Write a MongoDB query to find the name and city for those employees which salary is more than 10000.

```
db.Employee.find({Salary :{$gt:10000}},{EName:true,City:true})
```

16. Write a MongoDB query to arrange the name of the employees in ascending order along with all the columns.

```
db.Employee.find().sort({"EName":1});
```

17. Write a MongoDB query to arrange the city of the employees in descending order along with all the columns.

```
db.Employee.find().sort({"City":-1});
OR
db.Employee.find().sort({City:-1});
```

18. Write a MongoDB query to arrange the name of the employees in ascending order and, the city should be in descending order.

```
db.Employee.find().sort({EName:1,City : -1,})
```

19. Write a MongoDB query to display city wise sum of salary from employee collection.

```
db.Employee.aggregate([{$group : {_id : "$City", salary_sum : {$sum : "$Salary"}}}])
```

20. Write a MongoDB query to delete the document whose city name is London.

```
db.Employee.remove({City:"London"})
```

OR

```
db.Employee.deleteMany({City:"London"})
```