Lab-10 :

Part – A (Use collection “Student” created in Lab-9) :

1. Update the age of John's to 31.

→ db.Student.updateOne({ name : "John" } , { $set : { age : 31 } })

2. Update the city of all students from 'New York' to 'New Jersey'.

→ db.Student.updateMany({ city : "New York" } , { $set : { city : "New Jersey" } })

3. Set isActive to false for every student older than 35.

→ db.student.updateMany({ age : { $gt : 35 } } , { $set : { isActive : false } })

4. Increment the age of all students by 1 year.

→ db.Student.updateMany({} , { $inc : { age :1 } })

5. Set the city of 'Eva' to 'Cambridge'.

→ db.Student.updateOne({ name : "Eva" } , { $set : { city : "Cambridge" } })

6. Update 'Sophia's isActive status to false.

→ db.Student.updateOne({ name : "Sophia" } , { $set : { isActive : false } })

7. Update the city field of student aged below 30 to 'San Diego'.

→ db.Student.updateMany({ age : { $lt : 30 } } , { $set : { city : "San Diego" } })

8. Rename the age field to years for all documents.

→ db.Student.updateMany({} , { $rename : { age : "years" } })

9. Update 'Nick' to make him active (isActive = true).

→ db.Student.updateOne({ name : "Nick" } , { $set : { isActive : true } })

10. Update all documents to add a new field country with the value 'USA'.

→ db.Student.updateMany({} , { $set : { country : "USA" } })

11. Update 'David's city to 'Orlando'.

→ db.Student.updateOne({ name : "David" } , { $set : { city : "Or.lando" } })

12. Multiply the age of all students by 2.

→ db.Student.updateMany({} , { $mul : { years : 2 } })

13. Unset (remove) the city field for 'Tom'.

→ db.Student.updateOne({ name : "Tom" } , { $unset : { country : "" } })

14. Add a new field premiumUser and to true for users older than 30.

→ db.Student.updateMany({ years : { $gt : 30 } } , { $set : { premiumUser : true } })

15. Set isActive to true for 'Jane'.

→ db.Student.updateOne({ name : "Jane" } , { $set : { isActive : true } })

16. Update isActive field of 'Lucy' to false.

→ db.Student.updateOne({ name : "Lucy" } , { $set : { isActive : false } })

17. Delete a document of 'Nick' from the collection.

→ db.Student.deleteOne({ name : "Nick" })

18. Delete all students who are inactive (isActive = false).

→ db.Student.deleteMany({ isActive : false })

19. Delete all students who live in 'New York'.

→ db.Student.deleteMany({ country : "New York" })

20. Delete all the students aged above 35.

→ db.Student.deleteMany({ years : { $gt : 35 } })

21. Delete a student named ''Olivia'' from the collection.

→ db.Student.deleteOne({ name : "Olivia" })

22. Delete all the students whose age is below 25.

→ db.Student.deleteMany({ years : { $lt : 25 } })

23. Delete the first student whose isActive field is true.

→ db.Student.deleteOne({ isActive : true })

24. Delete all students from 'Los Angeles'.

→ db.Student.deleteMany({ city : "Los Angeles" })

25. Delete all students who have city field missing.

→ db.Student.deleteMany({ city : { $exists : false } })

26. Rename 'city' field to 'location' for all documents.

→ db.Student.updateMany({} , { $rename : { city : "location" } })

27. Rename the name field to FullName for 'John'.

→ db.Student.updateOne({ name : "John" } , { $rename : { name : "FullName" } })

28. Rename the isActive field to status for all documents.

→ db.Student.updateMany({} , { $rename : { isActive : "status" } })

29. Rename age to yearsOld for everyone from 'San Francisco' student only.

→ db.Student.updateMany({ location : "San Francisco" } , { $rename : { years : "yearsOld" } })

30. Create a Capped Collection named “Employee” as per follows :

a. Ecode and Ename are compulsory fields

b. Datatype of EID is int, Ename is string, Age is int and City is string Insert following documents into above “Employee” collection.

{"Ecode": 1, "Ename": "John"}

{"Ecode ": 2, "Ename": "Jane", "age": 25, "city": "Los Angeles"}

{"Ecode ": 3, "Ename": "Tom", "age": 35}

{"Ecode ": 4, "Ename": "Lucy", "age": 28, "city": "San Francisco", "isActive": true}

{"Ename": "Dino"}

→ db.createCollection("Employee", {

```
 capped: true, size: 1024,  max: 100,
 validator: {
   $jsonSchema: {
    bsonType: "object",
    required: ["Ecode", "Ename"],
    properties: {
     Ecode: {bsonType: "int"},
     Ename: {bsonType: "string"},
     Age: {bsonType: "int”},
     City: {bsonType: "string"}
    }
   }
 }
})
```

→ db.Employee.insertMany([

```
{ Ecode : 1 , Ename : "John" } ,
{ Ecode : 2 , Ename : "Jane" , age : 25 , city : "Los Angeles" } ,
{ Ecode: 3 , Ename : "Tom" , age : 35 } ,
{ Ecode: 4 , Ename : "Lucy" , age : 28 , city : "San Francisco" , isActive : true } ,
{ Ename : "Dino" } ] )
```