WORKING WITH MONGO DB

NAME : NIHARIKA B S
USN : 1BM19CS100

1. CREATE DATABASE IN MONGODB.

```
bmsce@bmsce-Precision-T1700:~$ mongo
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("567e15b6-00ef-48ad-8af9-604f6e8de048") }
MongoDB server version: 3.6.8
Server has startup warnings:
2022-04-13T19:39:21.099+0530 I STORAGE  [initandlisten]
2022-04-13T19:39:21.099+0530 I STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is stron
gly recommended with the WiredTiger storage engine
2022-04-13T19:39:21.099+0530 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/cor
e/prodnotes-filesystem
2022-04-13T19:39:24.590+0530 I CONTROL  [initandlisten]
2022-04-13T19:39:24.590+0530 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for
 the database.
2022-04-13T19:39:24.590+0530 I CONTROL  [initandlisten] **          Read and write access to data and
 configuration is unrestricted.
2022-04-13T19:39:24.590+0530 I CONTROL  [initandlisten]
>
> use Niharika_db
switched to db Niharika_db
```

2. CRUD (CREATE, READ, UPDATE, DELETE) OPERATIONS

To create a collection by the name “Student”. Let us take a look at the collection list prior to the creation of the new collection “Student”.

```
> db.createCollection("Student");
{ "ok" : 1 }
```

Create a collection by the name “Students” and store the following data in it.

```
> db.Student.insert({_id:1,StudName:"MichelleJacintha",Grade:"VII",Hobbies:"InternetSurfing"});
WriteResult({ "nInserted" : 1 })
```

```
> db.Student.insert({_id:2,StudName:"MikeHassan",Grade:"VII",Hobbies:"Swimming"});
WriteResult({ "nInserted" : 1 })
> db.Student.update({_id:3,StudName:"AryanDavid",Grade:"VII"},{$set:{Hobbies:"Skating"}},{upsert:true});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 3 })
> db.Student.insert({_id:4,StudName:"DuaLipa",Grade:"VII",Hobbies:"Singing"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:5,StudName:"RajeshBharadwaj",Grade:"VII",Hobbies:"Badminton"});
WriteResult({ "nInserted" : 1 })
```

FIND METHOD

A. To search for documents from the “Students” collection based on certain search criteria.

```
> db.Student.find({StudName:"DuaLipa"});
{ "_id" : 4, "StudName" : "DuaLipa", "Grade" : "VII", "Hobbies" : "Singing" }
```

B. To display only the StudName and Grade from all the documents of the Students collection. The identifier_id should be suppressed and NOT displayed.

C. To find

```
> db.Student.find({},{StudName:1,Grade:1,_id:0});
{ "StudName" : "MichelleJacintha", "Grade" : "VII" }
{ "StudName" : "MikeHassan", "Grade" : "VII" }
{ "Grade" : "VII", "StudName" : "AryanDavid" }
{ "StudName" : "DuaLipa", "Grade" : "VII" }
{ "StudName" : "RajeshBharadwaj", "Grade" : "VII" }
```

those documents where the Grade is set to ‘VII’

D. To find

```
> db.Student.find({Grade:{$eq:'VII'}}).pretty();
{
        "_id" : 1,
        "StudName" : "MichelleJacintha",
        "Grade" : "VII",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : 2,
        "StudName" : "MikeHassan",
        "Grade" : "VII",
        "Hobbies" : "Swimming"
}
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
{
        "_id" : 4,
        "StudName" : "DuaLipa",
        "Grade" : "VII",
        "Hobbies" : "Singing"
}
{
        "_id" : 5,
        "StudName" : "RajeshBharadwaj",
        "Grade" : "VII",
        "Hobbies" : "Badminton"
}
```

those documents from the Students collection where the Hobbies is set to either ‘singing’ or is set to ‘Skating’.

```
> db.Student.find({Hobbies :{ $in: ['Singing','Skating']}}).pretty ();
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
{
        "_id" : 4,
        "StudName" : "DuaLipa",
        "Grade" : "VII",
        "Hobbies" : "Singing"
}
```

E. To find documents from the Students collection where the StudName begins with “R”.

F. To find

```
> db.Student.find({StudName:/^R/}).pretty();
{
        "_id" : 5,
        "StudName" : "RajeshBharadwaj",
        "Grade" : "VII",
        "Hobbies" : "Badminton"
```

documents from the Students collection where the StudNamehas an “a” in any position.

```
> db.Student.find({StudName:/a/}).pretty();
{
        "_id" : 1,
        "StudName" : "MichelleJacintha",
        "Grade" : "VII",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : 2,
        "StudName" : "MikeHassan",
        "Grade" : "VII",
        "Hobbies" : "Swimming"
}
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
{
        "_id" : 4,
        "StudName" : "DuaLipa",
        "Grade" : "VII",
        "Hobbies" : "Singing"
}
```

G. To find the number of documents in the Students collection.

```
> db.Student.count();
5
```

H. To sort the documents from the Students collection in the descending order of StudName.

```
> db.Student.find().sort({StudName:-1}).pretty();
{
        "_id" : 5,
        "StudName" : "RajeshBharadwaj",
        "Grade" : "VII",
        "Hobbies" : "Badminton"
}
{
        "_id" : 2,
        "StudName" : "MikeHassan",
        "Grade" : "VII",
        "Hobbies" : "Swimming"
}
{
        "_id" : 1,
        "StudName" : "MichelleJacintha",
        "Grade" : "VII",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : 4,
        "StudName" : "DuaLipa",
        "Grade" : "VII",
        "Hobbies" : "Singing"
}
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
```

3.Save Method :

Save() method will insert a new document, if the document with the _id does not exist. If it exists it will replace the exisiting document.

```
> db.Student.save({StudName:"Vamsi", Grade:"VI"})
WriteResult({ "nInserted" : 1 })
```

Add a new field to existing Document:

```
> db.Student.update({_id:4},{$set:{Location:"Network"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find({});
{ "_id" : 1, "StudName" : "MichelleJacintha", "Grade" : "VII", "Hobbies" : "InternetSurfing" }
{ "_id" : 2, "StudName" : "MikeHassan", "Grade" : "VII", "Hobbies" : "Swimming" }
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : "Skating" }
{ "_id" : 4, "StudName" : "DuaLipa", "Grade" : "VII", "Hobbies" : "Singing", "Location" : "Network" }
{ "_id" : 5, "StudName" : "RajeshBharadwaj", "Grade" : "VII", "Hobbies" : "Badminton" }
{ "_id" : ObjectId("62569a60a083074f5c1a00a8"), "StudName" : "Vamsi", "Grade" : "VI" }
```

Remove the field in an existing Document

```
> db.Student.update({_id:4},{$unset:{Location:"Network"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find({});
{ "_id" : 1, "StudName" : "MichelleJacintha", "Grade" : "VII", "Hobbies" : "InternetSurfing" }
{ "_id" : 2, "StudName" : "MikeHassan", "Grade" : "VII", "Hobbies" : "Swimming" }
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : "Skating" }
{ "_id" : 4, "StudName" : "DuaLipa", "Grade" : "VII", "Hobbies" : "Singing" }
{ "_id" : 5, "StudName" : "RajeshBharadwaj", "Grade" : "VII", "Hobbies" : "Badminton" }
{ "_id" : ObjectId("62569a60a083074f5c1a00a8"), "StudName" : "Vamsi", "Grade" : "VI" }
```

Finding Document based on search criteria suppressing few fields

To find

```
> db.Student.find({_id:1},{StudName:1,Grade:1,_id:0});
{ "StudName" : "MichelleJacintha", "Grade" : "VII" }
```

those documents where the Grade is not set to ‘VII’

To find

```
> db.Student.find({Grade:{$ne:'VII'}}).pretty();
{
        "_id" : ObjectId("62569a60a083074f5c1a00a8"),
        "StudName" : "Vamsi",
        "Grade" : "VI"
}
```

documents from the Students collection where the StudName ends with n.

to set a particular field value to NULL

```
> db.Student.find({StudName:/n$/}).pretty();
{
        "_id" : 2,
        "StudName" : "MikeHassan",
        "Grade" : "VII",
        "Hobbies" : "Swimming"
}
```

```
> db.Student.update({_id:3},{$set:{Hobbies:null}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find({});
{ "_id" : 1, "StudName" : "MichelleJacintha", "Grade" : "VII", "Hobbies" : "InternetSurfing" }
{ "_id" : 2, "StudName" : "MikeHassan", "Grade" : "VII", "Hobbies" : "Swimming" }
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : null }
{ "_id" : 4, "StudName" : "DuaLipa", "Grade" : "VII", "Hobbies" : "Singing" }
{ "_id" : 5, "StudName" : "RajeshBharadwaj", "Grade" : "VII", "Hobbies" : "Badminton" }
{ "_id" : ObjectId("62569a60a083074f5c1a00a8"), "StudName" : "Vamsi", "Grade" : "VI" }
>
```

Count the number of documents in Student Collections

```
> db.Student.count()
6
```

Count the number of documents in Student Collections with grade :VII

```
> db.Student.count({Grade:"VII"})
5
```

food database using mongodb

Create a collection by name “food” and add to each document add a “fruits” array

```
> db.food.insert( { _id:1, fruits:['grapes','mango','apple'] } )
WriteResult({ "nInserted" : 1 })
> db.food.insert( { _id:2, fruits:['grapes','mango','cherry'] } )
WriteResult({ "nInserted" : 1 })
> db.food.insert( { _id:3, fruits:['banana','mango'] } )
WriteResult({ "nInserted" : 1 })
```

To find those documents from the “food” collection which has the “fruits array” constitute of “grapes”, “mango” and “apple”.

```
> db.food.find ( {fruits: ['grapes','mango','apple'] } ).pretty();
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
```

To find all the documets from the food collection which have elements mango and grapes in the array “fruits”

```
> db.food.find({fruits:{$all:["mango","grapes"]}})
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }
```

update on Array:
using particular id replace the element present in the 1st index position of the fruits array with apple