

Nikhil

Abraham

1BM19CS101

LAB 2

20-04-2022

- i) Create a database for Students and Create a Student Collection (_id,Name, USN,Semester, Dept_Name, CGPA, Hobbies(Set)).
- ii) Insert required documents to the collection.
- iii) First Filter on "Dept_Name:CSE" and then group it on "Semester" and compute the Average CGPA for that semester and filter those documents where the "Avg_CGPA" is greater than 7.5.
- iv) Command used to export MongoDB JSON documents from "Student" Collection into the "Students" database into a CSV file "Output.txt".

```
> db.Student.insert({_id:1,Name:"Michell",usn:"abc01",semester:"VI",dept_name:"cse",cgpa:8.0,Hobbies:"music"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:2,Name:"nehal",usn:"abc02",semester:"VI",dept_name:"cse",cgpa:8.5,Hobbies:"art"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:3,Name:"shawn",usn:"abc03",semester:"VI",dept_name:"cse",cgpa:9.1,Hobbies:"swimming"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:4,Name:"bhavana",usn:"abc04",semester:"VI",dept_name:"cse",cgpa:7.8,Hobbies:"dance"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:5,Name:"cathrin",usn:"abc05",semester:"VI",dept_name:"cse",cgpa:8.5,Hobbies:"trekking"});
WriteResult({ "nInserted" : 1 })
```

```
> db.Student.find({});
{ "_id" : 1, "Name" : "Michell", "usn" : "abc01", "semester" : "VI", "dept_name" : "cse", "cgpa" : 8, "Hobbies" : "music" }
{ "_id" : 2, "Name" : "nehal", "usn" : "abc02", "semester" : "VI", "dept_name" : "cse", "cgpa" : 8.5, "Hobbies" : "art" }
{ "_id" : 3, "Name" : "shawn", "usn" : "abc03", "semester" : "VI", "dept_name" : "cse", "cgpa" : 9.1, "Hobbies" : "swimming" }
{ "_id" : 4, "Name" : "bhavana", "usn" : "abc04", "semester" : "VI", "dept_name" : "cse", "cgpa" : 7.8, "Hobbies" : "dance" }
{ "_id" : 5, "Name" : "cathrin", "usn" : "abc05", "semester" : "VI", "dept_name" : "cse", "cgpa" : 8.5, "Hobbies" : "trekking" }
```

```
> db.Student.aggregate({$match:{dept_name:"cse"}},{ $group:{_id:"$semester",AvgCGPA:{$avg:"$cgpa"}},{ $match:{AvgCGPA:{$gt:7.5}}});
{ "id" : "VI", "AvgCGPA" : 8.379999999999999 }
>
```

```
bmsce@bmsce-Precision-T1700:~$ mongoexport --host localhost --db student --collection Student --csv --out /home/bmsce/Desktop/
output_stud.txt --fields "_id","Name","usn","semester","dept_name","cgpa","Hobbies";
2022-04-20T14:54:26.549+0530 csv flag is deprecated; please use --type=csv instead
2022-04-20T14:54:26.551+0530 connected to: localhost
2022-04-20T14:54:26.552+0530 exported 5 records
bmsce@bmsce-Precision-T1700:~$
```

```
Open  [+]
```

output_stud.txt
~/Desktop

1	_id	Name	usn	semester	dept_name	cgpa	Hobbies
2	1	Michell	abc01	VI	cse	8	music
3	2	nehal	abc02	VI	cse	8.5	art
4	3	shawn	abc03	VI	cse	9.1	swimming
5	4	bhavana	abc04	VI	cse	7.8	dance
6	5	cathrin	abc05	VI	cse	8.5	trekking

2) Create a mongodb collection Bank. Demonstrate the following by choosing fields of your choice.

1. Insert three documents
2. Use Arrays (Use Pull and Pop operation)
3. Use Index
4. Use Cursors
5. Updation

```
> use mydb
> db.createCollection("Bank");
{ "ok" : 1 }
> db.Bank.insert({id:1,name:"neha",type:"savings",contact:["9852364185","080-2258964"]});
WriteResult({ "nInserted" : 1 })
> db.Bank.insert({id:2,name:"bindu",type:"current",contact:["9852658818","080-2252468"]});
WriteResult({ "nInserted" : 1 })
> db.Bank.insert({id:3,name:"tarun",type:"savings",contact:["9852611156","080-2244468"]});
WriteResult({ "nInserted" : 1 })
>
```

```

> db.Bank.find({}).pretty();
{
  "_id" : ObjectId("625fd31304119c2f168b103c"),
  "id" : 1,
  "name" : "neha",
  "type" : "savings",
  "contact" : [
    "9852364185",
    "080-2258964"
  ]
}
{
  "_id" : ObjectId("625fd33d04119c2f168b103d"),
  "id" : 2,
  "name" : "bindu",
  "type" : "current",
  "contact" : [
    "9852658818",
    "080-2252468"
  ]
}
{
  "_id" : ObjectId("625fd36b04119c2f168b103e"),
  "id" : 3,
  "name" : "tarun",
  "type" : "savings",
  "contact" : [
    "9852611156",
    "080-2244468"
  ]
}

```

```

> db.Bank.updateMany({id:1},{ $pop:{contact:1}});
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.Bank.find({}).pretty();
{
  "_id" : ObjectId("625fd31304119c2f168b103c"),
  "id" : 1,
  "name" : "neha",
  "type" : "savings",
  "contact" : [
    "9852364185"
  ]
}

```

```
> db.Bank.updateMany({},{$pull:{contact:"080-2244468"}});
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 1 }
> db.Bank.find({}).pretty();
[
```

```
  { "_id" : ObjectId("625fd31304119c2f168b103c"),
    "id" : 1,
    "name" : "neha",
    "type" : "savings",
    "contact" : [
      "9852364185"
    ]
  },
```

```
  { "_id" : ObjectId("625fd33d04119c2f168b103d"),
    "id" : 2,
    "name" : "bindu",
    "type" : "current",
    "contact" : [
      "9852658818",
      "080-2252468"
    ]
  },
```

```
  { "_id" : ObjectId("625fd36b04119c2f168b103e"),
    "id" : 3,
    "name" : "tarun",
    "type" : "savings",
    "contact" : [
      "9852611156"
    ]
  }
]
```

```

> db.Bank.createIndex({Name:1,Type:1},{name:"Find current account holder"});
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
> db.Bank.getIndexes()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_",
    "ns" : "Neha.Bank"
  },
  {
    "v" : 2,
    "key" : {
      "Name" : 1,
      "Type" : 1
    },
    "name" : "Find current account holder",
    "ns" : "Neha.Bank"
  }
]

```

- 1) Using MongoDB,
 - i) Create a database for Faculty and Create a Faculty Collection(Faculty_id, Name, Designation ,Department, Age, Salary, Specialization(Set)).
 - ii) Insert required documents to the collection.
 - iii) First Filter on “Dept_Name:MECH” and then group it on “Designation” and compute the Average Salary for that Designation and filter those documents where the “Avg_Sal” is greater than 650000.
 - iv) Demonstrate usage of import and export commands

```

}
> db.createCollection("faculty");
{ "ok" : 1 }
> db.faculty.insert({_id:1,name:"Dr. Balaraman Ravindran",designation:"Professor",department:"CSE",age:45,salary:100000,specialization:['python','mysql','sklearn','tensorflow']});
WriteResult({ "nInserted" : 1 })
> db.faculty.insert({_id:2,name:"Dr. Mahadev Ghorki",designation:"Assistant Professor",department:"CSE",age:35,salary:80000,specialization:['python','numpy','sklearn','tensorflow','java']});
WriteResult({ "nInserted" : 1 })
> db.faculty.insert({_id:3,name:"Dr. Praveen Borade",designation:"Associate Professor",department:"ME",age:40,salary:75000,specialization:['autocad','aerodynamics','thermal physics']});
WriteResult({ "nInserted" : 1 })
> db.faculty.insert({_id:4,name:"Dr. Madhav Nayak",designation:"Assistant Professor",department:"ME",age:37,salary:95000,specialization:['autocad','flight-dynamics','Finite Element Analysis']});
WriteResult({ "nInserted" : 1 })
> db.faculty.aggregate ( {$match:{department:"ME"}}, {$group : {_id : "$designation", AverageSal :{$avg:"$salary"} } }, {$match:{AverageSal:{$gt:50000}}});
{ "_id" : "Associate Professor", "AverageSal" : 75000 }
{ "_id" : "Assistant Professor", "AverageSal" : 95000 }

```

2) Consider a table "Product" with the following columns:

Product_id

ProductName

ManufacturingDate

Price

Quantity

Write MongoDB queries for the following:

- 1) To display only the product name from all the documents of the product collection.
- 2) To display only the Product ID, ExpiryDate as well as the quantity from the document of the product collection where the _id column is 1.
- 3) To find those documents where the price is not set to 45000.
- 4) To find those documents from the Product collection where the quantity is set to 30 and the product name is set to 'LEDTV'.
- 5) To find documents from the Product collection where the Product name ends in 'r'.

```
> db.createCollection("product");
{ "ok" : 1 }
> db.product.insert({pid:1,pname:"keyboard",mdate:2001,price:1800,quantity:2});
WriteResult({ "nInserted" : 1 })
> db.product.insert({pid:2,pname:"mouse",mdate:2005,price:1500,quantity:5});
WriteResult({ "nInserted" : 1 })
> db.product.insert({pid:3,pname:"monitor",mdate:2015,price:10000,quantity:9});
WriteResult({ "nInserted" : 1 })
> db.product.insert({pid:4,pname:"motherboard",mdate:2021,price:15000,quantity:4});
WriteResult({ "nInserted" : 1 })
> db.product.find({}, {pname:1,_id:0})
{ "pname" : "keyboard" }
{ "pname" : "mouse" }
{ "pname" : "monitor" }
{ "pname" : "motherboard" }
> db.product.find({pid:1},{pid:1,_id:0,mdate:1,quantity:1});
{ "pid" : 1, "mdate" : 2001, "quantity" : 2 }
> db.product.find({price:{$ne:15000}},{pname:1,_id:0});
{ "pname" : "keyboard" }
{ "pname" : "mouse" }
{ "pname" : "motherboard" }
> db.product.find({pid:1},{pid:1,_id:0,mdate:1,quantity:1});
{ "pid" : 1, "mdate" : 2001, "quantity" : 2 }
> db.product.find({price:{$ne:15000}},{pname:1,_id:0});
{ "pname" : "keyboard" }
{ "pname" : "mouse" }
{ "pname" : "monitor" }
{ "pname" : "motherboard" }
> db.product.find({$and:[{quantity:{$eq:9}},{pname:{$eq:"monitor"}}]},{pname:1,_id:0})
{ "pname" : "monitor" }
> db.product.find({pname:/d$/},{pname:1,quantity:1,_id:0})
{ "pname" : "keyboard", "quantity" : 2 }
{ "pname" : "motherboard", "quantity" : 4 }
```

3) Create a mongodb collection Hospital. Demonstrate the following by choosing fields of your choice.

1. Insert three documents
2. Use Arrays (Use Pull and Pop operation)
3. Use Index
4. Use Cursors

5. Updation

```
> db.createCollection("hospital");
{ "ok" : 1 }
> db.hospital.insert({_id:1, Name: "Anshuman Agarwal", age:23, diseases:["fever", "diarrhoea", "wheezing", "gastritis"]});
WriteResult({ "nInserted" : 1 })
> db.hospital.insert({_id:2, Name: "Pinky Chaubey", age:35, diseases:["fever", "nausea", "food infection", "indigestion", "kidney stones"]});
WriteResult({ "nInserted" : 1 })
> db.hospital.insert({_id:3, Name: "Amresh Chowpati", age:63, diseases:["hyperglycemia", "diabetes mellitus", "food poisoning", "cold"]});
WriteResult({ "nInserted" : 1 })
> db.hospital.updateMany({}, {$pull:{diseases:"fever"}});
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 2 }
> db.hospital.updateOne({_id:1}, {$pop:{diseases:-1}});
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

```

> db.hospital.updateMany({},{$pull:{diseases:"fever"}});
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 2 }
> db.hospital.updateOne({_id:1},{$pop:{diseases:-1}});
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.hospital.find({"diseases.2":"nausea"});
> db.hospital.find({"diseases.1":"nausea"});
> d.hospital.find();
uncaught exception: ReferenceError: d is not defined :
@(shell):1:1
> db.hospital.find();
{ "_id" : 1, "Name" : "Anshuman Agarwal", "age" : 23, "diseases" : [ "wheezing", "gastritis" ] }
{ "_id" : 2, "Name" : "Pinky Chaubey", "age" : 35, "diseases" : [ "nausea", "food infection", "indigestion", "kidney stones" ] }
{ "_id" : 3, "Name" : "Amresh Chowpati", "age" : 63, "diseases" : [ "hyperglycemia", "diabetes mellitus", "food poisoning", "cold" ] }
> db.hospital.find({"diseases.0":"nausea"});
{ "_id" : 2, "Name" : "Pinky Chaubey", "age" : 35, "diseases" : [ "nausea", "food infection", "indigestion", "kidney stones" ] }
> db.hospital.update({_id:3},{$set:{'diseases.1':'sarscov'}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>

```