

Name: HRITIK SINGH

USN : IBM19CS063

SECTION : 6-B

BATCH - 1

1) Using MongoDB

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 CPGA for that semester and filter those documents where the “Avg_CPGA” is greater than 7.5.

iv) Command used to export MongoDB JSON documents from “Student” Collection into the “Students” database into a CSV file “O

```
> db.Student.insert({_id:1,Name:"Aravind",USN:"IBM19CS001",Sem:6,Dept_Name:"CSE",CGPA:"9.8",Hobbies:"Badminton"});
writeResult({ "nInserted" : 1 })
> db.Student.insert({_id:2,Name:"Anan",USN:"IBM19EC002",Sem:7,Dept_Name:"ECE",CGPA:"9.1",Hobbies:"Swimming"});
writeResult({ "nInserted" : 1 })
> db.Student.insert({_id:3,Name:"Latha",USN:"IBM19CS003",Sem:6,Dept_Name:"CSE",CGPA:"8.1",Hobbies:"Reading"});
writeResult({ "nInserted" : 1 })
> db.Student.insert({_id:4,Name:"San",USN:"IBM19CS004",Sem:6,Dept_Name:"CSE",CGPA:"6.5",Hobbies:"Cycling"});
writeResult({ "nInserted" : 1 })
> db.Student.insert({_id:5,Name:"Sunan",USN:"IBM19CS004",Sem:5,Dept_Name:"CSE",CGPA:"7.6",Hobbies:"Cycling"});
writeResult({ "nInserted" : 1 })
> db.Student.aggregate($match:{Dept_Name:"CSE"}),{$group:{$Sem,Avg_CPGA:{$avg:"$CGPA"}},{$match:{$CGPA:{$gt:7.5}}});
2022-04-18T14:41:30.990+0530 E QUERY [thread1] ReferenceError: Sem is not defined :
@(<shell>):1158
> db.Student.aggregate($match:{Dept_Name:"CSE"}),{$group:{$_id:$Sem,Avg_CPGA:{$avg:"$CGPA"}},{$match:{$CGPA:{$gt:7.5}}});
> db.Student.find();
{ "_id" : 1, "Name" : "Aravind", "USN" : "IBM19CS001", "Sem" : 6, "Dept_Name" : "CSE", "CGPA" : "9.8", "Hobbies" : "Badminton" }
{ "_id" : 2, "Name" : "Anan", "USN" : "IBM19EC002", "Sem" : 7, "Dept_Name" : "ECE", "CGPA" : "9.1", "Hobbies" : "Swimming" }
{ "_id" : 3, "Name" : "Latha", "USN" : "IBM19CS003", "Sem" : 6, "Dept_Name" : "CSE", "CGPA" : "8.1", "Hobbies" : "Reading" }
> db.Student.aggregate($match:{Dept_Name:"CSE"}),{$group:{$_id:$Sem,Avg_CPGA:{$avg:"$CGPA"}},{$match:{$CGPA:{$gt:7.5}}});
> db.Student.aggregate($match:{Dept_Name:"CSE"}),{$group:{$_id:$Sem,Avg_CPGA:{$avg:"$CGPA"}},{$match:{$Avg_CPGA:{$gt:7.5}}});
> db.Student.aggregate($match:{Dept_Name:"CSE"}),{$group:{$_id:$Sem,Avg_CPGA:{$avg:"$CGPA"}},{$match:{$Avg_CPGA:{$gt:7.5}}});
> First Filter on "Dept_Name:CSE" and then group it on "Semester" and
2022-04-18T14:57:33.423+0530 E QUERY [thread1] SyntaxError: Missing ; before statement @(<shell>):116
> compute the Average CPGA for that semester and filter those documents where the "Avg_CPGA" is greater than 7.5.
[2]+ Stopped mongo
```

- 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

```

> db.createCollection("Bank");
{ "ok" : 1 }
> db.insert({CustID:1, Name:"Trivikram Hegde", Type:"Savings", Contact:["9945678231", "888-22364587"]});
uncaught exception: TypeError: db.insert is not a function:
@shell>:1:1
> db.Bank.insert({CustID:1, Name:"Trivikram Hegde", Type:"Savings", Contact:["9945678231", "888-22364587"]});
writeResult({ "nInserted" : 1 })
> db.Bank.insert({CustID:2, Name:"Vishvesh Bhat", Type:"Savings", Contact:["6325985615", "888-23651452"]});
writeResult({ "nInserted" : 1 })
> db.Bank.insert({CustID:3, Name:"Vaiishak Bhat", Type:"Savings", Contact:["8971456321", "888-33529458"]});
writeResult({ "nInserted" : 1 })
> db.Bank.insert({CustID:4, Name:"Pranod P Parande", Type:"Current", Contact:["8745236589", "888-56324587"]});
writeResult({ "nInserted" : 1 })
> db.Bank.insert({CustID:4, Name:"Shreyas R S", Type:"Current", Contact:["9445678321", "844-65611729", "888-25639856"]});
writeResult({ "nInserted" : 1 })
> db.Bank.find({});
{ "_id" : ObjectId("625d77889329139694f188a2"), "CustID" : 1, "Name" : "Trivikram Hegde", "Type" : "Savings", "Contact" : [ "9945678231", "888-22364587" ] }
{ "_id" : ObjectId("625d77bd9329139694f188a3"), "CustID" : 2, "Name" : "Vishvesh Bhat", "Type" : "Savings", "Contact" : [ "6325985615", "888-23651452" ] }
{ "_id" : ObjectId("625d77e89329139694f188a4"), "CustID" : 3, "Name" : "Vaiishak Bhat", "Type" : "Savings", "Contact" : [ "8971456321", "888-33529458" ] }
{ "_id" : ObjectId("625d78229329139694f188a5"), "CustID" : 4, "Name" : "Pranod P Parande", "Type" : "Current", "Contact" : [ "8745236589", "888-56324587" ] }
{ "_id" : ObjectId("625d78659329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "844-65611729", "888-25639856" ] }
> db.Bank.updateMany({CustID:1},{ $pop:{Contact:1} });
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.Bank.find({});
{ "_id" : ObjectId("625d77889329139694f188a2"), "CustID" : 1, "Name" : "Trivikram Hegde", "Type" : "Savings", "Contact" : [ "9945678231" ] }
{ "_id" : ObjectId("625d77bd9329139694f188a3"), "CustID" : 2, "Name" : "Vishvesh Bhat", "Type" : "Savings", "Contact" : [ "6325985615", "888-23651452" ] }
{ "_id" : ObjectId("625d77e89329139694f188a4"), "CustID" : 3, "Name" : "Vaiishak Bhat", "Type" : "Savings", "Contact" : [ "8971456321", "888-33529458" ] }
{ "_id" : ObjectId("625d78229329139694f188a5"), "CustID" : 4, "Name" : "Pranod P Parande", "Type" : "Current", "Contact" : [ "8745236589", "888-56324587" ] }
{ "_id" : ObjectId("625d78659329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "844-65611729", "888-25639856" ] }

```

```

{ "_id" : ObjectId("625d78059329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "044-656-11729", "088-25419816" ] }
> db.Bank.updateMany({},{$pull:{Contact:"088-25419816"}});
{"acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.Bank.find();
{ "_id" : ObjectId("625d78059329139694f188a2"), "CustID" : 1, "Name" : "Trivikran Hegde", "Type" : "Savings", "Contact" : [ "9945678231" ] }
{ "_id" : ObjectId("625d78059329139694f188a3"), "CustID" : 2, "Name" : "Vishvash Bhat", "Type" : "Savings", "Contact" : [ "6321905015", "080-2-1651452" ] }
{ "_id" : ObjectId("625d78059329139694f188a4"), "CustID" : 3, "Name" : "Vaiubhak Bhat", "Type" : "Savings", "Contact" : [ "8971450321", "088-31-129458" ] }
{ "_id" : ObjectId("625d78059329139694f188a5"), "CustID" : 4, "Name" : "Pranod P Parande", "Type" : "Current", "Contact" : [ "9745230589", "088-56324587" ] }
{ "_id" : ObjectId("625d78059329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "044-656-11729" ] }
> db.Bank.createIndex([Name:1, Type:1],{name:''});
uncaught exception: SyntaxError: expected expression, got '['
@<shell>:1:14
> db.Bank.createIndex([Name:1, Type:1],{name:'Find current account holders'});
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
> db.Bank.find();
{ "_id" : ObjectId("625d78059329139694f188a2"), "CustID" : 1, "Name" : "Trivikran Hegde", "Type" : "Savings", "Contact" : [ "9945678231" ] }
{ "_id" : ObjectId("625d78059329139694f188a3"), "CustID" : 2, "Name" : "Vishvash Bhat", "Type" : "Savings", "Contact" : [ "6321905015", "080-2-1651452" ] }
{ "_id" : ObjectId("625d78059329139694f188a4"), "CustID" : 3, "Name" : "Vaiubhak Bhat", "Type" : "Savings", "Contact" : [ "8971450321", "088-31-129458" ] }
{ "_id" : ObjectId("625d78059329139694f188a5"), "CustID" : 4, "Name" : "Pranod P Parande", "Type" : "Current", "Contact" : [ "9745230589", "088-56324587" ] }
{ "_id" : ObjectId("625d78059329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "044-656-11729" ] }
> db.Bank.getIndexes()
[
  {
    "v" : 2,
    "key" : {

```

```

@<shell>:1:20
> db.Bank.update([{"_id":"625d78059329139694f188a6"},{$set:{CustID:5}},{$unset:{true}}];
uncaught exception: Identifier starts immediately after numeric literal
@<shell>:1:20
> db.Bank.update([{"_id":"625d78059329139694f188a6"},{$set:{CustID:5}},{$unset:{true}}];
writeError: {
  "matched" : 0,
  "upserted" : 1,
  "modified" : 0,
  "_id" : "625d78059329139694f188a6"
}
> db.Bank.find();
{ "_id" : ObjectId("625d78059329139694f188a2"), "CustID" : 1, "Name" : "Trivikran Hegde", "Type" : "Savings", "Contact" : [ "9945678231" ] }
{ "_id" : ObjectId("625d78059329139694f188a3"), "CustID" : 2, "Name" : "Vishvash Bhat", "Type" : "Savings", "Contact" : [ "6321905015", "080-2-1651452" ] }
{ "_id" : ObjectId("625d78059329139694f188a4"), "CustID" : 3, "Name" : "Vaiubhak Bhat", "Type" : "Savings", "Contact" : [ "8971450321", "088-31-129458" ] }
{ "_id" : ObjectId("625d78059329139694f188a5"), "CustID" : 4, "Name" : "Pranod P Parande", "Type" : "Current", "Contact" : [ "9745230589", "088-56324587" ] }
{ "_id" : ObjectId("625d78059329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "044-656-11729" ] }
{ "_id" : "625d78059329139694f188a6", "CustID" : 5 }
> db.Bank.update([{"_id":"625d78059329139694f188a6"},{"CustID:5"},{$set:{Name:"Sumantha K S",Type:"Savings",Contact:["9856321478","011-65807458"]}},{$unset:{true}}];
writeResult: { "matched" : 1, "upserted" : 0, "modified" : 1 }
> db.Bank.find();
{ "_id" : ObjectId("625d78059329139694f188a2"), "CustID" : 1, "Name" : "Trivikran Hegde", "Type" : "Savings", "Contact" : [ "9945678231" ] }
{ "_id" : ObjectId("625d78059329139694f188a3"), "CustID" : 2, "Name" : "Vishvash Bhat", "Type" : "Savings", "Contact" : [ "6321905015", "080-2-1651452" ] }
{ "_id" : ObjectId("625d78059329139694f188a4"), "CustID" : 3, "Name" : "Vaiubhak Bhat", "Type" : "Savings", "Contact" : [ "8971450321", "088-31-129458" ] }
{ "_id" : ObjectId("625d78059329139694f188a5"), "CustID" : 4, "Name" : "Pranod P Parande", "Type" : "Current", "Contact" : [ "9745230589", "088-56324587" ] }
{ "_id" : ObjectId("625d78059329139694f188a6"), "CustID" : 4, "Name" : "Shreyas R S", "Type" : "Current", "Contact" : [ "9445678321", "044-656-11729" ] }
{ "_id" : "625d78059329139694f188a6", "CustID" : 5, "Contact" : [ "9856321478", "011-65807458" ], "Name" : "Sumantha K S", "Type" : "Savings"
}

```

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

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 15000.