VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

BIG DATA ANALYTICS
(20CS6PEBDA)

Submitted by

Kizhakel Sharat Prasad(1BM19CS074)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
May-2022 to July-2022

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**BIG DATA ANALYTICS**" carried out by **Kizhakel Sharat Prasad(1BM19CS074),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **BIG DATA ANALYTICS - (20CS6PEBDA)** work prescribed for the said degree.

ANTARA ROY CHOUDHURY
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

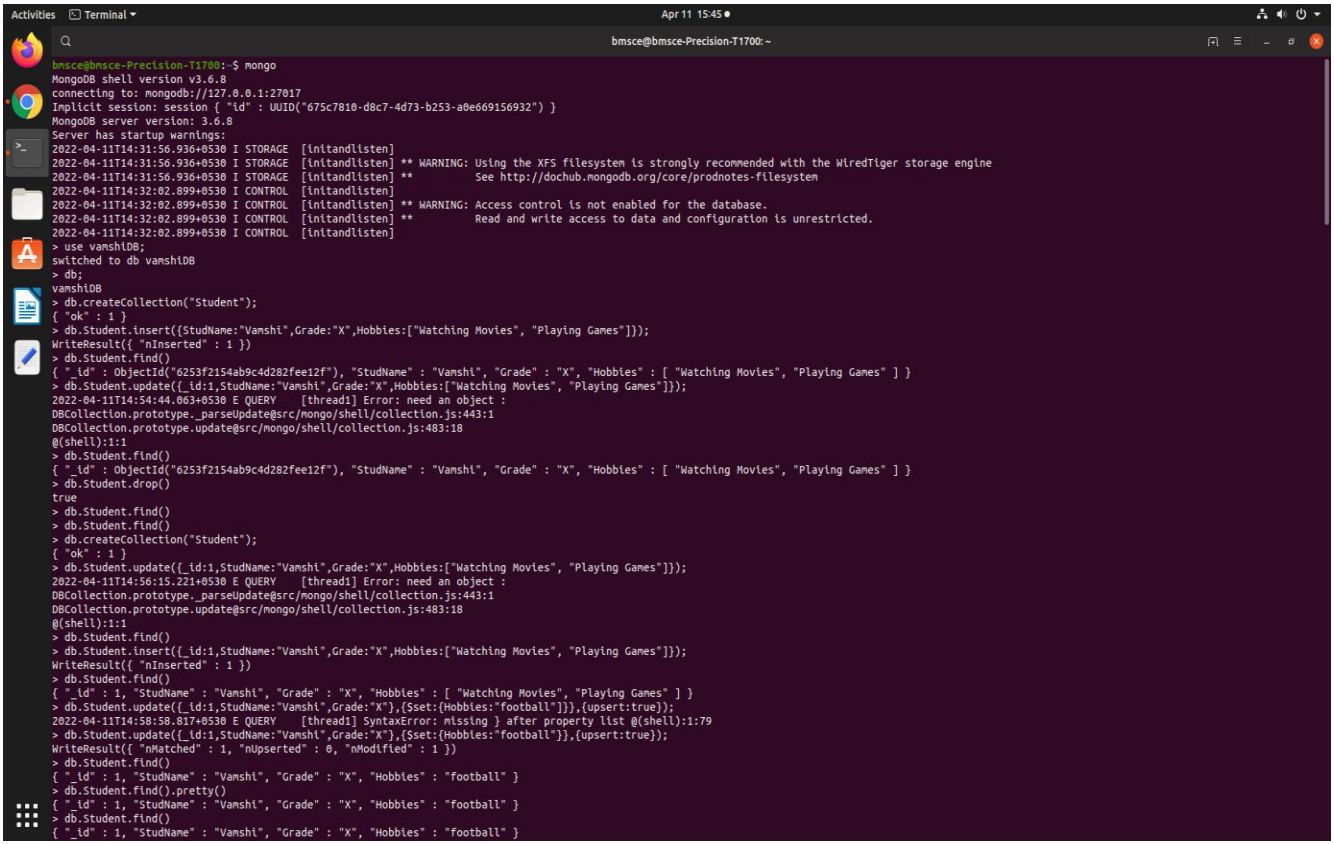
`

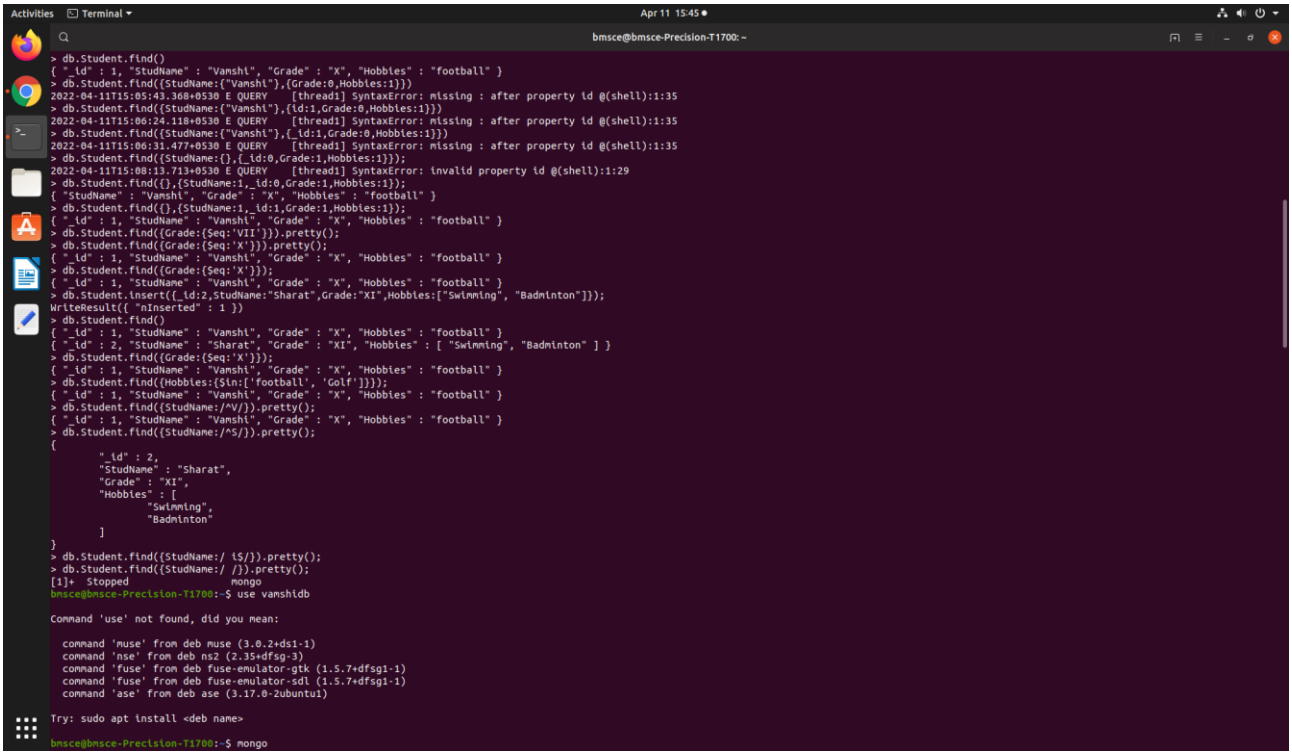
Index Sheet

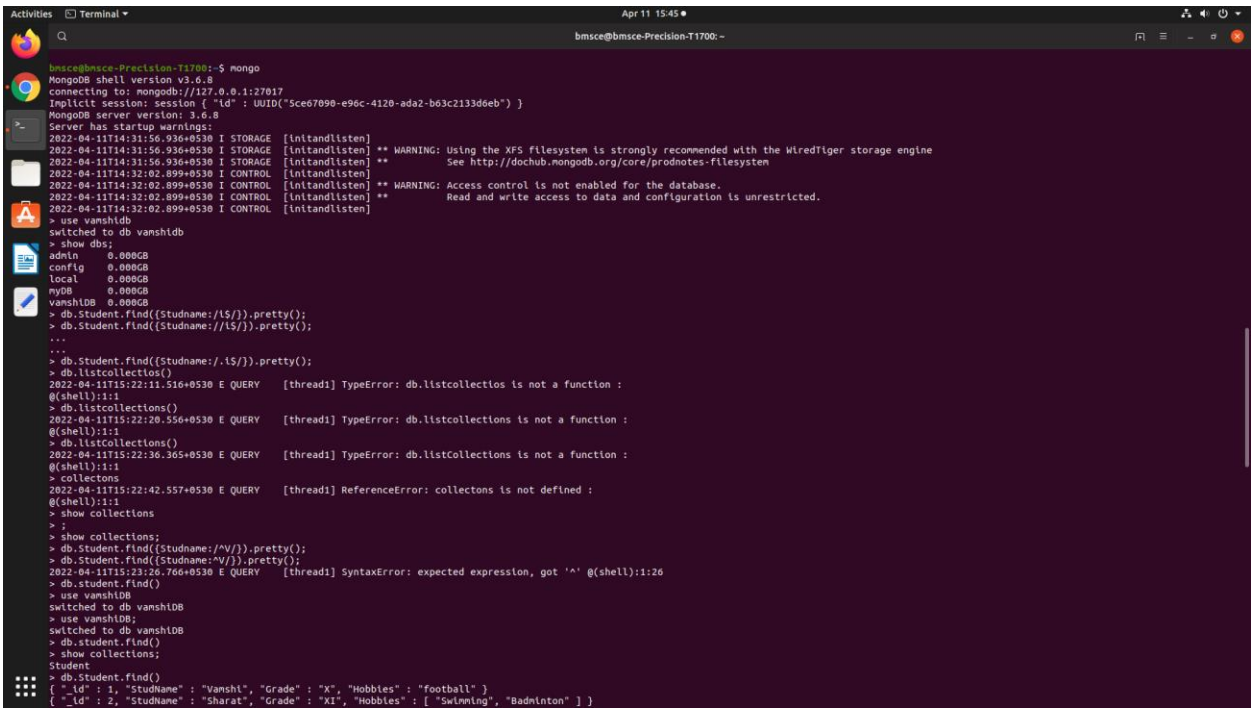
Course Outcome

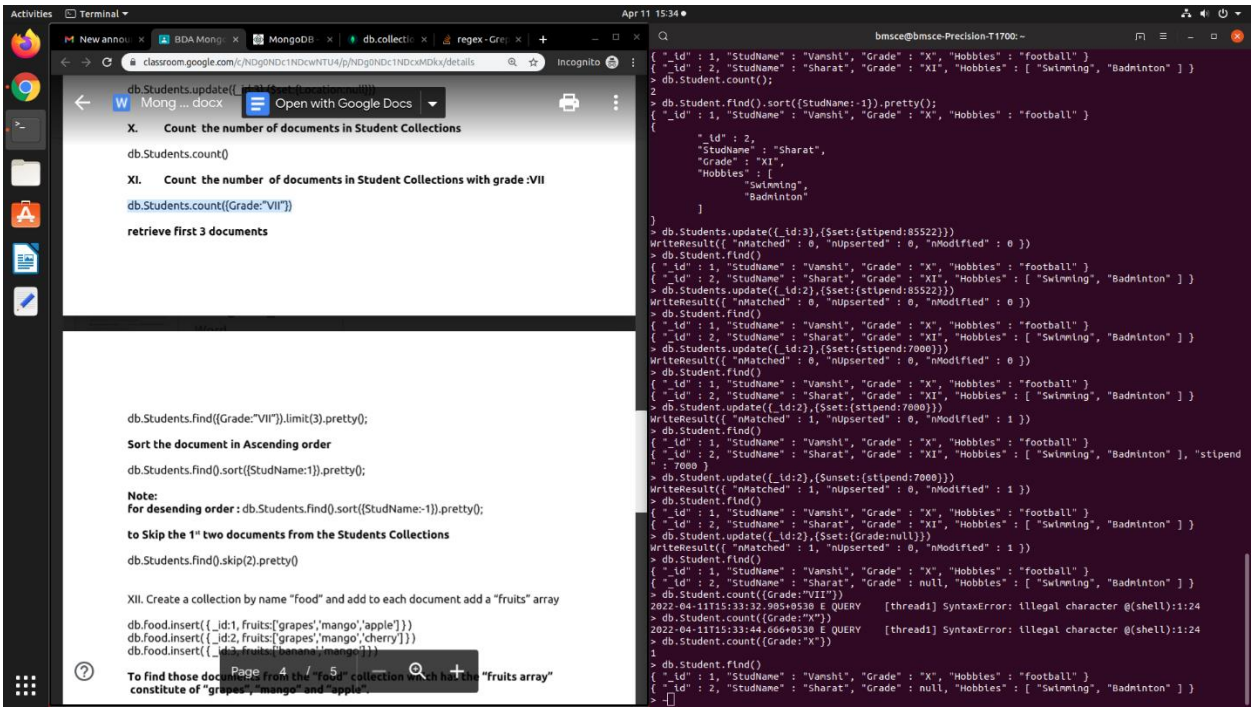
CO1	Apply the concept of NoSQL, Hadoop or Spark for a given task
CO2	Analyze the Big Data and obtain insight using data analytics mechanisms.
CO3	Design and implement Big data applications by applying NoSQL, Hadoop or Spark

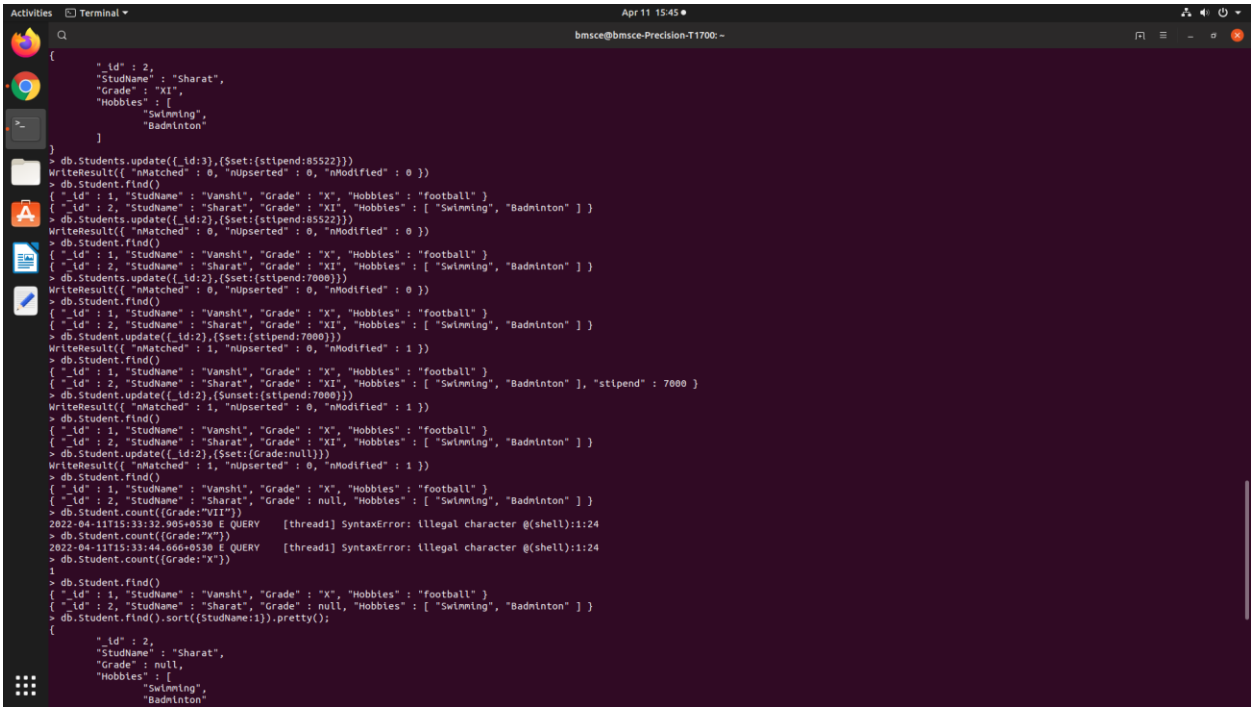
1. MongoDB- CRUD Demonstration

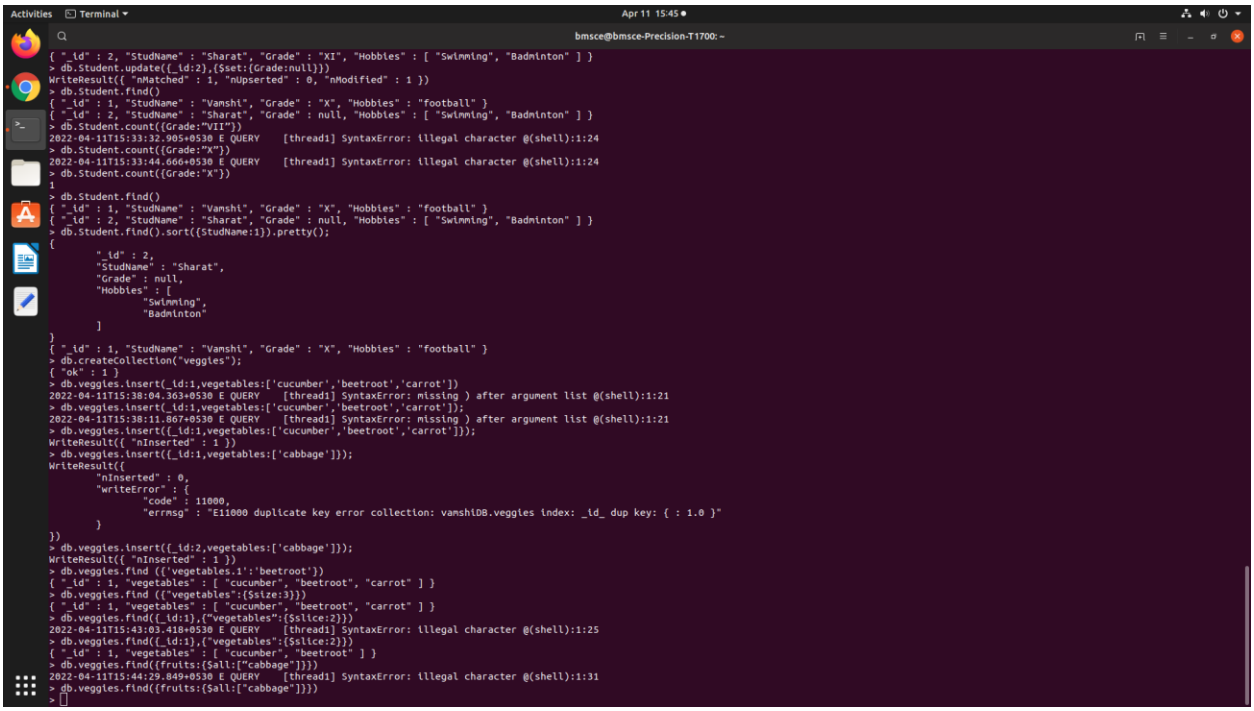



Activities
Terminal
Apr 11 15:45
bmsce@bmsce-Precision-T1700: ~
> db.Student.find()
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({StudName:{"Vamshi"},{Grade:0,Hobbies:1}})
2022-04-11T15:05:43.368+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:35
> db.Student.find({StudName:{"Vamshi"},{id:1,Grade:0,Hobbies:1}})
2022-04-11T15:06:24.118+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:35
> db.Student.find({StudName:{"Vamshi"},{_id:1,Grade:0,Hobbies:1}})
2022-04-11T15:06:31.477+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:35
> db.Student.find({StudName:{},{_id:0,Grade:1,Hobbies:1}});
2022-04-11T15:08:13.713+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:29
> db.Student.find({},{StudName:1,_id:0,Grade:1,Hobbies:1});
{ "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({},{StudName:1,_id:1,Grade:1,Hobbies:1});
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({Grade:{$eq:'VII'}}).pretty();
> db.Student.find({Grade:{$eq:'X'}}).pretty();
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({Grade:{$eq:'X'}});
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.insert({_id:2,StudName:"Sharat",Grade:"XI",Hobbies:["Swimming", "Badminton"]});
WriteResult({ "nInserted" : 1 })
> db.Student.find()
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
{ "_id" : 2, "StudName" : "Sharat", "Grade" : "XI", "Hobbies" : ["Swimming", "Badminton"] }
> db.Student.find({Grade:{$eq:'X'}});
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({Hobbies:{$in:['football', 'Golf']}});
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({StudName:/^V/}).pretty();
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.Student.find({StudName:/^S/}).pretty();
{
"_id" : 2,
"StudName" : "Sharat",
"Grade" : "XI",
"Hobbies" : [
"Swimming",
"Badminton"
]
}
> db.Student.find({StudName:/ i$/}).pretty();
> db.Student.find({StudName:/ /}).pretty();
[1]+ Stopped mongo
bmsce@bmsce-Precision-T1700:~$ use vamshidb
Command 'use' not found, did you mean:
command 'muse' from deb muse (3.0.2+ds1-1)
command 'nse' from deb ns2 (2.35+dfsg-3)
command 'fuse' from deb fuse-emulator-gtk (1.5.7+dfsg1-1)
command 'fuse' from deb fuse-emulator-sdl (1.5.7+dfsg1-1)
command 'ase' from deb ase (3.17.0-2ubuntu1)
Try: sudo apt install <deb name>
bmsce@bmsce-Precision-T1700:~$ mongo

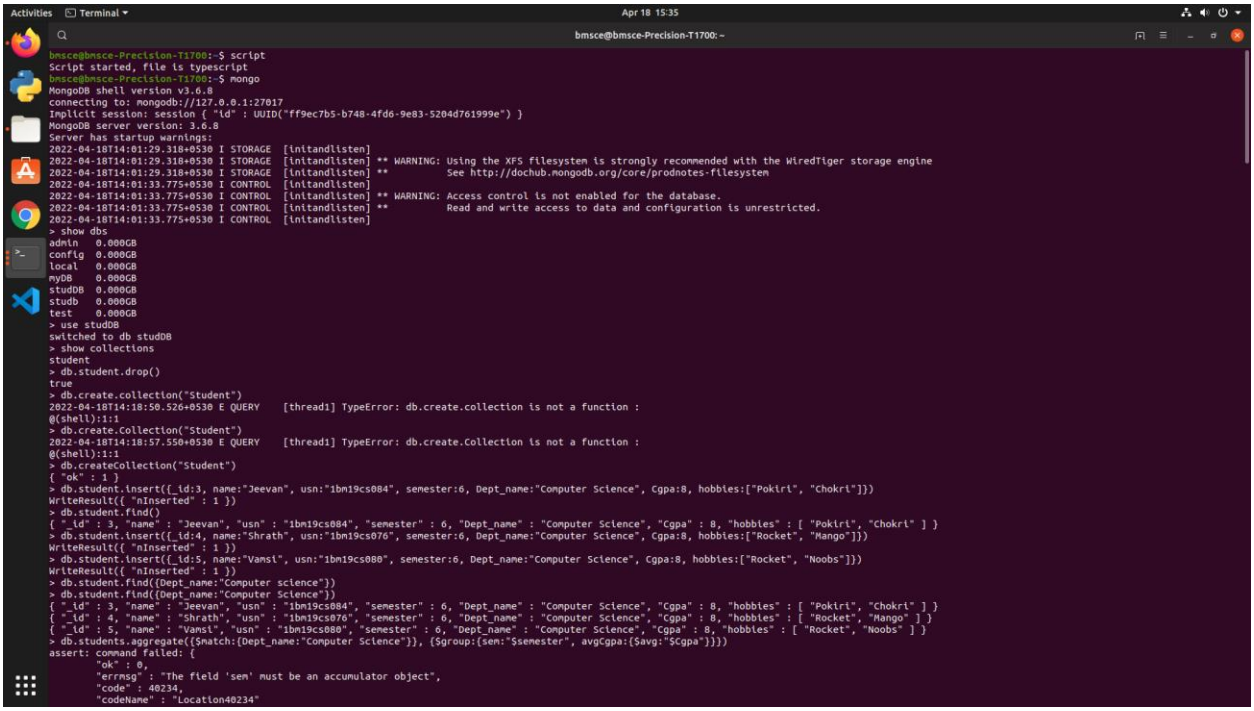

Activities
Terminal
Apr 11 15:45
bmsce@bmsce-Precision-T1700: ~
bmsce@bmsce-Precision-T1700:~$ mongo
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("5ce67090-e96c-4120-ada2-b63c2133d6eb") }
MongoDB server version: 3.6.8
Server has startup warnings:
2022-04-11T14:31:56.936+0530 I STORAGE [initandlisten]
2022-04-11T14:31:56.936+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2022-04-11T14:31:56.936+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
2022-04-11T14:32:02.899+0530 I CONTROL [initandlisten]
2022-04-11T14:32:02.899+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2022-04-11T14:32:02.899+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2022-04-11T14:32:02.899+0530 I CONTROL [initandlisten]
> use vamshidb
switched to db vamshidb
> show dbs;
admin 0.000GB
config 0.000GB
local 0.000GB
myDB 0.000GB
vamshiDB 0.000GB
> db.Student.find({Studname:/i$/}).pretty();
> db.Student.find({Studname://i$/}).pretty();
...
...
> db.Student.find({Studname:/.i$/}).pretty();
> db.listcollectios()
2022-04-11T15:22:11.516+0530 E QUERY [thread1] TypeError: db.listcollectios is not a function :
@(shell):1:1
> db.listcollections()
2022-04-11T15:22:20.556+0530 E QUERY [thread1] TypeError: db.listcollections is not a function :
@(shell):1:1
> db.listCollections()
2022-04-11T15:22:36.365+0530 E QUERY [thread1] TypeError: db.listCollections is not a function :
@(shell):1:1
> collectons
2022-04-11T15:22:42.557+0530 E QUERY [thread1] ReferenceError: collectons is not defined :
@(shell):1:1
> show collections
> ;
> show collections;
> db.Student.find({Studname:/^V/}).pretty();
> db.Student.find({Studname:^V/}).pretty();
2022-04-11T15:23:26.766+0530 E QUERY [thread1] SyntaxError: expected expression, got '^' @(shell):1:26
> db.student.find()
> use vamshiDB
switched to db vamshiDB
> use vamshiDB;
switched to db vamshiDB
> db.student.find()
> show collections;
Student
> db.Student.find()
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
{ "_id" : 2, "StudName" : "Sharat", "Grade" : "XI", "Hobbies" : ["Swimming", "Badminton"] }


Activities
Terminal
Apr 11 15:34
bmsce@bmsce-Precision-T1700: ~
Mong ... docx
Open with Google Docs
X. Count the number of documents in Student Collections
db.Students.count()
XI. Count the number of documents in Student Collections with grade :VII
db.Students.count({Grade:"VII"})
retrieve first 3 documents
db.Students.find({Grade:"VII"}).limit(3).pretty();
Sort the document in Ascending order
db.Students.find().sort({StudName:1}).pretty();
Note:
for desending order : db.Students.find().sort({StudName:-1}).pretty();
to Skip the 1st two documents from the Students Collections
db.Students.find().skip(2).pretty()
XII. Create a collection by name "food" and add to each document add a "fruits" array
db.food.insert({_id:1, fruits:['grapes','mango','apple']})
db.food.insert({_id:2, fruits:['grapes','mango','cherry']})
db.food.insert({_id:3, fruits:['banana','mango']})
Page 4 / 5
To find those documents from the "food" collection which has the "fruits array"
constitute of "grapes", "mango" and "apple".

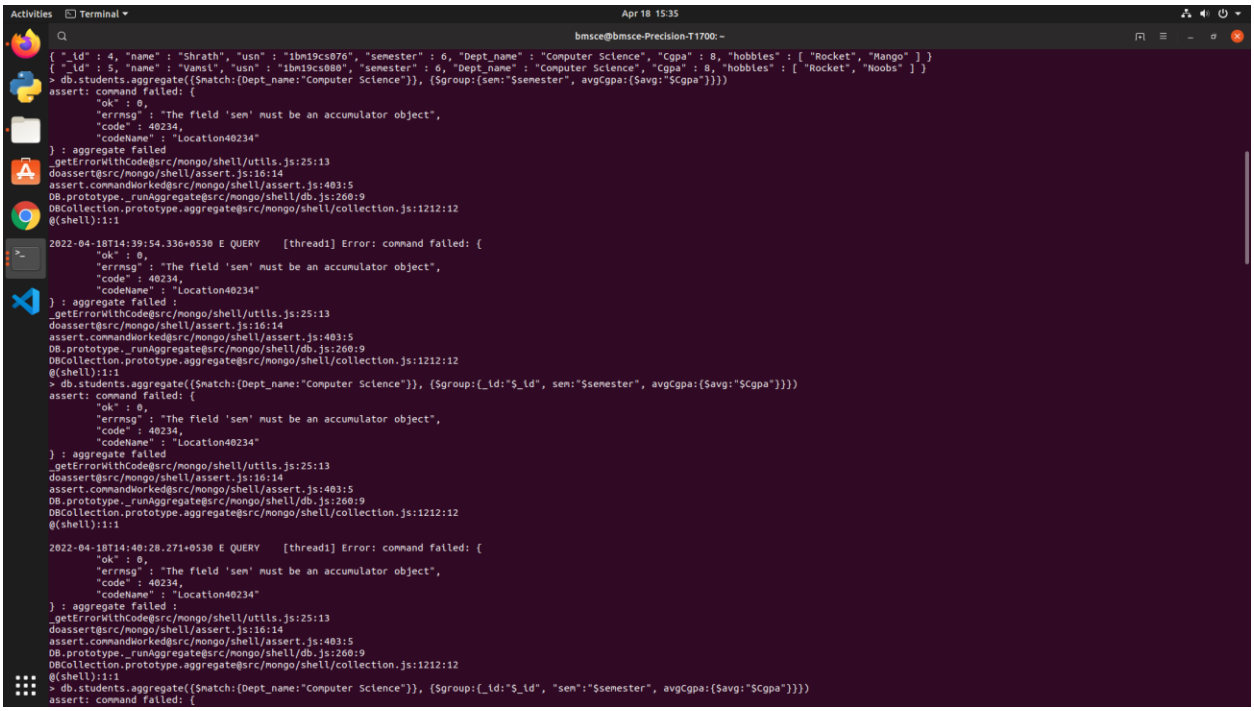

Activities
Terminal
Apr 11 15:45
bmsce@bmsce-Precision-T1700: ~

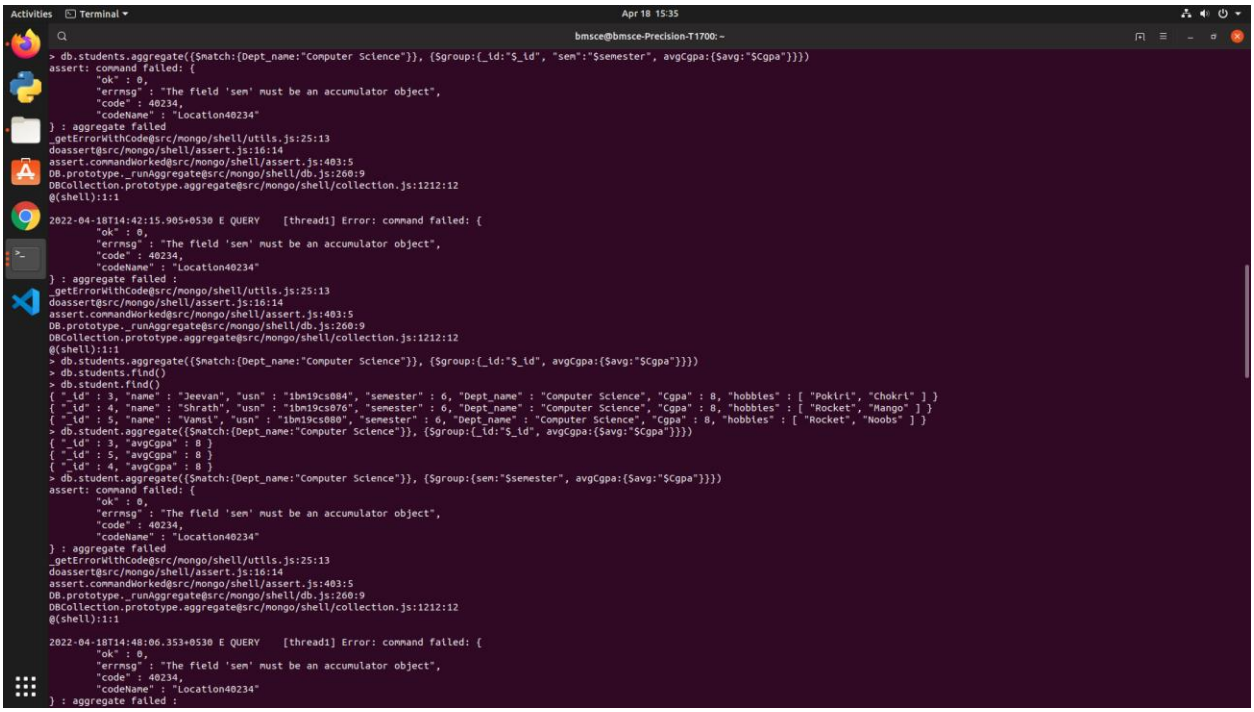


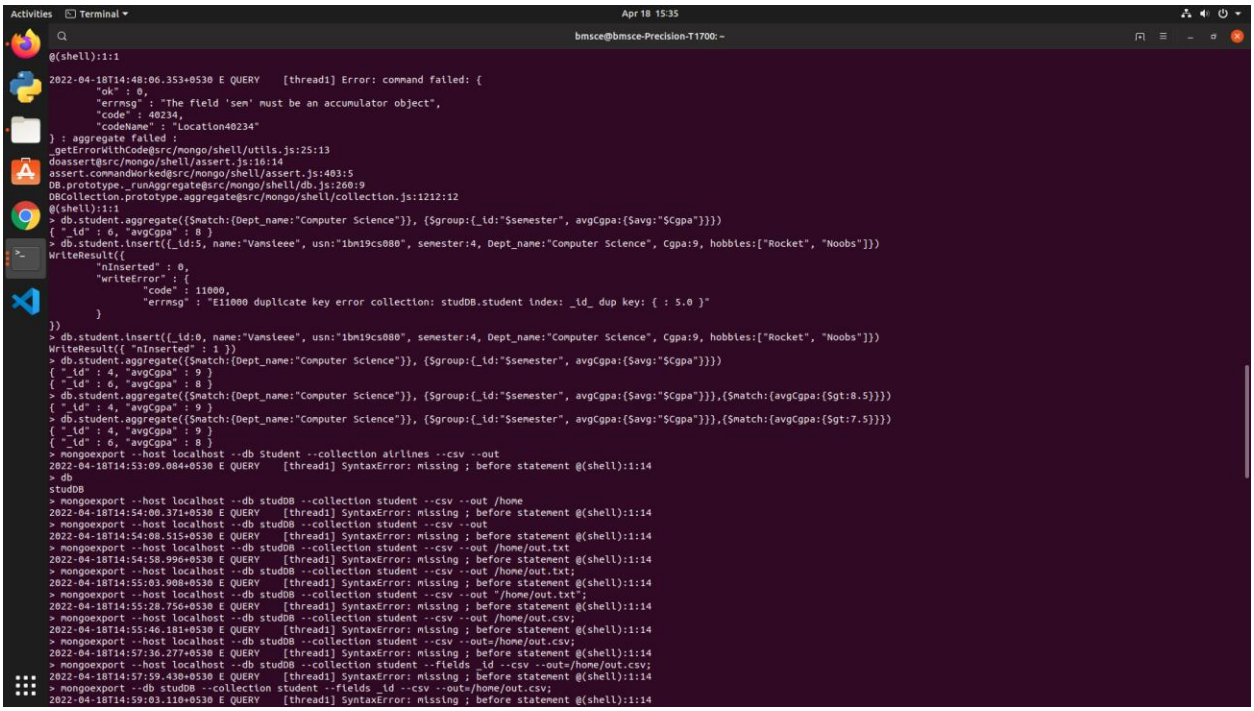

```
{ "_id" : 2, "StudName" : "Sharat", "Grade" : "XI", "Hobbies" : [ "Swimming", "Badminton" ] }
> db.Student.update({_id:2},{$set:{Grade:null}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find()
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
{ "_id" : 2, "StudName" : "Sharat", "Grade" : null, "Hobbies" : [ "Swimming", "Badminton" ] }
> db.Student.count({Grade:"VII"})
2022-04-11T15:33:32.905+0530 E QUERY    [thread1] SyntaxError: illegal character @(shell):1:24
> db.Student.count({Grade:"X"})
2022-04-11T15:33:44.606+0530 E QUERY    [thread1] SyntaxError: illegal character @(shell):1:24
> db.Student.count({Grade:"X"})
1
> db.Student.find()
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
{ "_id" : 2, "StudName" : "Sharat", "Grade" : null, "Hobbies" : [ "Swimming", "Badminton" ] }
> db.Student.find().sort({StudName:1}).pretty();
{
        "_id" : 2,
        "StudName" : "Sharat",
        "Grade" : null,
        "Hobbies" : [
                "Swimming",
                "Badminton"
        ]
}
{ "_id" : 1, "StudName" : "Vamshi", "Grade" : "X", "Hobbies" : "football" }
> db.createCollection("veggies");
{ "ok" : 1 }
> db.veggies.insert(_id:1,vegetables:['cucumber','beetroot','carrot'])
2022-04-11T15:38:04.363+0530 E QUERY    [thread1] SyntaxError: missing ) after argument list @(shell):1:21
> db.veggies.insert(_id:1,vegetables:['cucumber','beetroot','carrot']);
2022-04-11T15:38:11.867+0530 E QUERY    [thread1] SyntaxError: missing ) after argument list @(shell):1:21
> db.veggies.insert({_id:1,vegetables:['cucumber','beetroot','carrot']});
WriteResult({ "nInserted" : 1 })
> db.veggies.insert({_id:1,vegetables:['cabbage']});
WriteResult({
        "nInserted" : 0,
        "writeError" : {
                "code" : 11000,
                "errmsg" : "E11000 duplicate key error collection: vamshiDB.veggies index: _id_ dup key: { : 1.0 }"
        }
})
> db.veggies.insert({_id:2,vegetables:['cabbage']});
WriteResult({ "nInserted" : 1 })
> db.veggies.find ({'vegetables.1':'beetroot'})
{ "_id" : 1, "vegetables" : [ "cucumber", "beetroot", "carrot" ] }
> db.veggies.find ({"vegetables":{$size:3}})
{ "_id" : 1, "vegetables" : [ "cucumber", "beetroot", "carrot" ] }
> db.veggies.find({_id:1},{"vegetables":{$slice:2}})
2022-04-11T15:43:03.418+0530 E QUERY    [thread1] SyntaxError: illegal character @(shell):1:25
> db.veggies.find({_id:1},{"vegetables":{$slice:2}})
{ "_id" : 1, "vegetables" : [ "cucumber", "beetroot" ] }
> db.veggies.find({fruits:{$all:["cabbage"]}})
2022-04-11T15:44:29.849+0530 E QUERY    [thread1] SyntaxError: illegal character @(shell):1:31
> db.veggies.find({fruits:{$all:["cabbage"]}})
>
```

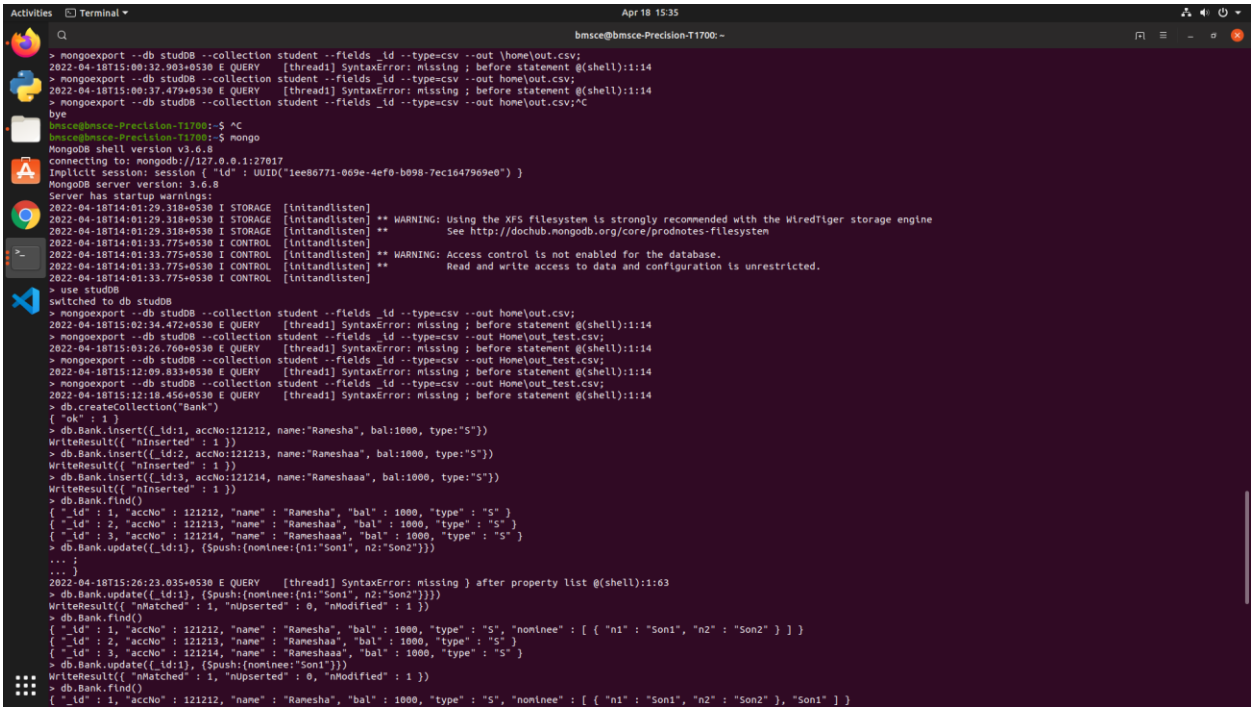


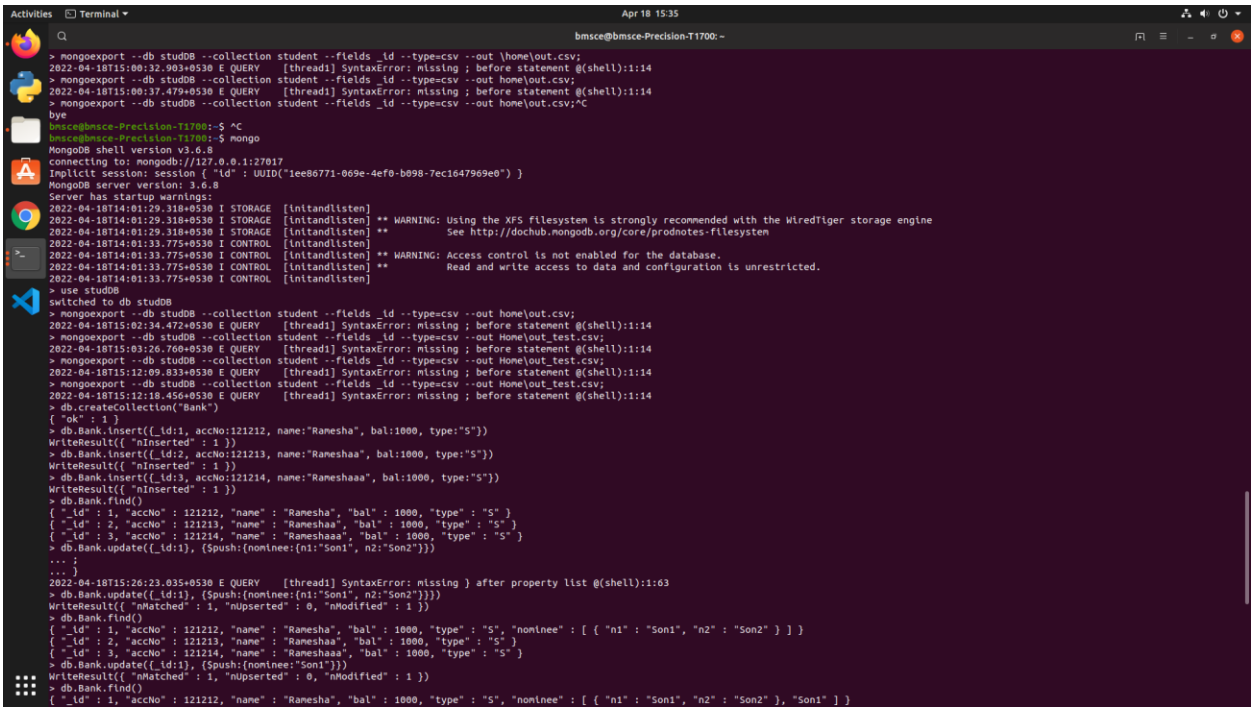

```
bmsce@bmsce-Precision-T1700:~$ script
Script started, file is typescript
bmsce@bmsce-Precision-T1700:~$ mongo
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("ff9ec7b5-b748-4fd6-9e83-5204d761999e") }
MongoDB server version: 3.6.8
Server has startup warnings:
2022-04-18T14:01:29.318+0530 I STORAGE  [initandlisten]
2022-04-18T14:01:29.318+0530 I STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2022-04-18T14:01:29.318+0530 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/prodnotes-filesystem
2022-04-18T14:01:33.775+0530 I CONTROL  [initandlisten]
2022-04-18T14:01:33.775+0530 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2022-04-18T14:01:33.775+0530 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2022-04-18T14:01:33.775+0530 I CONTROL  [initandlisten]
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
myDB    0.000GB
studDB  0.000GB
studb   0.000GB
test    0.000GB
> use studDB
switched to db studDB
> show collections
student
> db.student.drop()
true
> db.create.collection("Student")
2022-04-18T14:18:50.526+0530 E QUERY    [thread1] TypeError: db.create.collection is not a function :
@(shell):1:1
> db.create.Collection("Student")
2022-04-18T14:18:57.550+0530 E QUERY    [thread1] TypeError: db.create.Collection is not a function :
@(shell):1:1
> db.createCollection("Student")
{ "ok" : 1 }
> db.student.insert({_id:3, name:"Jeevan", usn:"1bm19cs084", semester:6, Dept_name:"Computer Science", Cgpa:8, hobbies:["Pokiri", "Chokri"]})
WriteResult({ "nInserted" : 1 })
> db.student.find()
{ "_id" : 3, "name" : "Jeevan", "usn" : "1bm19cs084", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : [ "Pokiri", "Chokri" ] }
> db.student.insert({_id:4, name:"Shrath", usn:"1bm19cs076", semester:6, Dept_name:"Computer Science", Cgpa:8, hobbies:["Rocket", "Mango"]})
WriteResult({ "nInserted" : 1 })
> db.student.insert({_id:5, name:"Vamsi", usn:"1bm19cs080", semester:6, Dept_name:"Computer Science", Cgpa:8, hobbies:["Rocket", "Noobs"]})
WriteResult({ "nInserted" : 1 })
> db.student.find({Dept_name:"Computer science"})
> db.student.find({Dept_name:"Computer Science"})
{ "_id" : 3, "name" : "Jeevan", "usn" : "1bm19cs084", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : [ "Pokiri", "Chokri" ] }
{ "_id" : 4, "name" : "Shrath", "usn" : "1bm19cs076", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : [ "Rocket", "Mango" ] }
{ "_id" : 5, "name" : "Vamsi", "usn" : "1bm19cs080", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : [ "Rocket", "Noobs" ] }
> db.students.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{sem:"$semester", avgCgpa:{$avg:"$Cgpa"}}})
assert: command failed: {
        "ok" : 0,
        "errmsg" : "The field 'sem' must be an accumulator object",
        "code" : 40234,
        "codeName" : "Location40234"
```

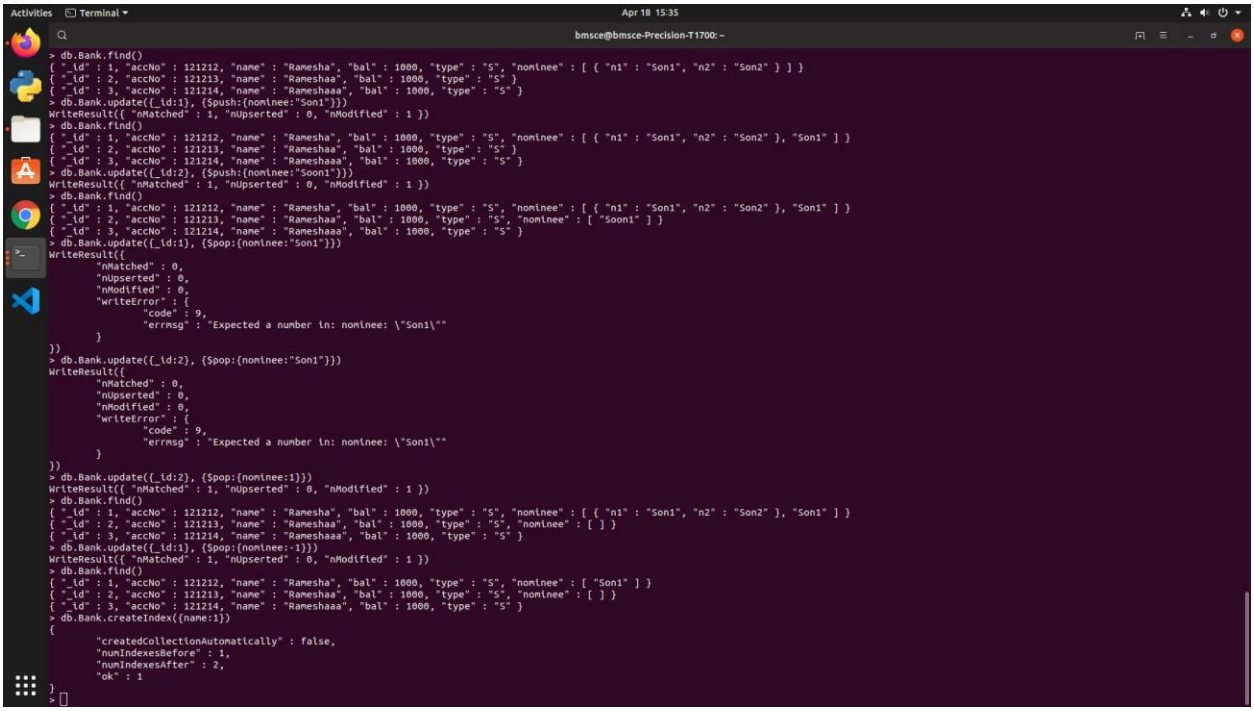

Activities
Terminal
Apr 18 15:35
bmsce@bmsce-Precision-T1700: ~
{ "_id" : 4, "name" : "Shrath", "usn" : "1bm19cs076", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : ["Rocket", "Mango"] }
{ "_id" : 5, "name" : "Vansi", "usn" : "1bm19cs080", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : ["Rocket", "Noobs"] }
> db.students.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{sem:"$semester", avgCgpa:{$avg:"$Cgpa"}}})
assert: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
2022-04-18T14:39:54.336+0530 E QUERY [thread1] Error: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
> db.students.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$_id", sem:"$semester", avgCgpa:{$avg:"$Cgpa"}}})
assert: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
2022-04-18T14:40:28.271+0530 E QUERY [thread1] Error: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
> db.students.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$_id", "sem":"$semester", avgCgpa:{$avg:"$Cgpa"}}})
assert: command failed: {


Activities
Terminal
Apr 18 15:35
bmsce@bmsce-Precision-T1700: ~
> db.students.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$_id", "sem":"$semester", avgCgpa:{$avg:"$Cgpa"}}})
assert: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
2022-04-18T14:42:15.905+0530 E QUERY [thread1] Error: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
> db.students.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$_id", avgCgpa:{$avg:"$Cgpa"}}})
> db.students.find()
> db.student.find()
{ "_id" : 3, "name" : "Jeevan", "usn" : "1bm19cs084", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : ["Pokirl", "Chokrl"] }
{ "_id" : 4, "name" : "Shrath", "usn" : "1bm19cs076", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : ["Rocket", "Mango"] }
{ "_id" : 5, "name" : "Vansi", "usn" : "1bm19cs080", "semester" : 6, "Dept_name" : "Computer Science", "Cgpa" : 8, "hobbies" : ["Rocket", "Noobs"] }
> db.student.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$_id", avgCgpa:{$avg:"$Cgpa"}}})
{ "_id" : 3, "avgCgpa" : 8 }
{ "_id" : 5, "avgCgpa" : 8 }
{ "_id" : 4, "avgCgpa" : 8 }
> db.student.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{sem:"$semester", avgCgpa:{$avg:"$Cgpa"}}})
assert: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
2022-04-18T14:48:06.353+0530 E QUERY [thread1] Error: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed :


Activities
Terminal
Apr 18 15:35
bmsce@bmsce-Precision-T1700: ~
@(shell):1:1
2022-04-18T14:48:06.353+0530 E QUERY [thread1] Error: command failed: {
"ok" : 0,
"errmsg" : "The field 'sem' must be an accumulator object",
"code" : 40234,
"codeName" : "Location40234"
} : aggregate failed :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:16:14
assert.commandWorked@src/mongo/shell/assert.js:403:5
DB.prototype._runAggregate@src/mongo/shell/db.js:260:9
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1212:12
@(shell):1:1
> db.student.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$semester", avgCgpa:{$avg:"$Cgpa"}}})
{ "_id" : 6, "avgCgpa" : 8 }
> db.student.insert({_id:5, name:"Vamsleee", usn:"1bm19cs080", semester:4, Dept_name:"Computer Science", Cgpa:9, hobbies:["Rocket", "Noobs"]})
WriteResult({
"nInserted" : 0,
"writeError" : {
"code" : 11000,
"errmsg" : "E11000 duplicate key error collection: studDB.student index: _id_ dup key: { : 5.0 }"
}
})
> db.student.insert({_id:0, name:"Vamsleee", usn:"1bm19cs080", semester:4, Dept_name:"Computer Science", Cgpa:9, hobbies:["Rocket", "Noobs"]})
WriteResult({ "nInserted" : 1 })
> db.student.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$semester", avgCgpa:{$avg:"$Cgpa"}}})
{ "_id" : 4, "avgCgpa" : 9 }
{ "_id" : 6, "avgCgpa" : 8 }
> db.student.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$semester", avgCgpa:{$avg:"$Cgpa"}}},{$match:{avgCgpa:{$gt:8.5}}})
{ "_id" : 4, "avgCgpa" : 9 }
> db.student.aggregate({$match:{Dept_name:"Computer Science"}}, {$group:{_id:"$semester", avgCgpa:{$avg:"$Cgpa"}}},{$match:{avgCgpa:{$gt:7.5}}})
{ "_id" : 4, "avgCgpa" : 9 }
{ "_id" : 6, "avgCgpa" : 8 }
> mongoexport --host localhost --db Student --collection airlines --csv --out
2022-04-18T14:53:09.084+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> db
studDB
> mongoexport --host localhost --db studDB --collection student --csv --out /home
2022-04-18T14:54:00.371+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --csv --out
2022-04-18T14:54:08.515+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --csv --out /home/out.txt
2022-04-18T14:54:58.996+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --csv --out /home/out.txt;
2022-04-18T14:55:03.908+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --csv --out "/home/out.txt";
2022-04-18T14:55:28.756+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --csv --out /home/out.csv;
2022-04-18T14:55:46.181+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --csv --out=/home/out.csv;
2022-04-18T14:57:36.277+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --host localhost --db studDB --collection student --fields _id --csv --out=/home/out.csv;
2022-04-18T14:57:59.430+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --csv --out=/home/out.csv;
2022-04-18T14:59:03.110+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14


Activities
Terminal
Apr 18 15:35
bmsce@bmsce-Precision-T1700: ~
> mongoexport --db studDB --collection student --fields _id --type=csv --out \home\out.csv;
2022-04-18T15:00:32.903+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out home\out.csv;
2022-04-18T15:00:37.479+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out home\out.csv;^C
bye
bmsce@bmsce-Precision-T1700:~$ ^C
bmsce@bmsce-Precision-T1700:~$ mongo
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("1ee86771-069e-4ef0-b098-7ec1647969e0") }
MongoDB server version: 3.6.8
Server has startup warnings:
2022-04-18T14:01:29.318+0530 I STORAGE [initandlisten]
2022-04-18T14:01:29.318+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2022-04-18T14:01:29.318+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten]
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten]
> use studDB
switched to db studDB
> mongoexport --db studDB --collection student --fields _id --type=csv --out home\out.csv;
2022-04-18T15:02:34.472+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out Home\out_test.csv;
2022-04-18T15:03:26.760+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out Home\out_test.csv;
2022-04-18T15:12:09.833+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out Home\out_test.csv;
2022-04-18T15:12:18.456+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> db.createCollection("Bank")
{ "ok" : 1 }
> db.Bank.insert({_id:1, accNo:121212, name:"Ramesha", bal:1000, type:"S"})
WriteResult({ "nInserted" : 1 })
> db.Bank.insert({_id:2, accNo:121213, name:"Rameshaa", bal:1000, type:"S"})
WriteResult({ "nInserted" : 1 })
> db.Bank.insert({_id:3, accNo:121214, name:"Rameshaaa", bal:1000, type:"S"})
WriteResult({ "nInserted" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S" }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S" }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$push:{nominee:{n1:"Son1", n2:"Son2"}}})
... ;
...)
2022-04-18T15:26:23.035+0530 E QUERY [thread1] SyntaxError: missing } after property list @(shell):1:63
> db.Bank.update({_id:1}, {$push:{nominee:{n1:"Son1", n2:"Son2"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S" }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$push:{nominee:"Son1"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }, "Son1"] }


Activities
Terminal
Apr 18 15:35
bmsce@bmsce-Precision-T1700: ~
> mongoexport --db studDB --collection student --fields _id --type=csv --out \home\out.csv;
2022-04-18T15:00:32.903+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out home\out.csv;
2022-04-18T15:00:37.479+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out home\out.csv;^C
bye
bmsce@bmsce-Precision-T1700:~$ ^C
bmsce@bmsce-Precision-T1700:~$ mongo
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("1ee86771-069e-4ef0-b098-7ec1647969e0") }
MongoDB server version: 3.6.8
Server has startup warnings:
2022-04-18T14:01:29.318+0530 I STORAGE [initandlisten]
2022-04-18T14:01:29.318+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2022-04-18T14:01:29.318+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten]
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2022-04-18T14:01:33.775+0530 I CONTROL [initandlisten]
> use studDB
switched to db studDB
> mongoexport --db studDB --collection student --fields _id --type=csv --out home\out.csv;
2022-04-18T15:02:34.472+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out Home\out_test.csv;
2022-04-18T15:03:26.760+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out Home\out_test.csv;
2022-04-18T15:12:09.833+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> mongoexport --db studDB --collection student --fields _id --type=csv --out Home\out_test.csv;
2022-04-18T15:12:18.456+0530 E QUERY [thread1] SyntaxError: missing ; before statement @(shell):1:14
> db.createCollection("Bank")
{ "ok" : 1 }
> db.Bank.insert({_id:1, accNo:121212, name:"Ramesha", bal:1000, type:"S"})
WriteResult({ "nInserted" : 1 })
> db.Bank.insert({_id:2, accNo:121213, name:"Rameshaa", bal:1000, type:"S"})
WriteResult({ "nInserted" : 1 })
> db.Bank.insert({_id:3, accNo:121214, name:"Rameshaaa", bal:1000, type:"S"})
WriteResult({ "nInserted" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S" }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S" }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$push:{nominee:{n1:"Son1", n2:"Son2"}}})
... ;
... }
2022-04-18T15:26:23.035+0530 E QUERY [thread1] SyntaxError: missing } after property list @(shell):1:63
> db.Bank.update({_id:1}, {$push:{nominee:{n1:"Son1", n2:"Son2"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S" }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$push:{nominee:"Son1"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }, "Son1"] }


Activities
Terminal
Apr 18 15:35
bmsce@bmsce-Precision-T1700: ~
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S" }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$push:{nominee:"Son1"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }, "Son1"] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S" }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:2}, {$push:{nominee:"Soon1"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }, "Son1"] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S", "nominee" : ["Soon1"] }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$pop:{nominee:"Son1"}})
WriteResult({
"nMatched" : 0,
"nUpserted" : 0,
"nModified" : 0,
"writeError" : {
"code" : 9,
"errmsg" : "Expected a number in: nominee: \"Son1\""
}
})
> db.Bank.update({_id:2}, {$pop:{nominee:"Son1"}})
WriteResult({
"nMatched" : 0,
"nUpserted" : 0,
"nModified" : 0,
"writeError" : {
"code" : 9,
"errmsg" : "Expected a number in: nominee: \"Son1\""
}
})
> db.Bank.update({_id:2}, {$pop:{nominee:1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : [{ "n1" : "Son1", "n2" : "Son2" }, "Son1"] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S", "nominee" : [ ] }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.update({_id:1}, {$pop:{nominee:-1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Bank.find()
{ "_id" : 1, "accNo" : 121212, "name" : "Ramesha", "bal" : 1000, "type" : "S", "nominee" : ["Son1"] }
{ "_id" : 2, "accNo" : 121213, "name" : "Rameshaa", "bal" : 1000, "type" : "S", "nominee" : [ ] }
{ "_id" : 3, "accNo" : 121214, "name" : "Rameshaaa", "bal" : 1000, "type" : "S" }
> db.Bank.createIndex({name:1})
{
"createdCollectionAutomatically" : false,
"numIndexesBefore" : 1,
"numIndexesAfter" : 2,
"ok" : 1
}

2. Perform the following DB operations using Cassandra.

1.Create a keyspace by name Employee
2. Create a column family by name Employee-Info
with attributes
Emp_Id Primary Key, Emp_Name,
Designation, Date_of_Joining, Salary,
Dept_Name
3. Insert the values into the table in batch
4. Update Employee name and Department of Emp-Id 121
5. Sort the details of Employee records based on salary
6. Alter the schema of the table Employee_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.
7. Update the altered table to add project names.
8.Create a TTL of 15 seconds to display the values of Employee

NAME: KIZHAKEL SHARAT PRASAD

USN:1BM19CS074

BDA LAB WEEK3 CASSANDRA ASSIGNMENT WITH SCREENSHOTS:

1. Create a key space by name Employee

```
cqlsh> describe keyspaces;

system_schema  system        system_distributed
system_auth    test_keyspace  system_traces

cqlsh> create keyspace Employee with replication={'class':'SimpleStrategy','replication_factor':2};
cqlsh> describe keyspace Employee;

CREATE KEYSPACE employee WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '2'}  AND durable_writes = true;

cqlsh> use employee;
```

2. Create a column family by name Employee-Info with attributes Emp_Id Primary Key, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name

```
cqlsh:employee> CREATE TABLE Employee_Info(cluster_col text,Emp_id int,Emp_Name text,Designation text,Date_of_Joining timestamp,Salary int,Dept_Name text,primary key(clust
r_col,Salary)) WITH CLUSTERING ORDER BY(Salary DESC);
cqlsh:employee> select*from employee;
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured table employee"
cqlsh:employee> select*from employee_info;

 cluster_col | salary | date_of_joining | dept_name | designation | emp_id | emp_name
-------------+--------+-----------------+-----------+-------------+--------+----------
```

3. Insert the values into the table in batch

```
(0 rows)
cqlsh:employee> BEGIN BATCH
           ... INSERT INTO Employee_Info(cluster_col,emp_id,emp_name,designation,Date_of_Joining,Salary,Dept_Name) VALUES ('xyz',1,'Ravi','MTS','2020-08-24',12000,'TESTING
');
           ... INSERT INTO Employee_Info(cluster_col,emp_id,emp_name,designation,Date_of_Joining,Salary,Dept_Name) VALUES ('xyz',2,'Vamshi','MANAGER','2021-03-20',50000,'D
EVELOPEMENT');
           ... INSERT INTO Employee_Info(cluster_col,emp_id,emp_name,designation,Date_of_Joining,Salary,Dept_Name) VALUES ('xyz',121,'Kiran','SDE','2019-04-21',10000,'PROD
UCTION');
           ... INSERT INTO Employee_Info(cluster_col,emp_id,emp_name,designation,Date_of_Joining,Salary,Dept_Name) VALUES ('xyz',3,'Ramesh','ANALYST','2020-05-07',20000,'Q
UALITY');
           ... APPLY BATCH;
cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name
-------------+--------+---------------------------------+--------------+-------------+--------+----------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |    Ravi
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |   PRODUCTION |         SDE |    121 |   Kiran

(4 rows)
```

4. Update Employee name and Department of Emp-Id 121

```
cqlsh:employee> update Employee_Info SET emp_name='karthik',dept_name='Compliance' where cluster_col='xyz' and salary=10000 IF emp_id=121;

 [applied]
-----------
      True

cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name | projects
-------------+--------+---------------------------------+--------------+-------------+--------+----------+----------------------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi |         {'AI', 'DS'}
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh |           {'DEVOPS'}
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |    Ravi |              {'ML'}
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |   Compliance |         SDE |    121 |  karthik | {'QUANTUM COMPUTING'}
```

5. Sort the details of Employee records based on salary

```
cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name | projects
-------------+--------+---------------------------------+--------------+-------------+--------+----------+----------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi |     null
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh |     null
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |    Ravi |     null
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |      Finance |         SDE |    121 |  Jignesh |     null
```

6. Alter the schema of the table Employee_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.

```
cqlsh:employee> alter table Employee_Info add Projects set<text>;
cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name | projects
-------------+--------+---------------------------------+--------------+-------------+--------+----------+----------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi |     null
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh |     null
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |    Ravi |     null
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |      Finance |         SDE |    121 |  Jignesh |     null

(4 rows)
```

7. Update the altered table to add project names.

```
cqlsh:employee> update Employee_Info SET projects=projects+{'ML'} where cluster_col='xyz' and salary=12000 IF emp_id=1;

 [applied]
-----------
      True

cqlsh:employee> update Employee_Info SET projects=projects+{'AI','DS'} where cluster_col='xyz' and salary=50000 IF emp_id=2;

 [applied]
-----------
      True

cqlsh:employee> update Employee_Info SET projects=projects+{'DEVOPS'} where cluster_col='xyz' and salary=20000 IF emp_id=3;

 [applied]
-----------
      True

cqlsh:employee> update Employee_Info SET projects=projects+{'QUANTUM COMPUTING'} where cluster_col='xyz' and salary=10000 IF emp_id=121;

 [applied]
-----------
      True

cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name | projects
-------------+--------+---------------------------------+--------------+-------------+--------+----------+------------------------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi |          {'AI', 'DS'}
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh |           {'DEVOPS'}
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |     Ravi |                {'ML'}
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |      Finance |         SDE |    121 |  Jignesh | {'QUANTUM COMPUTING'}
```

8. Create a TTL of 15 seconds to display the values of Employees.

```
(4 rows)
cqlsh:employee> INSERT INTO Employee_Info(cluster_col,emp_id,emp_name,designation,Date_of_Joining,Salary,Dept_Name) VALUES ('xyz',121,'Modi','SDE','2022-04-
TION') using TTL 15;
cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name | projects
-------------+--------+---------------------------------+--------------+-------------+--------+----------+------------------------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi |          {'AI', 'DS'}
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh |           {'DEVOPS'}
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |     Ravi |                {'ML'}
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |      Finance |         SDE |    121 |  Jignesh | {'QUANTUM COMPUTING'}
         xyz |   1000 | 2022-04-20 18:30:00.000000+0000 |   PRODUCTION |         SDE |    121 |    Modi |                   null

(5 rows)
cqlsh:employee> SELECT*FROM Employee_Info;

 cluster_col | salary | date_of_joining                 | dept_name    | designation | emp_id | emp_name | projects
-------------+--------+---------------------------------+--------------+-------------+--------+----------+------------------------
         xyz |  50000 | 2021-03-19 18:30:00.000000+0000 | DEVELOPEMENT |     MANAGER |      2 |   Vamshi |          {'AI', 'DS'}
         xyz |  20000 | 2020-05-06 18:30:00.000000+0000 |      QUALITY |     ANALYST |      3 |   Ramesh |           {'DEVOPS'}
         xyz |  12000 | 2020-08-23 18:30:00.000000+0000 |      TESTING |         MTS |      1 |     Ravi |                {'ML'}
         xyz |  10000 | 2019-04-20 18:30:00.000000+0000 |      Finance |         SDE |    121 |  Jignesh | {'QUANTUM COMPUTING'}

(4 rows)
```

3. Perform the following DB operations using Cassandra.

1.Create a keyspace by name Library
2. Create a column family by name Library-Info with attributes
   Stud_Id Primary Key,
   Counter_value of type Counter,
   Stud_Name, Book-Name, Book-Id,
   Date_of_issue
3. Insert the values into the table in batch
4. Display the details of the table created and increase the value of the counter
5. Write a query to show that a student with id 112 has taken a book “BDA” 2 times.
6. Export the created column to a csv file
7. Import a given csv dataset from local file system into Cassandra column family

NAME: KIZHAKEL SHARAT PRASAD

USN: 1BM19CS074

BDA LAB WEEK 4 CASSANDRA ASSIGNMENT WITH SCREENSHOTS:

1. Create a key space by name Library

```
cqlsh> describe keyspaces;

system_schema  system        system_distributed  system_traces
system_auth    test_keyspace  employee

cqlsh> CREATE KEYSPACE library WITH replication={'class':'SimpleStrategy','replication_factor':'1'};
cqlsh> describe keyspace library;

CREATE KEYSPACE library WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '1'}  AND durable_writes = true;

cqlsh> use library;
```

2. Create a column family by name Library-Info with attributes Stud_Id Primary Key, Counter_value of type Counter,
Stud_Name, Book-Name, Book-Id, Date_of_issue

```
cqlsh> use library;
cqlsh:library> create table Library_Info (Stud_Id int,Counter_value counter,Stud_Name text,Book_Name text,Book_Id int,Date_of_issue timestamp,PRIMARY KEY(Stud_Id,Stud_Name,
Book_Name,Book_Id,Date_of_issue));
cqlsh:library> describe table Library_Info

CREATE TABLE library.library_info (
    stud_id int,
    stud_name text,
    book_name text,
    book_id int,
    date_of_issue timestamp,
    counter_value counter,
    PRIMARY KEY (stud_id, stud_name, book_name, book_id, date_of_issue)
) WITH CLUSTERING ORDER BY (stud_name ASC, book_name ASC, book_id ASC, date_of_issue ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';
```

3. Insert the values into the table in batch

4. Display the details of the table created and increase the value of the counter

```
cqlsh:library> update Library_info set counter_value=counter_value+1 where stud_id=1 and stud_name='Vamshi' and book_name='OOMD' and book_id=100 and date_of_issue='2022-
18';
cqlsh:library> update Library_info set counter_value=counter_value+1 where stud_id=2 and stud_name='Ravi' and book_name='CNS' and book_id=101 and date_of_issue='2022-03-
;
cqlsh:library> update Library_info set counter_value=counter_value+1 where stud_id=112 and stud_name='Ramesh' and book_name='BDA' and book_id=102 and date_of_issue='2022
-19';
cqlsh:library> select*from Library_Info;

 stud_id | stud_name | book_name | book_id | date_of_issue                | counter_value
---------+-----------+-----------+---------+------------------------------+---------------
       1 |    Vamshi |      OOMD |     100 | 2022-04-17 18:30:00.000000+0000 |             1
       2 |      Ravi |       CNS |     101 | 2022-03-14 18:30:00.000000+0000 |             1
     112 |    Ramesh |       BDA |     102 | 2022-03-18 18:30:00.000000+0000 |             1

(3 rows)
```

5. Write a query to show that a student with id 112 has taken a book “BDA” 2 times.

```
cqlsh:library> update Library_info set counter_value=counter_value+1 where stud_id=112 and stud_name='Ramesh' and book_name='BDA' and book_id=102 and date_
-19';
cqlsh:library> select*from Library_Info;

 stud_id | stud_name | book_name | book_id | date_of_issue                | counter_value
---------+-----------+-----------+---------+------------------------------+---------------
       1 |    Vamshi |      OOMD |     100 | 2022-04-17 18:30:00.000000+0000 |             1
       2 |      Ravi |       CNS |     101 | 2022-03-14 18:30:00.000000+0000 |             1
     112 |    Ramesh |       BDA |     102 | 2022-03-18 18:30:00.000000+0000 |             2

(3 rows)
```

6. Export the created column to a csv file

```
cqlsh:library> copy Library_Info (stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) to 'C:\Users\shara\OneDrive\Documents\Library_Info.csv' with header=true

Using 7 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_name, book_id, date_of_issue, counter_value].
Processed: 3 rows; Rate:       2 rows/s; Avg. rate:       1 rows/s
3 rows exported to 1 files in 3.164 seconds.
```

	A	B	C	D	E	F	G	H
1	stud_id	stud_nam	book_nam	book_id	date_of_is	counter_value		
2	112	Ramesh	BDA	102	2022-03-1	2		
3	1	Vamshi	OOMD	100	2022-04-1	1		
4	2	Ravi	CNS	101	2022-03-1	1		
5								

7. Import a given csv dataset from local file system into Cassandra column family

```
cqlsh:library> create table Library_test(stud_id int,counter_value counter,stud_name text,book_name text,book_id int,date_of_issue timestamp,primary key(stud_id,stud_name,b
ook_name,book_id,date_of_issue));
cqlsh:library> COPY Library_test(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) FROM 'C:\Users\shara\OneDrive\Documents\Library_Info.csv' with header=true
;
Using 7 child processes

Starting copy of library.library_test with columns [stud_id, stud_name, book_name, book_id, date_of_issue, counter_value].
PProcess ImportProcess-8:      1 rows/s; Avg. rate:       1 rows/s
```

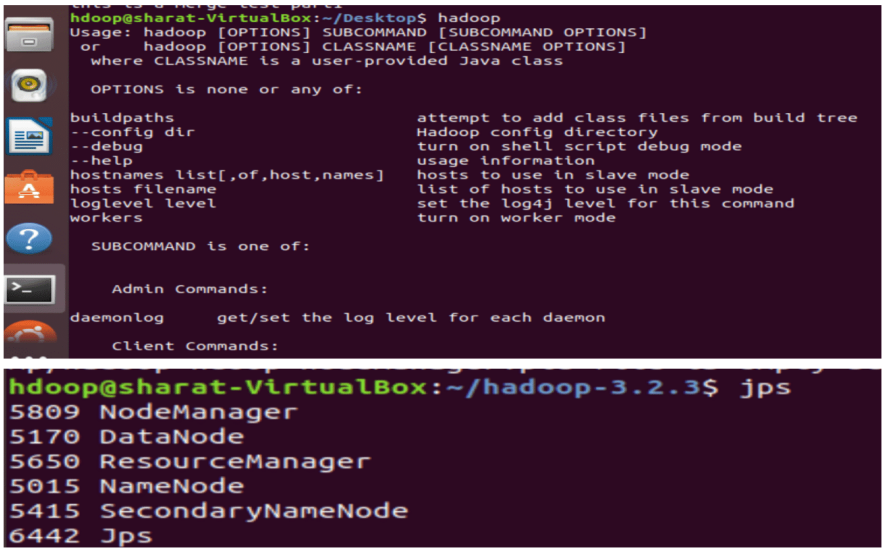
```
AttributeError: 'NoneType' object has no attribute 'add_timer'
Processed: 3 rows; Rate:       1 rows/s; Avg. rate:       0 rows/s
3 rows imported from 1 files in 6.260 seconds (0 skipped).
cqlsh:library> select*from Library_test;

 stud_id | stud_name | book_name | book_id | date_of_issue                     | counter_value
---------+-----------+-----------+---------+-----------------------------------+---------------
       1 |    Vamshi |      OOMD |     100 | 2022-04-17 18:30:00.000000+0000 |             1
       2 |      Ravi |       CNS |     101 | 2022-03-14 18:30:00.000000+0000 |             1
     112 |    Ramesh |       BDA |     102 | 2022-03-18 18:30:00.000000+0000 |             2

(3 rows)
cqlsh:library>
```

4. Execution of HDFS Commands for interaction with Hadoop Environment.

HADOOP WEEK5 LAB ASSIGNMENT

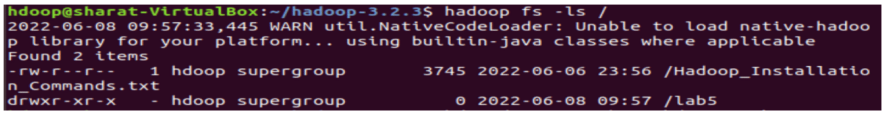
USN: 1BM19CS074
NAME: KIZHAKEL SHARAT PRASAD

1. Successful installation proof



2. Mkdir

3. Ls



4. put

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -put /home/hdoop/Desktop/a.txt
 /lab5
2022-06-08 10:02:57,394 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hadoop fs -ls /
2022-06-08 10:03:08,676 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hdoop supergroup       3745 2022-06-06 23:56 /Hadoop_Installatio
n_Commands.txt
drwxr-xr-x   - hdoop supergroup          0 2022-06-08 10:02 /lab5
```

5. copyFromLocal

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -copyFromLocal /home/hdoop/Des
ktop/b.txt /home/hdoop/Desktop/c.txt /lab5
2022-06-08 10:07:13,375 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -cat /lab52022-06-08 10:07:36,
122 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your p
latform... using builtin-java classes where applicable
cat: `/lab5': Is a directory
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -ls /lab5
2022-06-08 10:08:26,214 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
Found 3 items
-rw-r--r--   1 hdoop supergroup         15 2022-06-08 10:02 /lab5/a.txt
-rw-r--r--   1 hdoop supergroup          0 2022-06-08 10:07 /lab5/b.txt
-rw-r--r--   1 hdoop supergroup          0 2022-06-08 10:07 /lab5/c.txt
```

6. Get

I.get

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -get /lab5/a.txt /home/hdoop/D
esktop/test.txt
2022-06-08 10:10:29,649 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -ls /home/hdoop/Desktop
2022-06-08 10:11:08,053 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
ls: `/home/hdoop/Desktop': No such file or directory
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ ls /home/hdoop/Desktop
a.txt  b.txt  c.txt  test.txt
```

Ii.getmerge

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -getmerge /lab5/b.txt /lab5/c.
txt /home/hdoop/Desktop/merge.txt
2022-06-08 10:15:24,732 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ ls /home/hdoop/Desktop
a.txt  b.txt  c.txt  merge.txt  test.txt
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ cat /home/hdoop/Desktop/merge.txt
```

iii. getfacl

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -getfacl /lab5
2022-06-08 10:17:26,801 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
# file: /lab5
# owner: hdoop
# group: supergroup
user::rwx
group::r-x
other::r-x
```

7.copyToLocal

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -copyToLocal /lab5/a.txt /home
/hdoop/Documents
2022-06-08 10:19:08,660 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ ls /home/hdoop/Documents
a.txt
```

8.cat

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hdfs dfs -cat /lab5/a.txt
2022-06-08 10:19:48,077 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
this is a test
```

9.mv

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hadoop fs -mv /lab5/a.txt /lab5_part2
2022-06-08 10:22:18,644 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
```

10.cp

```
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hadoop fs -cp /lab5/b.txt /lab5_part2
2022-06-08 10:23:16,644 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
hdoop@sharat-VirtualBox:~/hadoop-3.2.3$ hadoop fs -ls /lab5_part2
2022-06-08 10:23:21,944 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hdoop supergroup         15 2022-06-08 10:02 /lab5_part2/a.txt
-rw-r--r--   1 hdoop supergroup          0 2022-06-08 10:23 /lab5_part2/b.txt
```