# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**"JnanaSangama", Belgaum -590014, Karnataka.**

**LAB REPORT**

**on**

# BIG DATA ANALYTICS
# (20CS6PEBDA)

***Submitted by***

**MD Suraj Kumar (1BM20CS079)**

***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-2023 to July-2023**

# 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 **MD Suraj Kumar (1BM20CS079),** who is a 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.

**Dr.Selva Kumar S**
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

`

# Index Sheet

| Sl.No. | Experiment Title |
|---|---|
| 1 | Cassandra Lab Program1: - Employee Database |
| 2 | Cassandra Lab Program1: - Library Database |
| 3 | MongoDB- CRUD Demonstration |
| 4 | Hadoop installation |
| 5 | Hadoop Commands |
| 6 | Hadoop Program: Average Temperature |
| 7 | Hadoop Program: Word Count |
| 8 | Hadoop program: Join operation |
| 9 | Scala Program |
| 10 | Scala Program: Word Count |

## Course Outcome

| | |
|---|---|
| CO1 | Apply the concept of NoSQL, Hadoop or Spark for a given task |
| C02 | Analyze the Big Data and obtain insight using data analytics mechanisms. |
| C03 | Design and implement big data applications by applying NoSQL, Hadoop or Spark |

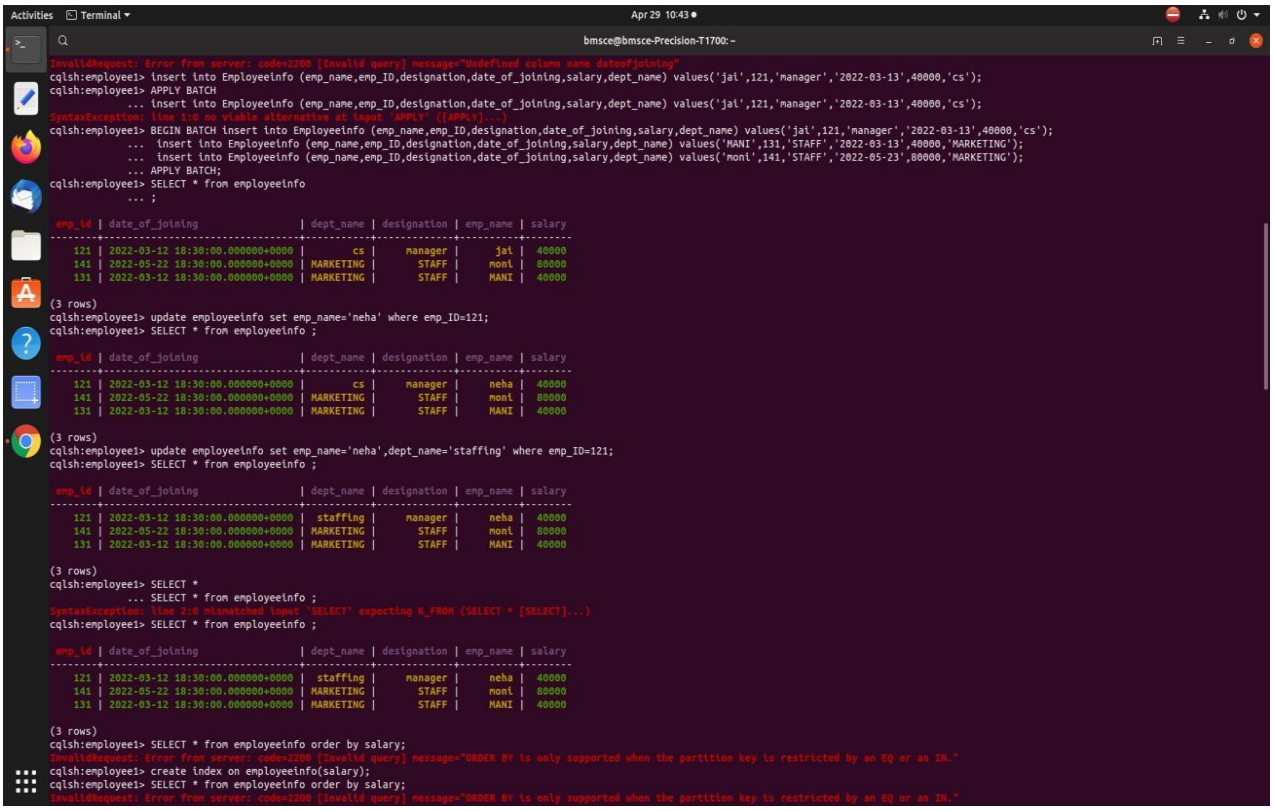
# LAB – 01

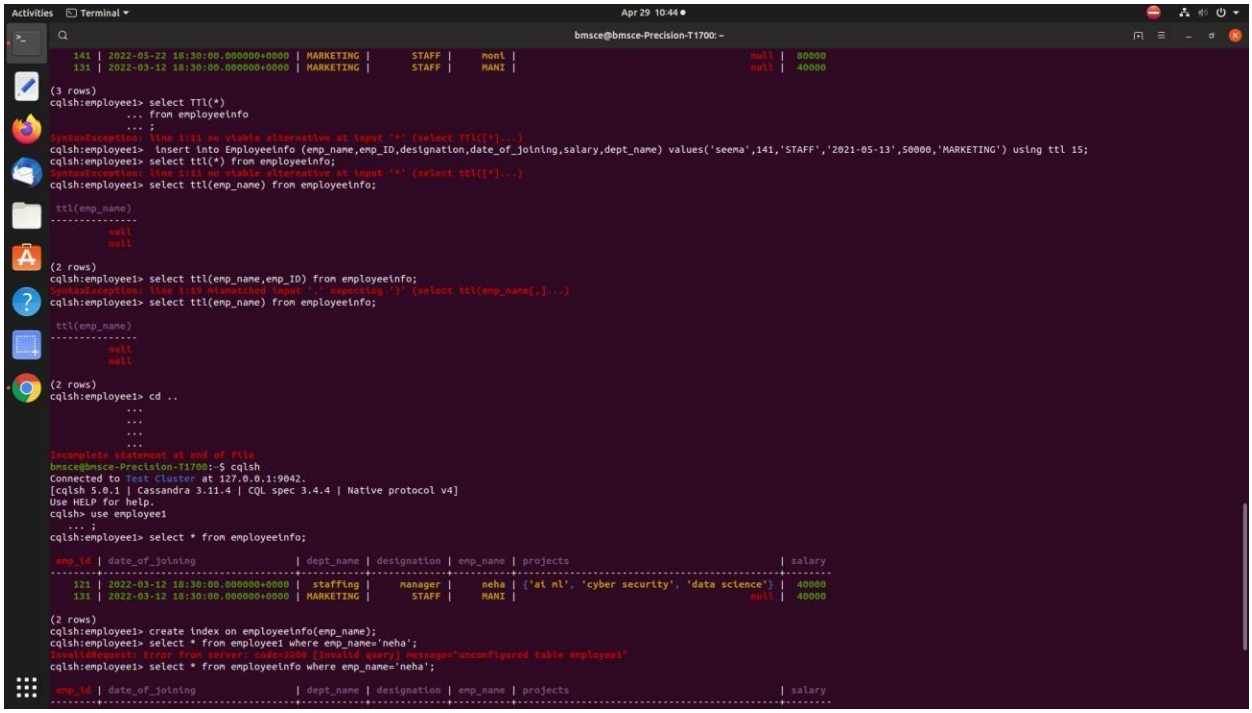
**AIM: Perform the following DB operations using Cassandra.**

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 Employees.

```
cqlsh:employee> CREATE KEYSPACE employee WITH REPLICATION={ 'class' : 'SimpleStrategy', 'replication_factor' : 1};

cqlsh:employee> USE employee;

cqlsh:employee> create table employee_info(emp_id int PRIMARY KEY, emp_name text,
           ... designation text, date_of_joining timestamp, salary double PRIMARY KEY, dept_name text);

cqlsh:employee> CREATE TABLE employee_info(emp_id int, emp_name text, designation text, date_of_joining timestamp, salary double, dept_name text, PRIMARY KEY(emp_id, salary));

cqlsh:employee> BEGIN BATCH INSERT INTO
           ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
           ... VALUES(100,'John','MANAGER','2021-09-11',30000,'TESTING');
           ... INSERT INTO
           ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
           ... VALUES(111,'Tom','ASSOCIATE','2021-06-22',25000,'DEVELOPING');
           ... INSERT INTO
           ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
           ... VALUES(121,'Elsa','MANAGER','2021-03-30',35000,'HR');
           ... INSERT INTO
```

```
        ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
        ... VALUES(115,'Chris','ASSISTANT','2021-12-30',20000,'DEVELOPING');
        ... INSERT INTO
        ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
        ... VALUES(105,'Sarah','ASSOCIATE','2021-06-25',25000,'TESTING');
        ... APPLY BATCH;
cqlsh:employee> SELECT * FROM employee_info
```


Activities
Terminal
Apr 29 10:43
bmsce@bmsce-Precision-T1700: ~
InvalidRequest: Error from server: code=2200 [Invalid query] message="Undefined column name dateofjoining"
cqlsh:employee1> insert into Employeeinfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('jai',121,'manager','2022-03-13',40000,'cs');
cqlsh:employee1> APPLY BATCH
... insert into Employeeinfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('jai',121,'manager','2022-03-13',40000,'cs');
SyntaxException: line 1:0 no viable alternative at input 'APPLY' ([APPLY]...)
cqlsh:employee1> BEGIN BATCH insert into Employeeinfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('jai',121,'manager','2022-03-13',40000,'cs');
... insert into Employeeinfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('MANI',131,'STAFF','2022-03-13',40000,'MARKETING');
... insert into Employeeinfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('moni',141,'STAFF','2022-05-23',80000,'MARKETING');
... APPLY BATCH;
cqlsh:employee1> SELECT * from employeeinfo
... ;
emp_id | date_of_joining | dept_name | designation | emp_name | salary
121 | 2022-03-12 18:30:00.000000+0000 | cs | manager | jai | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | moni | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | MANI | 40000
(3 rows)
cqlsh:employee1> update employeeinfo set emp_name='neha' where emp_ID=121;
cqlsh:employee1> SELECT * from employeeinfo ;
emp_id | date_of_joining | dept_name | designation | emp_name | salary
121 | 2022-03-12 18:30:00.000000+0000 | cs | manager | neha | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | moni | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | MANI | 40000
(3 rows)
cqlsh:employee1> update employeeinfo set emp_name='neha',dept_name='staffing' where emp_ID=121;
cqlsh:employee1> SELECT * from employeeinfo ;
emp_id | date_of_joining | dept_name | designation | emp_name | salary
121 | 2022-03-12 18:30:00.000000+0000 | staffing | manager | neha | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | moni | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | MANI | 40000
(3 rows)
cqlsh:employee1> SELECT *
... SELECT * from employeeinfo ;
SyntaxException: line 2:0 mismatched input 'SELECT' expecting K_FROM (SELECT * [SELECT]...)
cqlsh:employee1> SELECT * from employeeinfo ;
emp_id | date_of_joining | dept_name | designation | emp_name | salary
121 | 2022-03-12 18:30:00.000000+0000 | staffing | manager | neha | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | moni | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | MANI | 40000
(3 rows)
cqlsh:employee1> SELECT * from employeeinfo order by salary;
InvalidRequest: Error from server: code=2200 [Invalid query] message="ORDER BY is only supported when the partition key is restricted by an EQ or an IN."
cqlsh:employee1> create index on employeeinfo(salary);
cqlsh:employee1> SELECT * from employeeinfo order by salary;
InvalidRequest: Error from server: code=2200 [Invalid query] message="ORDER BY is only supported when the partition key is restricted by an EQ or an IN."


Activities
Terminal
Apr 29 10:44
bmsce@bmsce-Precision-T1700: ~
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | moni | null | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | MANI | null | 40000
(3 rows)
cqlsh:employee1> select TTl(*)
... from employeeinfo
... ;
SyntaxException: line 1:11 no viable alternative at input '*' (select TTl([*])...)
cqlsh:employee1> insert into Employeeinfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('seema',141,'STAFF','2021-05-13',50000,'MARKETING') using ttl 15;
cqlsh:employee1> select ttl(*) from employeeinfo;
SyntaxException: line 1:11 no viable alternative at input '*' (select ttl([*])...)
cqlsh:employee1> select ttl(emp_name) from employeeinfo;
ttl(emp_name)
null
null
(2 rows)
cqlsh:employee1> select ttl(emp_name,emp_ID) from employeeinfo;
SyntaxException: line 1:19 mismatched input ',' expecting ')' (select ttl(emp_name[,]...)
cqlsh:employee1> select ttl(emp_name) from employeeinfo;
ttl(emp_name)
null
null
(2 rows)
cqlsh:employee1> cd ..
...
...
...
...
Incomplete statement at end of file
bmsce@bmsce-Precision-T1700:~$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.4 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> use employee1
... ;
cqlsh:employee1> select * from employeeinfo;
emp_id | date_of_joining | dept_name | designation | emp_name | projects | salary
121 | 2022-03-12 18:30:00.000000+0000 | staffing | manager | neha | {'ai ml', 'cyber security', 'data science'} | 40000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | MANI | null | 40000
(2 rows)
cqlsh:employee1> create index on employeeinfo(emp_name);
cqlsh:employee1> select * from employee1 where emp_name='neha';
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured table employee1"
cqlsh:employee1> select * from employeeinfo where emp_name='neha';
emp_id | date_of_joining | dept_name | designation | emp_name | projects | salary

# LAB – 02

**AIM- 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

**1.** cqlsh:library> CREATE KEYSPACE library WITH replication = {'class': 'SimpleStrategy','replication_factor':1}; cqlsh:library> USE library ;

cqlsh:library> CREATE TABLE Library_info(stud_id int, stud_name text, book_name text, book_id text, date_of_issue timestamp, counter_value counter, PRIMARY KEY(stud_id,stud_name, book_name, book_id, date_of_issue));

cqlsh:library> BEGIN COUNTER BATCH

... UPDATE library_info set counter_value +=1 where stud_id = 111 and stud_name = 'Manoj' and book_name = 'Operations Research' and book_id = '56TXT' and date_of_issue = '2021-09-12';

... UPDATE library_info set counter_value +=1 where stud_id = 112 and stud_name = 'Kamal' and book_name = 'Engineering Mathematics-3' and book_id = '5ERW4' and date_of_issue = '2021-04-10';

... UPDATE library_info set counter_value +=1 where stud_id = 113 and stud_name = 'Mahesh' and book_name = 'Robinson Crusoe' and book_id = '34EDC' and date_of_issue = '2021-02-01';

... UPDATE library_info set counter_value +=1 where stud_id = 114 and stud_name = 'Raj' and book_name = 'Engineering Drawing' and book_id = '123ER' and date_of_issue = '2021-04-03';

... APPLY BATCH;

cqlsh:library> SELECT * FROM library_info ;

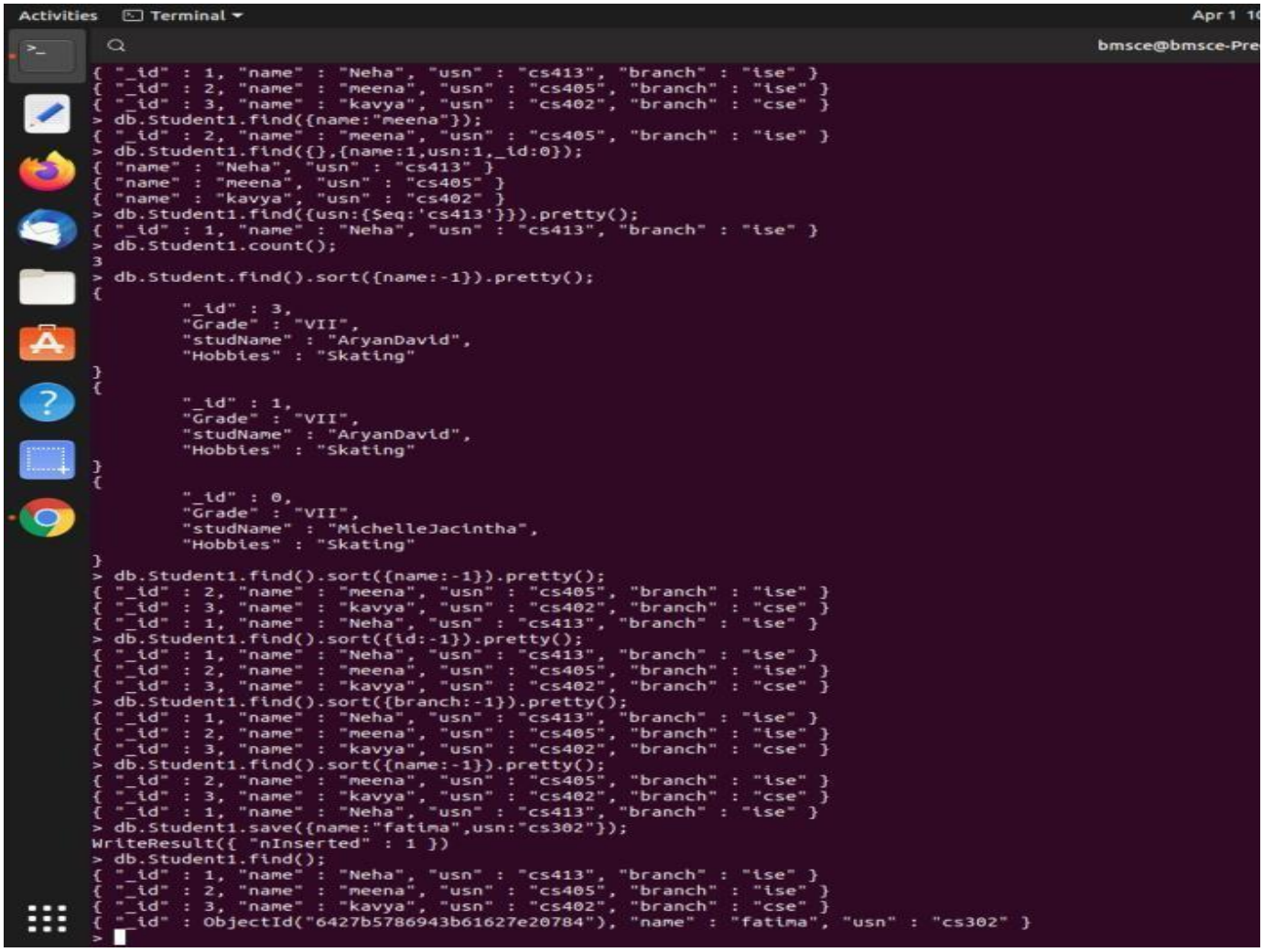
```
cqlsh> use library;
cqlsh:library> create table library_info (
          ... ... stud_id int,
          ... ... stud_name text,
          ... ... book_id int,
          ... ... book_name text,
          ... ... date_of_issue timestamp,
```

```
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 112 and stud_name = 'Ram' and book_id = 200 and book_name = 'DSA' and date_of_issue = '2022-04-06';
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 113 and stud_name = 'sohan' and book_id = 300 and book_name = 'JAVA' and date_of_issue = '2022-04-07';
cqlsh:library> update library_info set counter_value = counter_value+1  where stud_id = 111 and stud_name = 'Raj' and book_id = 100 and book_name = 'ADA' and date_of_issue = '2022-04-05';
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 114 and stud_name = 'rohan' and book_id = 400 and book_name = 'UNIX' and date_of_issue = '2022-04-07';
cqlsh:library> select * from library_info ;

 stud_id | book_id | stud_name | book_name | date_of_issue                | counter_value
---------+---------+-----------+-----------+------------------------------+---------------
     114 |     400 |     rohan |      UNIX | 2022-04-06 18:30:00.000000+0000 |             1
     111 |     100 |       Raj |       ADA | 2022-04-04 18:30:00.000000+0000 |             1
     112 |     200 |       Ram |       DSA | 2022-04-05 18:30:00.000000+0000 |             1
     113 |     300 |     sohan |      JAVA | 2022-04-06 18:30:00.000000+0000 |             1

(4 rows)
```

```
     113 |     300 |     sohan |      JAVA | 2022-04-06 18:30:00.000000+0000 |             1

(4 rows)
cqlsh:library>  copy library_info(stud_id,stud_name,book_id,book_name,date_of_issue,counter_value ) to '/home/bmscecse/Documents/library/lib.csv' with header=true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_id, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate:     122 rows/s; Avg. rate:     122 rows/s
4 rows exported to 1 files in 0.069 seconds.
cqlsh:library>  copy library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value ) to '/home/bmscecse/Documents/library/lib.csv' with header=true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate:     129 rows/s; Avg. rate:     129 rows/s
4 rows exported to 1 files in 0.066 seconds.
cqlsh:library> copy library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value ) from '/home/bmscecse/Documents/library/lib.csv' with header = true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate:       7 rows/s; Avg. rate:      11 rows/s
4 rows imported from 1 files in 0.375 seconds (0 skipped).
cqlsh:library> select * from library_info ;

 stud_id | book_id | stud_name | book_name | date_of_issue                | counter_value
---------+---------+-----------+-----------+------------------------------+---------------
     114 |     400 |     rohan |      UNIX | 2022-04-06 18:30:00.000000+0000 |             2
     111 |     100 |       Raj |       ADA | 2022-04-04 18:30:00.000000+0000 |             2
     112 |     200 |       Ram |       DSA | 2022-04-05 18:30:00.000000+0000 |             2
     113 |     300 |     sohan |      JAVA | 2022-04-06 18:30:00.000000+0000 |             2

(4 rows)
cqlsh:library>  truncate library_info;
cqlsh:library> select * from library_info ;

 stud_id | book_id | stud_name | book_name | date_of_issue | counter_value
---------+---------+-----------+-----------+---------------+---------------

(0 rows)
cqlsh:library> copy library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value ) from '/home/bmscecse/Documents/library/lib.csv' with header = true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate:       8 rows/s; Avg. rate:      11 rows/s
4 rows imported from 1 files in 0.370 seconds (0 skipped).
cqlsh:library> select * from library_info ;

 stud_id | book_id | stud_name | book_name | date_of_issue                | counter_value
---------+---------+-----------+-----------+------------------------------+---------------
     114 |     400 |     rohan |      UNIX | 2022-04-06 18:30:00.000000+0000 |             1
     111 |     100 |       Raj |       ADA | 2022-04-04 18:30:00.000000+0000 |             1
     112 |     200 |       Ram |       DSA | 2022-04-05 18:30:00.000000+0000 |             1
     113 |     300 |     sohan |      JAVA | 2022-04-06 18:30:00.000000+0000 |             1

(4 rows)
cqlsh:library>
```

# LAB – 03

**AIM- MongoDB- CRUD Demonstration:**




bmsce@bmsce-Precision-T1700:~$

mongo MongoDB shell version

v3.6.8

connecting to: mongodb://127.0.0.1:27017

Implicit session: session { "id" :

UUID("d66acdb3-8482-417d-8b75-d65dae4b53ee") } MongoDB server version:

3.6.8

> use

Student

switched to db

Student

> db.createCollection("student");

{ "ok" : 1 }

> db.Student.insert({_id:1,StudName:"Megha",Grade:"vii",Hobbies:"Intern
etSurfing"}); WriteResult({ "nInserted" : 1 })

> db.Student.update({_id:3,StudName:"Ayan",Grade:"vii"},{$set:{Hobbies:"skating"}},{u
psert:true}); WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 3 })

> db.Student.find({StudName:"Ayan"}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
> db.Student.find({},{StudName:1,Grade:1,_id:0});
{ "StudName" : "Megha", "Grade" : "vii" }
{ "Grade" : "vii", "StudName" : "Ayan" }

```
> db.Student.find({Grade:{$eq:'vii'}}).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
> db.Student.find({Grade:{$eq:'vii'}});
{ "_id" : 1, "StudName" : "Megha", "Grade" : "vii", "Hobbies" : "InternetSurfing" }
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
> db.Student.find({Grade:{$eq:'vii'}}).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
> db.Student.find({Hobbies:{$in:['Chess','Skating']}}).pretty();
> db.Student.find({Hobbies:{$in:['Skating']}}).pretty();
> db.Student.find({Hobbies:{$in:['skating']}}).pretty();
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
```

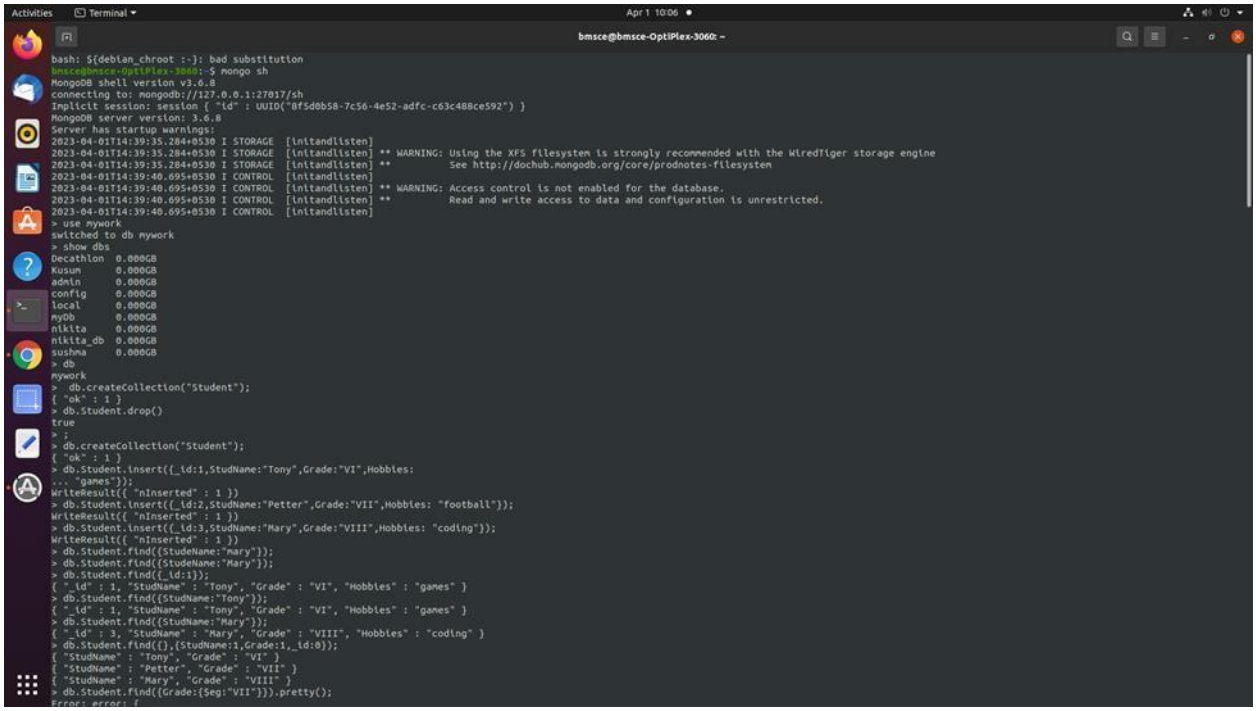
```
> db.Student.find({StudName:/^M/}).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
> db.Student.find({StudName:/e/}).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
>       db.Student.c
ount(); 2
> db.Student.find().sort({StudName:-1}).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
>       db.Student.save({StudName:"Vamsi",Greade
:"vi"}) WriteResult({ "nInserted" : 1 })
>       db.Students.update({_id:4},{$set:{Location:"Network"}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
```

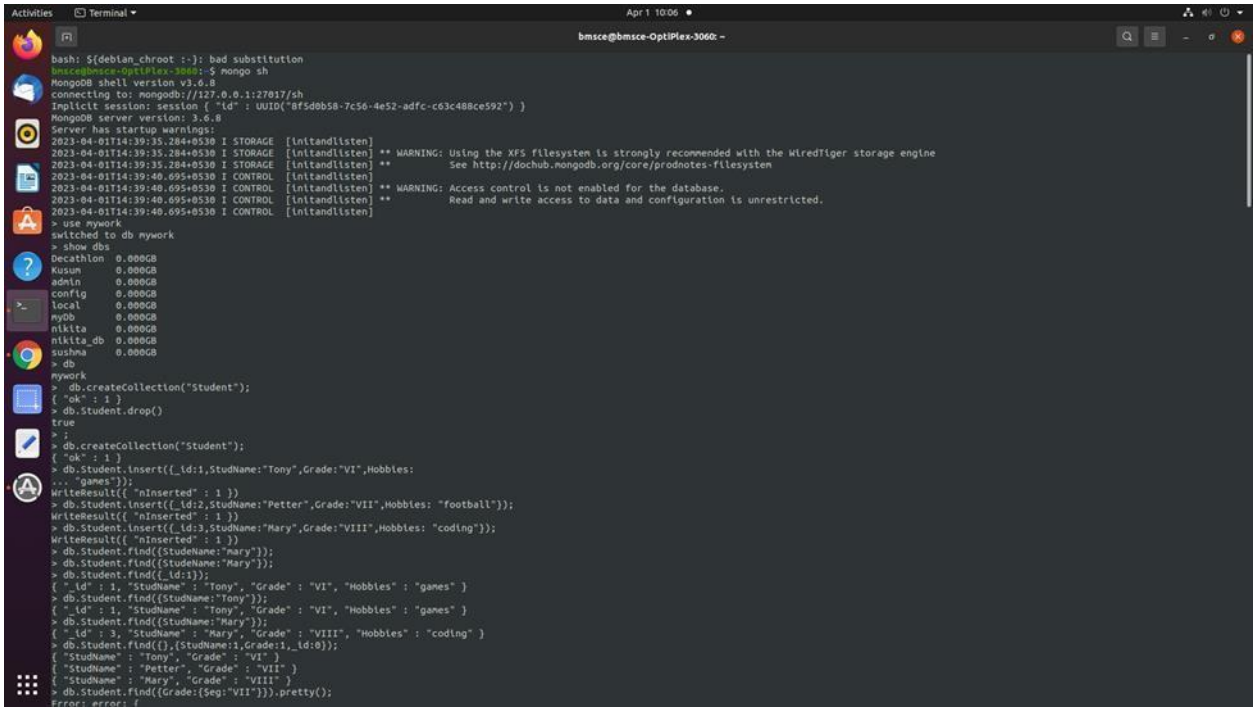
```
>          db.Students.update({_id:4},{$unset:{Location:"Network"
}}) WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" :
0 })
> db.Student.find({_id:1},{StudName:1,Grade:1,_id:0});
{ "StudName" : "Megha", "Grade" : "vii" }
> db.Student.find({Grade:{$ne:'VII'}}).pretty();
{
          "_id" : 1,
          "StudName" : "Megha",
          "Grade" : "vii",
          "Hobbies" : "InternetSurfing"
}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
{
          "_id" : ObjectId("6253f413e88b8c9e787b194e"),
          "StudName" : "Vamsi",
          "Grade" : "vi"
}
> db.Student.find({StudName:/s$/}).pretty();
>          db.Students.update({_id:3},{$set:{Location:null}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>          db.Students.c
ount() 0
>          db.Students.count({Grade:
"VII"}) 0
> db.Student.find({Grade:"VII"}).limit(3).pretty();
> db.Student.update({_id:3},{$set:{Location:null}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

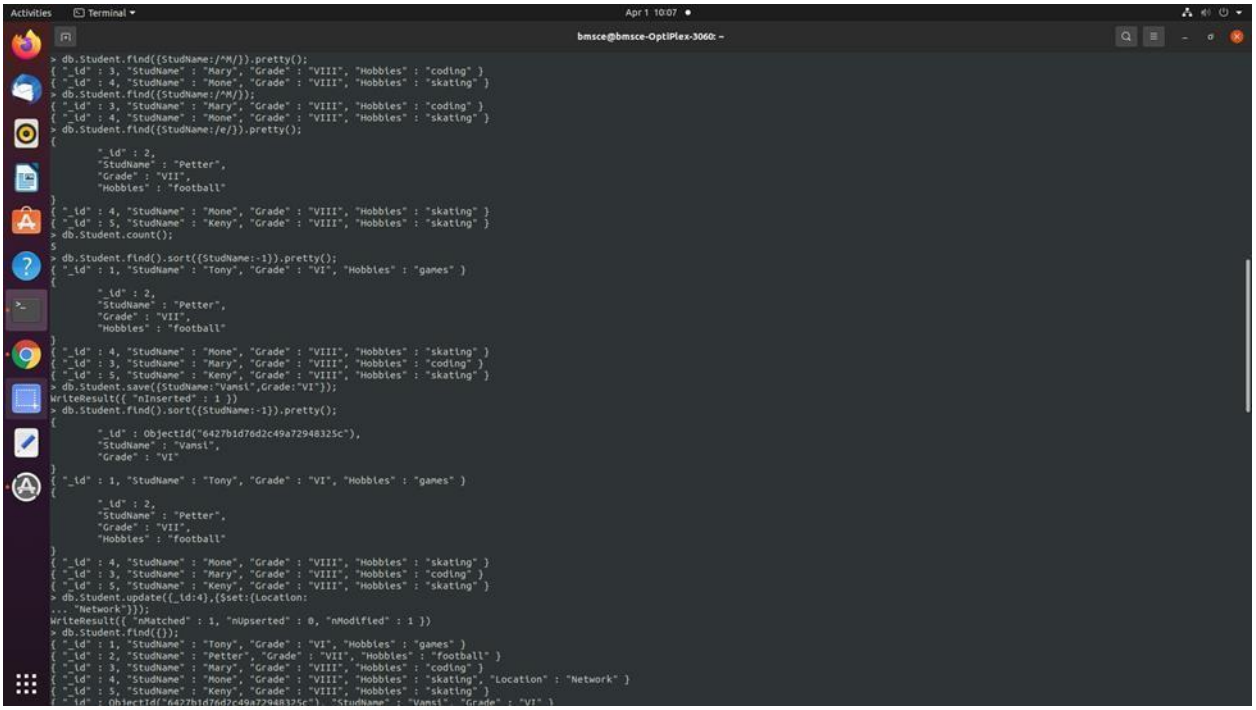
```
> db.Student.count({Grade:"VII"})
0
>       db.Students.count({Grade:
"vii"}) 0
>       db.Student.c
ount() 3
>       db.Student.count({Grade:
"vii"}) 2
> db.Student.find({Grade:"vii"}).limit(3).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : 3,
        "Grade" : "vii",
        "StudName" : "Ayan",
        "Hobbies" : "skating",
        "Location" : null
}
> db.Student.find().sort({StudName:1}).pretty();
{
        "_id"    :    3,
        "Grade"       :
        "vii",
        "StudName" : "Ayan",
        "Hobbies" : "skating",
```

```
        "Location" : null
}
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : ObjectId("6253f413e88b8c9e787b194e"),
        "StudName" : "Vamsi",
        "Grade" : "vi"
}
> db.Student.find().skip(2).pretty()
{
        "_id" : ObjectId("6253f413e88b8c9e787b194e"),
        "StudName" : "Vamsi",
        "Grade" : "vi"
}
> db.food.insert( { _id:1, fruits:['grapes','mango','apple';] } )
2022-04-11T15:05:51.894+0530 E QUERY    [thread1] SyntaxError: missing ] after element list @(shell):1:57
>       db.food.insert({_id:1,fruits:['grapes','mango','ap
ple']}) WriteResult({ "nInserted" : 1 })
>       db.food.insert({_id:2,fruits:['grapes','mango','che
rry']}) WriteResult({ "nInserted" : 1 })
>       db.food.insert({_id:3,fruits:['banana','ma
ngo']}) WriteResult({ "nInserted" : 1 })
> db.food.find({fruits:['grapes','mango','apple']}).pretty();
```

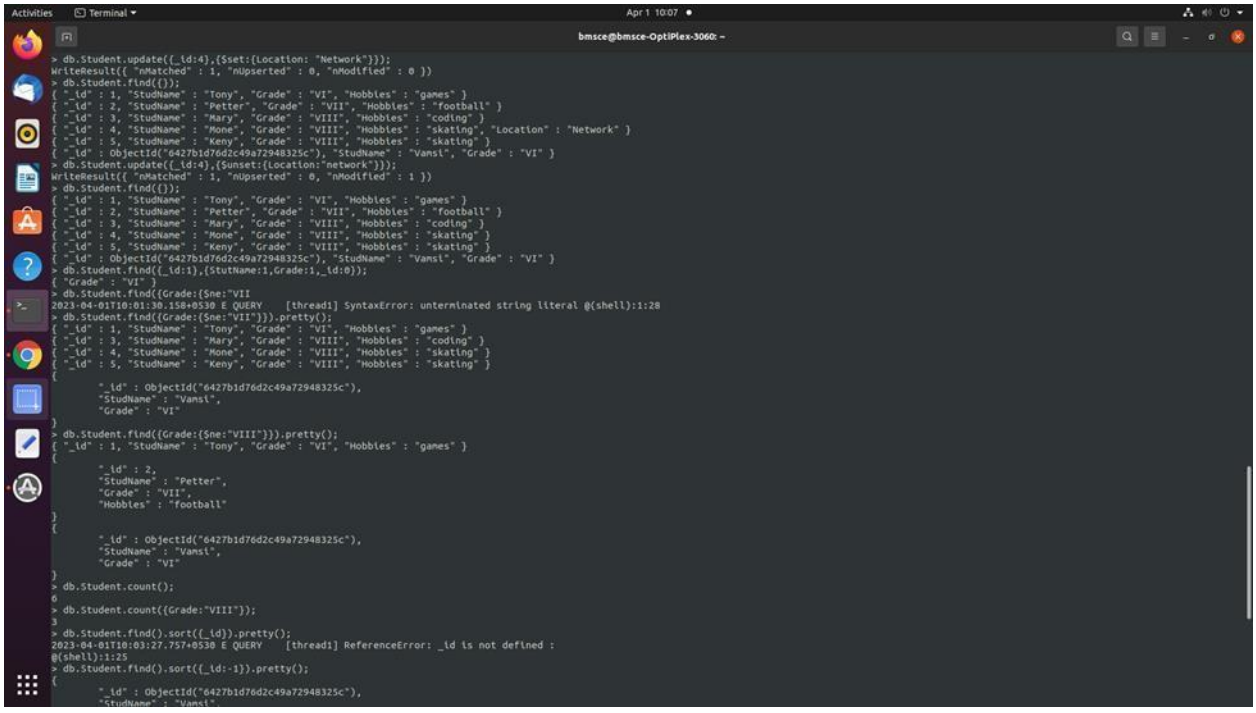
```
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
> db.food.find({'fruits.1':'grapes'})

> db.food.find({"fruits":{$size:2}})

{ "_id" : 3, "fruits" : [ "banana", "mango" ] }

> db.food.find({_id:1},{"fruits":{$slice:2}})

{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }

> db.food.find({fruits:{$all:["mango","grapes"]}})

{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }

{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }

>       db.food.update({_id:3},{$set:{"fruits.1":"apple"}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>       db.food.update({_id:2},{$push:{price:{grapes:80,mango:200,cherry:1

00}}}) WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })


>db.Customers.insert({_custID:1,AcctBal:'100000',AcctType:"saving"});

WriteResult({ "nInserted" : 1 })

> db.Customers.aggregate({$group:{_id:"$custID",TotAccBal:{$sum:"$AccBal"}}});

{ "_id" : null, "TotAccBal" : 0 }

db.Customers.aggregate({$match:{AcctType:"saving"}},{$group:{_id:"$custID",TotAccBal:{$sum:"$AccBal"}}});

{ "_id" : null, "TotAccBal" : 0 }

db.Customers.aggregate({$match:{AcctType:"saving"}},{$group:{_id:"$custID",TotAccBal:{$sum:"$AccBal"}}},{$
match:{TotAccBal:{$gt:1200}}
```


Activities
Terminal
Apr 1 10:06
bmsce@bmsce-OptiPlex-3060: ~
bash: ${debian_chroot :-}: bad substitution
bmsce@bmsce-OptiPlex-3060:~$ mongo sh
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017/sh
Implicit session: session { "id" : UUID("8f5d0b58-7c56-4e52-adfc-c63c488ce592") }
MongoDB server version: 3.6.8
Server has startup warnings:
2023-04-01T14:39:35.284+0530 I STORAGE [initandlisten]
2023-04-01T14:39:35.284+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2023-04-01T14:39:35.284+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten]
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten]
> use mywork
switched to db mywork
> show dbs
Decathlon 0.000GB
Kusum 0.000GB
admin 0.000GB
config 0.000GB
local 0.000GB
myDb 0.000GB
nikita 0.000GB
nikita_db 0.000GB
sushma 0.000GB
> db
mywork
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.drop()
true
> ;
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.insert({_id:1,StudName:"Tony",Grade:"VI",Hobbies:
... "games"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:2,StudName:"Petter",Grade:"VII",Hobbies: "football"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:3,StudName:"Mary",Grade:"VIII",Hobbies: "coding"});
WriteResult({ "nInserted" : 1 })
> db.Student.find({StudeName:"mary"});
> db.Student.find({StudeName:"Mary"});
> db.Student.find({_id:1});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
> db.Student.find({StudName:"Tony"});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
> db.Student.find({StudName:"Mary"});
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
> db.Student.find({},{StudName:1,Grade:1,_id:0});
{ "StudName" : "Tony", "Grade" : "VI" }
{ "StudName" : "Petter", "Grade" : "VII" }
{ "StudName" : "Mary", "Grade" : "VIII" }
> db.Student.find({Grade:{$eq:"VII"}}).pretty();
Error: error: {


Activities
Terminal
Apr 1 10:06
bmsce@bmsce-OptiPlex-3060: ~
bash: ${debian_chroot :-}: bad substitution
bmsce@bmsce-OptiPlex-3060:~$ mongo sh
MongoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017/sh
Implicit session: session { "id" : UUID("8f5d0b58-7c56-4e52-adfc-c63c488ce592") }
MongoDB server version: 3.6.8
Server has startup warnings:
2023-04-01T14:39:35.284+0530 I STORAGE [initandlisten]
2023-04-01T14:39:35.284+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2023-04-01T14:39:35.284+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten]
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2023-04-01T14:39:40.695+0530 I CONTROL [initandlisten]
> use mywork
switched to db mywork
> show dbs
Decathlon 0.000GB
Kusum 0.000GB
admin 0.000GB
config 0.000GB
local 0.000GB
myDb 0.000GB
nikita 0.000GB
nikita_db 0.000GB
sushma 0.000GB
> db
mywork
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.drop()
true
> ;
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.insert({_id:1,StudName:"Tony",Grade:"VI",Hobbies:
... "games"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:2,StudName:"Petter",Grade:"VII",Hobbies: "football"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:3,StudName:"Mary",Grade:"VIII",Hobbies: "coding"});
WriteResult({ "nInserted" : 1 })
> db.Student.find({StudeName:"mary"});
> db.Student.find({StudeName:"Mary"});
> db.Student.find({_id:1});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
> db.Student.find({StudName:"Tony"});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
> db.Student.find({StudName:"Mary"});
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
> db.Student.find({},{StudName:1,Grade:1,_id:0});
{ "StudName" : "Tony", "Grade" : "VI" }
{ "StudName" : "Petter", "Grade" : "VII" }
{ "StudName" : "Mary", "Grade" : "VIII" }
> db.Student.find({Grade:{$eq:"VII"}}).pretty();
Error: error: {




```
> db.Student.find({StudName:/^M/}).pretty();
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
> db.Student.find({StudName:/^M/});
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
> db.Student.find({StudName:/e/}).pretty();
{
        "_id" : 2,
        "StudName" : "Petter",
        "Grade" : "VII",
        "Hobbies" : "football"
}
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
> db.Student.count();
5
> db.Student.find().sort({StudName:-1}).pretty();
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{
        "_id" : 2,
        "StudName" : "Petter",
        "Grade" : "VII",
        "Hobbies" : "football"
}
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
> db.Student.save({StudName:"Vansi",Grade:"VI"});
WriteResult({ "nInserted" : 1 })
> db.Student.find().sort({StudName:-1}).pretty();
{
        "_id" : ObjectId("6427b1d76d2c49a72948325c"),
        "StudName" : "Vansi",
        "Grade" : "VI"
}
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{
        "_id" : 2,
        "StudName" : "Petter",
        "Grade" : "VII",
        "Hobbies" : "football"
}
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
> db.Student.update({_id:4},{$set:{Location:
... "Network"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find({});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{ "_id" : 2, "StudName" : "Petter", "Grade" : "VII", "Hobbies" : "football" }
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating", "Location" : "Network" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : ObjectId("6427b1d76d2c49a72948325c"), "StudName" : "Vansi", "Grade" : "VI" }
```




```
> db.Student.update({_id:4},{$set:{Location: "Network"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })
> db.Student.find({});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{ "_id" : 2, "StudName" : "Petter", "Grade" : "VII", "Hobbies" : "football" }
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating", "Location" : "Network" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : ObjectId("6427b1d76d2c49a72948325c"), "StudName" : "Vansi", "Grade" : "VI" }
> db.Student.update({_id:4},{$unset:{Location:"network"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find({});
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{ "_id" : 2, "StudName" : "Petter", "Grade" : "VII", "Hobbies" : "football" }
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : ObjectId("6427b1d76d2c49a72948325c"), "StudName" : "Vansi", "Grade" : "VI" }
> db.Student.find({_id:1},{StutName:1,Grade:1,_id:0});
{ "Grade" : "VI" }
> db.Student.find({Grade:{$ne:"VII
2023-04-01T10:01:30.158+0530 E QUERY    [thread1] SyntaxError: unterminated string literal @(shell):1:28
> db.Student.find({Grade:{$ne:"VII"}}).pretty();
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{ "_id" : 3, "StudName" : "Mary", "Grade" : "VIII", "Hobbies" : "coding" }
{ "_id" : 4, "StudName" : "Mone", "Grade" : "VIII", "Hobbies" : "skating" }
{ "_id" : 5, "StudName" : "Keny", "Grade" : "VIII", "Hobbies" : "skating" }
{
        "_id" : ObjectId("6427b1d76d2c49a72948325c"),
        "StudName" : "Vansi",
        "Grade" : "VI"
}
> db.Student.find({Grade:{$ne:"VIII"}}).pretty();
{ "_id" : 1, "StudName" : "Tony", "Grade" : "VI", "Hobbies" : "games" }
{
        "_id" : 2,
        "StudName" : "Petter",
        "Grade" : "VII",
        "Hobbies" : "football"
}
{
        "_id" : ObjectId("6427b1d76d2c49a72948325c"),
        "StudName" : "Vansi",
        "Grade" : "VI"
}
> db.Student.count();
6
> db.Student.count({Grade:"VIII"});
3
> db.Student.find().sort({_id}).pretty();
2023-04-01T10:03:27.757+0530 E QUERY    [thread1] ReferenceError: _id is not defined :
@(shell):1:25
> db.Student.find().sort({_id:-1}).pretty();
{
        "_id" : ObjectId("6427b1d76d2c49a72948325c"),
        "StudName" : "Vansi",
```

# LAB – 04

## AIM- Screenshot of Hadoop Installed

```
Microsoft Windows [Version 10.0.22000.739]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\WINDOWS\system32>jps
7072 DataNode
13492 Jps
15844 ResourceManager
16196 NameNode
1388 NodeManager

C:\WINDOWS\system32>hdfs dfs -ls -R /
drwxr-xr-x   - khush supergroup          0 2022-06-27 14:09 /input
drwxr-xr-x   - khush supergroup          0 2022-06-21 09:03 /input/inputtest
-rw-r--r--   1 khush supergroup         21 2022-06-21 09:03 /input/inputtest/output.txt
-rw-r--r--   1 khush supergroup         21 2022-06-21 08:19 /input/sample.txt
-rw-r--r--   1 khush supergroup         21 2022-06-27 14:09 /input/sample2.txt
drwxr-xr-x   - khush supergroup          0 2022-06-21 13:30 /test
-rw-r--r--   1 khush supergroup         19 2022-06-21 13:30 /test/sample.txt

C:\WINDOWS\system32>hadoop version
Hadoop 3.3.3
Source code repository https://github.com/apache/hadoop.git -r d37586cbda38c338d9fe481addda5a05fb516f71
Compiled by stevel on 2022-05-09T16:36Z
Compiled with protoc 3.7.1
From source with checksum eb96dd4a797b6989ae0cdb9db6efc6
This command was run using /C:/hadoop-3.3.3/share/hadoop/common/hadoop-common-3.3.3.jar

C:\WINDOWS\system32>
```

# LAB – 05

**AIM- Execution of HDFS Commands for Interaction with Hadoop**

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -cat /mydir/file1.txt
21/04/19 23:38:07 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
I am using Hadoop
line1
line2
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -copyFromLocal ~/file1.txt /my
dir
21/04/19 23:19:36 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /mydir
21/04/19 23:20:13 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup         30 2021-04-19 23:19 /mydir/file1.txt
hduser@lab-VirtualBox:/usr/local/sbin$
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -copyToLocal /mydir ~/hadoopco
py
21/04/19 23:29:39 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /
21/04/19 23:48:41 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 2 items
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:45 /mydir
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:41 /newdir
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -cp /mydir/sample.txt /newdir
21/04/19 23:48:56 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /newdir
21/04/19 23:49:22 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 2 items
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:21 /newdir/mydr
-rw-r--r--   1 hduser supergroup         13 2021-04-19 23:48 /newdir/sample.txt
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -mkdir /mydir
21/04/19 22:58:30 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /
21/04/19 23:41:08 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 3 items
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:19 /mydir
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:21 /mydr
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:39 /newdir
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -mv /mydr /newdir
21/04/19 23:41:38 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /
21/04/19 23:41:44 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 2 items
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:19 /mydir
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:41 /newdir
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /newdir
21/04/19 23:42:05 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 1 items
drwxr-xr-x   - hduser supergroup          0 2021-04-19 23:21 /newdir/mydr
hduser@lab-VirtualBox:/usr/local/sbin$
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -put ~/file1.txt /mydr
21/04/19 23:21:41 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$ hadooop fs -ls /mydr
hadooop: command not found
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /mydr
21/04/19 23:22:20 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup         30 2021-04-19 23:21 /mydr/file1.txt
```

# LAB – 06

**AIM-Create a Map Reduce program to**

a) find average temperature for each year from the NCDC data set.

**AverageDriver**

```
package temp;
import org.apache.hadoop.fs.Path; import
org.apache.hadoop.io.IntWritable; import
org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; public class AverageDriver {
public static void main(String[] args) throws Exception {
if (args.length != 2) {
System.err.println("Please Enter the input and output parameters"); System.exit(-1);
}
Job job = new Job();
job.setJarByClass(AverageDriver.class);
job.setJobName("Max temperature"); FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
job.setMapperClass(AverageMapper.class); job.setReducerClass(AverageReducer.class);23
job.setOutputKeyClass(Text.class); job.setOutputValueClass(IntWritable.class);
System.exit(job.waitForCompletion(true) ? 0 : 1);
}}
```

**AverageMapper**

```
package temp;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
```

```
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class AverageMapper extends Mapper<LongWritable, Text, Text, IntWritable> { public
static final int MISSING = 9999;
public void map(LongWritable key, Text value, Mapper<LongWritable, Text, Text,
IntWritable>.Context context) throws IOException, InterruptedException {
int temperature;
String line = value.toString();
String year = line.substring(15, 19);
if (line.charAt(87) == '+') {

temperature = Integer.parseInt(line.substring(88, 92));
} else {
temperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);24
if (temperature != 9999 && quality.matches("[01459]"))
context.write(new Text(year), new IntWritable(temperature));
}
}
```

**AverageReducer**

```
package temp;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class AverageReducer extends Reducer<Text, IntWritable, Text, IntWritable> { public
void
reduce(Text key, Iterable<IntWritable> values, Reducer<Text, IntWritable, Text,
```

IntWritable>.Context context) throws IOException, InterruptedException {

int max_temp = 0;

int count = 0;

for (IntWritable value : values) {

max_temp += value.get();

count++;

}

context.write(key, new IntWritable(max_temp / count));

}

}

**OUTPUT-**

```
hduser@lab-VirtualBox:/home/lab$ hadoop dfs -cat /tempmax/part-r-00000
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication
.util.KerberosUtil (file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-
2.6.0.jar) to method sun.security.krb5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop
.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflec
tive access operations
WARNING: All illegal access operations will be denied in a future release
21/05/10 16:08:48 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
03      111
05      22
```

b) **find the mean max temperature for every**

month MeanMaxDriver

MeanMaxDriver.class

package meanmax;

import org.apache.hadoop.fs.Path; import

org.apache.hadoop.io.IntWritable; import

org.apache.hadoop.io.Text;

```
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; public class MeanMaxDriver {
public static void main(String[] args) throws Exception {
if (args.length != 2) {
System.err.println("Please Enter the input and output parameters"); System.exit(-1);
}
Job job = new Job();
job.setJarByClass(MeanMaxDriver.class);
job.setJobName("Max temperature");
FileInputFormat.addInputPath(job, new Path(args[0])); FileOutputFormat.setOutputPath(job,
new Path(args[1])); job.setMapperClass(MeanMaxMapper.class);
job.setReducerClass(MeanMaxReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

MeanMax Mapper

**MeanMaxMapper.class**

```
package meanmax;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MeanMaxMapper extends Mapper<LongWritable, Text, Text, IntWritable> { public
static final int MISSING = 9999;
```

```
public void map(LongWritable key, Text value, Mapper<LongWritable, Text,
Text,IntWritable>.Context context) throws IOException, InterruptedException
{ int temperature;
String line = value.toString();
String month = line.substring(19, 21);
if (line.charAt(87) == '+') {
temperature = Integer.parseInt(line.substring(88, 92));
} else {
temperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (temperature != 9999 && quality.matches("[01459]"))
context.write(new Text(month), new IntWritable(temperature));
}
}
```

**MeanMax Reducer**

MeanMaxReducer.class

```
package meanmax;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class MeanMaxReducer extends Reducer<Text, IntWritable, Text, IntWritable> { public
void reduce(Text key, Iterable<IntWritable> values, Reducer<Text, IntWritable, Text,
IntWritable>.Context context) throws IOException, InterruptedException {
int max_temp = 0;
```

```
int total_temp = 0;
int count = 0;
int days = 0;
for (IntWritable value : values) { int temp = value.get();
if (temp > max_temp) max_temp = temp;
count++;
if (count == 3) {
total_temp += max_temp;
max_temp = 0;
count = 0;
days++;
}
}
context.write(key, new IntWritable(total_temp / days)); }
}
```

**OUTPUT-**

# LAB – 07

**AIM- For a given Text file, Create a Map Reduce program to sort the content in an alphabetic order listing only top 10 maximum occurrences of words**.

**Driver Code**

```
package wordCount;
import java.io.IOException;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;
public class WCDriver extends Configured implements Tool {
public int run(String args[]) throws IOException29
{
if (args.length < 2)
{
System.out.println("Please give valid inputs");
return -1;
}
JobConf conf = new JobConf(WCDriver.class);
FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
conf.setMapperClass(WCMapper.class);
```

```
conf.setReducerClass(WCReducer.class);
conf.setMapOutputKeyClass(Text.class);
conf.setMapOutputValueClass(IntWritable.class);
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
JobClient.runJob(conf);
return 0;
}
// Main Method
public static void main(String args[]) throws Exception
{
int exitCode = ToolRunner.run(new WCDriver(), args);
System.out.println(exitCode);
}
}
```

**Mapper Code**

```
package wordCount;30
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class WCMapper extends MapReduceBase implements Mapper<LongWritable,Text,
Text, IntWritable> {
// Map function
```

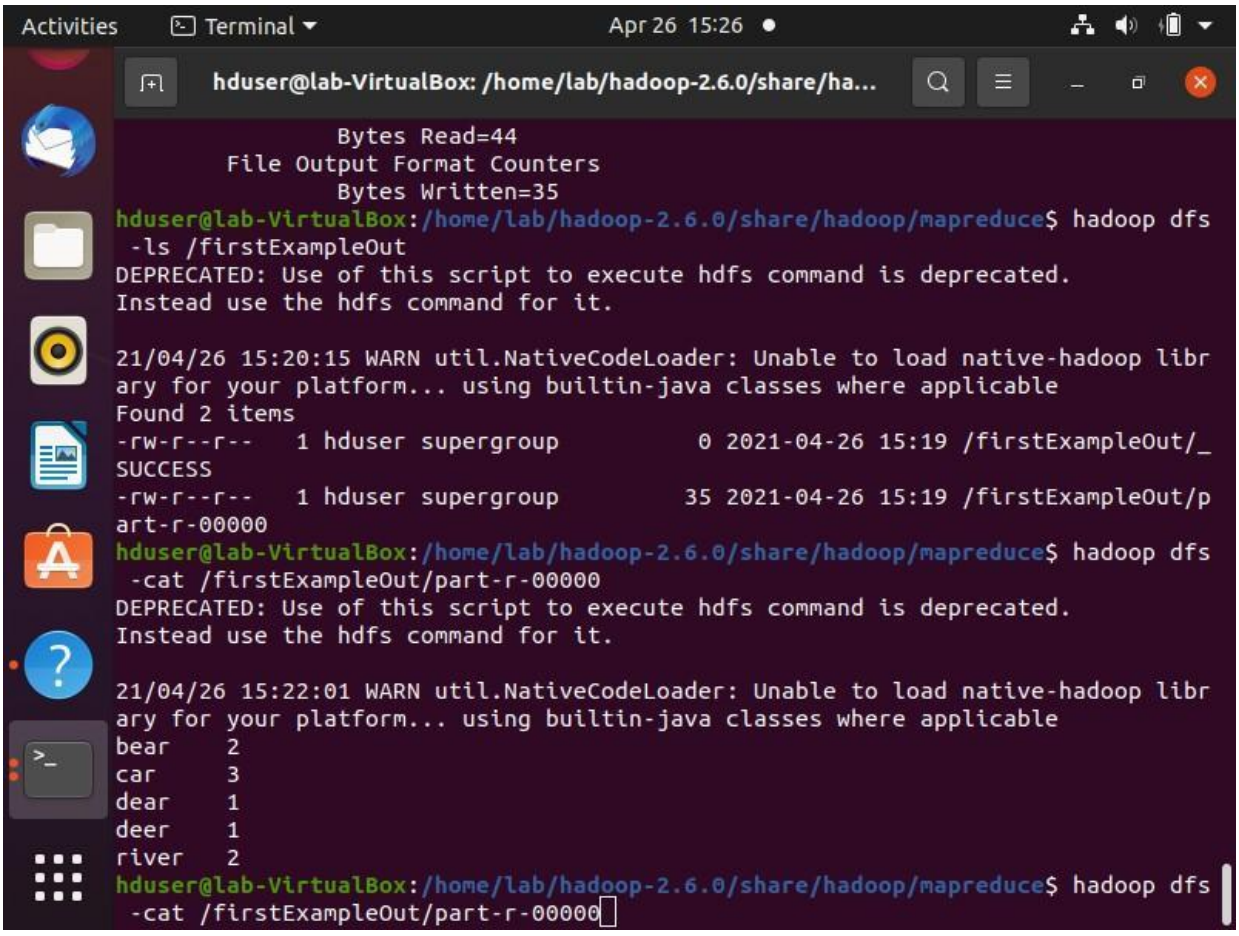
```
public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
Reporter

rep) throws IOException

{

String line = value.toString();

// Splitting the line on spaces

for (String word : line.split(" "))

{

if (word.length() > 0)

{

output.collect(new Text(word), new IntWritable(1));

}

}

}

}
```

**Reducer Code**

```
package wordCount;

import java.io.IOException;

import java.util.Iterator;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.MapReduceBase;

import  org.apache.hadoop.mapred.OutputCollector;

import org.apache.hadoop.mapred.Reducer;

import org.apache.hadoop.mapred.Reporter;

public class WCReducer extends MapReduceBase implements Reducer<Text,IntWritable, Text,
IntWritable> {

// Reduce function

public void reduce(Text key, Iterator<IntWritable> value, OutputCollector<Text, IntWritable>
```

```
output,Reporter rep) throws IOException
{
int count = 0;
// Counting the frequency of each words
while (value.hasNext())
{
IntWritable i = value.next();
count += i.get();
}
output.collect(key, new IntWritable(count));
}
}
```

OUTPUT-



# LAB – 08

**AIM- Create a Map Reduce program to demonstrating join operation**

**JoinDriver.java**

```
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.mapred.lib.MultipleInputs;
import org.apache.hadoop.util.*;
public class JoinDriver extends Configured implements Tool {
public static class KeyPartitioner implements Partitioner<TextPair, Text> {
@Override
public void configure(JobConf job) {}
@Override
public int getPartition(TextPair key, Text value, int numPartitions) {
return (key.getFirst().hashCode() & Integer.MAX_VALUE) %
numPartitions;
}
}
@Override
public int run(String[] args) throws Exception {
if (args.length != 3) {
System.out.println("Usage: <Department Emp Strength input>
<Department Name input> <output>");
return -1;
}40
```

```
JobConf conf = new JobConf(getConf(), getClass());
conf.setJobName("Join 'Department Emp Strength input' with 'Department Nameinput'");
Path AInputPath = new Path(args[0]);
Path BInputPath = new Path(args[1]);
Path outputPath = new Path(args[2]);
MultipleInputs.addInputPath(conf, AInputPath, TextInputFormat.class, Posts.class);
MultipleInputs.addInputPath(conf, BInputPath, TextInputFormat.class, User.class);
FileOutputFormat.setOutputPath(conf, outputPath);
conf.setPartitionerClass(KeyPartitioner.class);
conf.setOutputValueGroupingComparator(TextPair.FirstComparator.class);
conf.setMapOutputKeyClass(TextPair.class);
conf.setReducerClass(JoinReducer.class);
conf.setOutputKeyClass(Text.class);
JobClient.runJob(conf);
return 0;
}
public static void main(String[] args) throws Exception {
int exitCode = ToolRunner.run(new JoinDriver(), args);
System.exit(exitCode);
}
}
```

**JoinReducer.java**

```
import java.io.IOException;
import java.util.Iterator;41
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
public class JoinReducer extends MapReduceBase implements Reducer<TextPair, Text, Text,
Text> {
@Override
```

```
public void reduce (TextPair key, Iterator<Text> values, OutputCollector<Text, Text>output,
Reporter reporter)
throws IOException
{
Text nodeId = new Text(values.next());
while (values.hasNext()) {
Text node = values.next();
Text outValue = new Text(nodeId.toString() + "\t\t" + node.toString());
output.collect(key.getFirst(), outValue);
}
}
}
```

**User.java**

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;42
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.IntWritable;
public class User extends MapReduceBase implements Mapper<LongWritable, Text, TextPair,
Text> {
@Override
public void map(LongWritable key, Text value, OutputCollector<TextPair, Text> output,
Reporter reporter)
```

```
throws IOException
{
String valueString = value.toString();
String[] SingleNodeData = valueString.split("\t");
output.collect(new TextPair(SingleNodeData[0], "1"), new
Text(SingleNodeData[1]));
}
}
```

**Posts.java**

```
import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
public class Posts extends MapReduceBase implements Mapper<LongWritable, Text, TextPair, Text> {
@Override
public void map(LongWritable key, Text value, OutputCollector<TextPair, Text> output, Reporter reporter)
throws IOException43
{
String valueString = value.toString();
String[] SingleNodeData = valueString.split("\t");
output.collect(new TextPair(SingleNodeData[3], "0"), new
Text(SingleNodeData[9]));
}
}
```

**TextPair.java**

```
import java.io.*;
import org.apache.hadoop.io.*;
public class TextPair implements WritableComparable<TextPair> {
```

```
private Text first;
private Text second;
public TextPair() {
set(new Text(), new Text());
}
public TextPair(String first, String second) {
Set(new Text(first), new Text(second));
}
public TextPair(Text first, Text second) {
set(first, second);
}
public void set(Text first, Text second) {
this.first = first;
this.second = second;
}
public Text getFirst() {
return first;44
}
public Text getSecond() {
return second;
}
@Override
public void write(DataOutput out) throws IOException {
first.write(out);
second.write(out);
}
@Override
public void readFields(DataInput in) throws IOException {
```

```
first.readFields(in);
second.readFields(in);
}
@Override
public int hashCode() {
return first.hashCode() * 163 + second.hashCode();
}
@Override
public boolean equals(Object o) {
if (o instanceof TextPair) {
TextPair tp = (TextPair) o;
return first.equals(tp.first) && second.equals(tp.second);
}
return false;
}
@Override
public String toString() {45
return first + "\t" + second;
}
@Override
public int compareTo(TextPair tp) {
int cmp = first.compareTo(tp.first);
if (cmp != 0) {
return cmp;
}
return second.compareTo(tp.second);
}
// ^^ TextPair
```

```
// vv TextPairComparator
public static class Comparator extends WritableComparator {
private static final Text.Comparator TEXT_COMPARATOR = new Text.Comparator();
public Comparator() {
super(TextPair.class);
}
@Override
public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {
try {
int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1);
int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);
int cmp = TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);
if (cmp != 0) {
return cmp;
}
return TEXT_COMPARATOR.compare(b1, s1 + firstL1, l1 - firstL1,46
b2, s2 + firstL2, l2 - firstL2);
} catch (IOException e) {
throw new IllegalArgumentException(e);
}
}
}
static {
WritableComparator.define(TextPair.class, new Comparator());
}
public static class FirstComparator extends WritableComparator {
private static final Text.Comparator TEXT_COMPARATOR = new Text.Comparator();
public FirstComparator() {
```

```
super(TextPair.class);
}
@Override
public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {
try {
int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1);
int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);
return TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);
} catch (IOException e) {
throw new IllegalArgumentException(e);
}
}
@Override
public int compare(WritableComparable a, WritableComparable b) {
if (a instanceof TextPair && b instanceof TextPair) {47
return ((TextPair) a).first.compareTo(((TextPair) b).first);
}
return super.compare(a, b);
}
} }
```

OUTPUT –

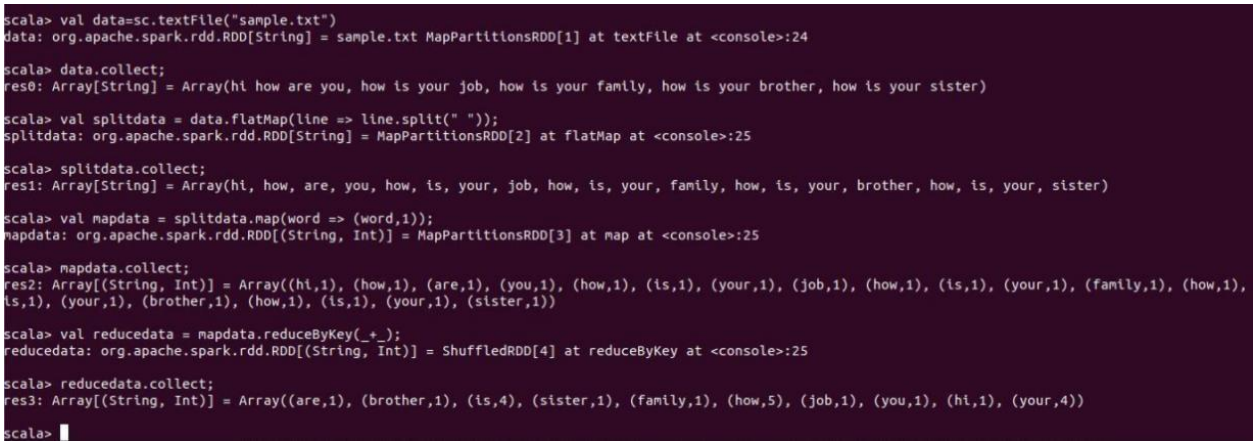
```
hduser@bmsce-Precision-T1700:/home/bmsce$ hdfs dfs -cat /join/output/*
A11     Finance         50
B12     HR              100
C13     Manufacturing           250
Dept_ID Dept_Name               Total_Employee
```

# LAB – 09

**AIM - Program to print word count on scala shell and print “Hello world” on scala IDE**

```
val data=sc.textFile("sparkdata.txt")
data.collect;
val splitdata = data.flatMap(line => line.split("
")); splitdata.collect;
val mapdata = splitdata.map(word => (word,1));
mapdata.collect;
val reducedata = mapdata.reduceByKey(_+_);
reducedata.collect;
```

**OUTPUT –**



# LAB – 10

**AIM - Using RDD and FlaMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark**

```
val textFile = sc.textFile("/home/bhoom/Desktop/wc.txt")
val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_+_)
import scala.collection.immutable.ListMap
val sorted=ListMap(counts.collect.sortWith(_._2 > _._2):_*)// sort in descending order based on values
println(sorted)
for((k,v)<-sorted) {
if(v>4) {
print(k+",")
print(v)
println() }
}
```

OUTPUT –

```
scala> val textFile = sc.textFile("sample.txt")
textFile: org.apache.spark.rdd.RDD[String] = sample.txt MapPartitionsRDD[1] at textFile at <console>:24

scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:25

scala> import scala.collection.immutable.ListMap
import scala.collection.immutable.ListMap

scala> val sorted=ListMap(counts.collect.sortWith(_._2 > _._2):_*)// sort in descending order based
sorted: scala.collection.immutable.ListMap[String,Int] = Map(how -> 5, is -> 4, your -> 4, are -> 1, brother -> 1, sister -> 1, family -> 1, job -> 1, you -> 1, hi
-> 1)

scala> println(sorted)
Map(how -> 5, is -> 4, your -> 4, are -> 1, brother -> 1, sister -> 1, family -> 1, job -> 1, you -> 1, hi -> 1)

scala> for((k,v)<-sorted)
     | {
     |
Display all 689 possibilities? (y or n)
     |
     | if(v>4)
     | {
     | print(k+",")
     | print(v)
     | println()
     | }
     | }
how,5
```