VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB REPORT
on

BIG DATA ANALYTICS (20CS6PEBDA)

Submitted by

PREMA(1BM19CS121)

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 **PREMA(1BM19CS121),** 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

CO 1	Apply the concept of NoSQL, Hadoop or Spark for a given task
CO 2	Analyze the Big Data and obtain insight using data analytics mechanisms.
CO 3	Design and implement Big data applications by applying NoSQL, Hadoop orSpark

LAB -2

Perform the following DB operations using Cassandra.

1. Create a keyspace by name Employee

```
create keyspace employee with replication = {
  ... 'class':'SimpleStrategy',
  ... 'replication_factor':1};
cqlsh> use employee;
```

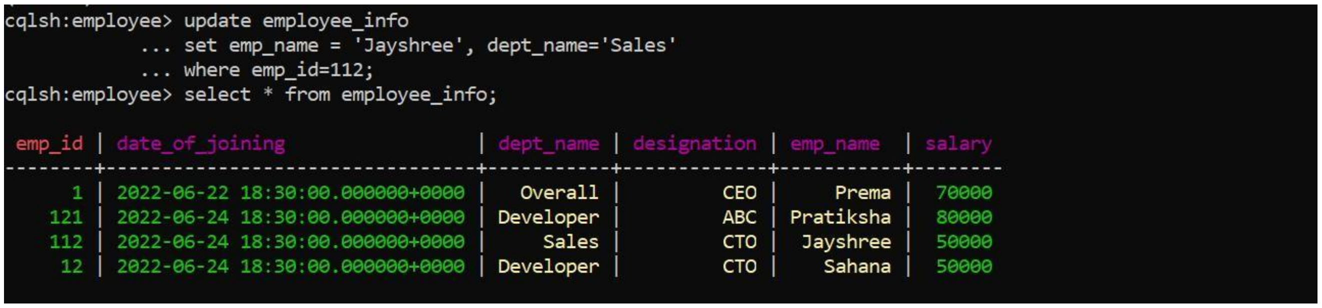
2. Create a column family by name Employee-Info with attributes Emp_Id PrimaryKey, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name

```
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));
```

3. Insert the values into the table in batch

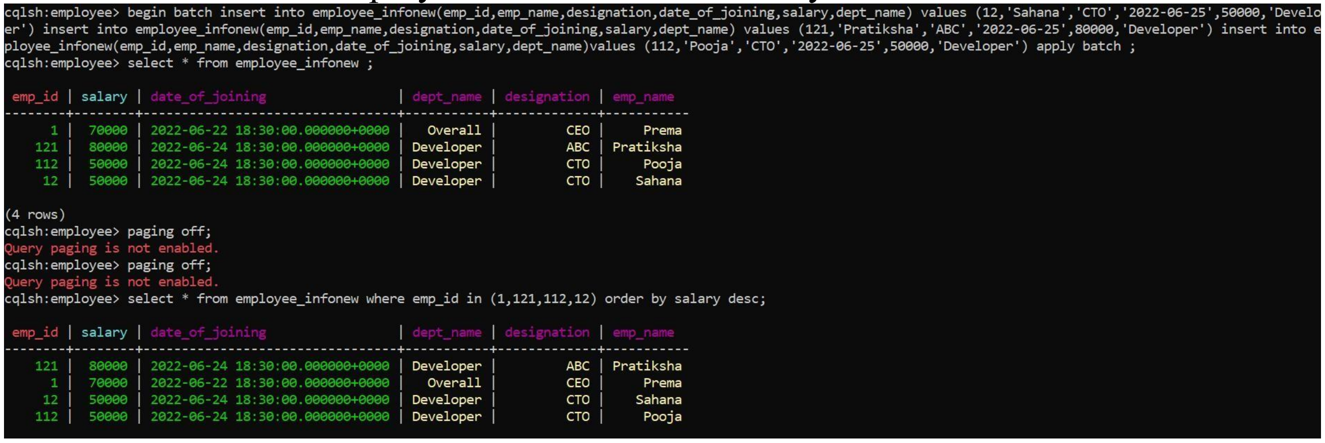
```
begin batch insert into
employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
values (1,'Prema','CEO','2022-06-23',70000,'Overall') insert into
employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)
values (12,'Sahana','CTO','2022-06-25',50000,'Developer') insert into employee_info(e
mp_id,emp_name,designation,date_of_joining,salary,dept_name) values
(121,'Pratiksha','ABC','2022-06-25',80000,'Developer') insert into
employee_info(emp_id,emp_name,designa
tion,date_of_joining,salary,dept_name)values (112,'Pooja','CTO','2022-06-
25',50000,'Developer') apply batch ;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name  | salary
--------+---------------------------------+-----------+-------------+-----------+--------
      1 | 2022-06-22 18:30:00.000000+|   Overall |         CEO |     Prema |  70000
    121 | 2022-06-24 18:30:00.000000+ | Developer |         ABC | Pratiksha |  80000
    112 | 2022-06-24 18:30:00.000000+| Developer |         CTO |     Pooja |  50000
     12 | 2022-06-24 18:30:00.000000+ | Developer |         CTO |    Sahana |  50000
```

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



```
cqlsh:employee> update employee_info
            ... set emp_name = 'Jayshree', dept_name='Sales'
            ... where emp_id=112;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name  | salary
--------+---------------------------------+-----------+-------------+-----------+--------
      1 | 2022-06-22 18:30:00.000000+0000 |   Overall |         CEO |     Prema |  70000
    121 | 2022-06-24 18:30:00.000000+0000 | Developer |         ABC | Pratiksha |  80000
    112 | 2022-06-24 18:30:00.000000+0000 |     Sales |         CTO |  Jayshree |  50000
     12 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |    Sahana |  50000
```

5. Sort the details of Employee records based on salary

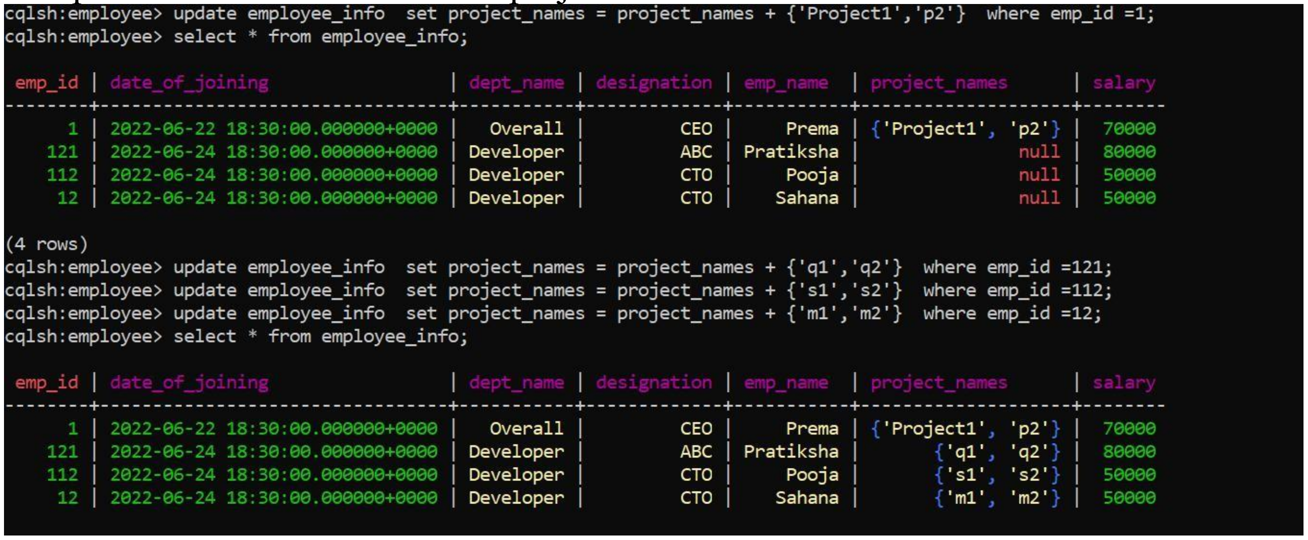


```
cqlsh:employee> begin batch insert into employee_infonew(emp_id,emp_name,designation,date_of_joining,salary,dept_name) values (12,'Sahana','CTO','2022-06-25',50000,'Develo
er') insert into employee_infonew(emp_id,emp_name,designation,date_of_joining,salary,dept_name) values (121,'Pratiksha','ABC','2022-06-25',80000,'Developer') insert into e
ployee_infonew(emp_id,emp_name,designation,date_of_joining,salary,dept_name)values (112,'Pooja','CTO','2022-06-25',50000,'Developer') apply batch ;
cqlsh:employee> select * from employee_infonew ;

 emp_id | salary | date_of_joining                 | dept_name | designation | emp_name
--------+--------+---------------------------------+-----------+-------------+-----------
      1 |  70000 | 2022-06-22 18:30:00.000000+0000 |   Overall |         CEO |     Prema
    121 |  80000 | 2022-06-24 18:30:00.000000+0000 | Developer |         ABC | Pratiksha
    112 |  50000 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |     Pooja
     12 |  50000 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |    Sahana

(4 rows)
cqlsh:employee> paging off;
Query paging is not enabled.
cqlsh:employee> paging off;
Query paging is not enabled.
cqlsh:employee> select * from employee_infonew where emp_id in (1,121,112,12) order by salary desc;

 emp_id | salary | date_of_joining                 | dept_name | designation | emp_name
--------+--------+---------------------------------+-----------+-------------+-----------
    121 |  80000 | 2022-06-24 18:30:00.000000+0000 | Developer |         ABC | Pratiksha
      1 |  70000 | 2022-06-22 18:30:00.000000+0000 |   Overall |         CEO |     Prema
     12 |  50000 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |    Sahana
    112 |  50000 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |     Pooja
```

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.

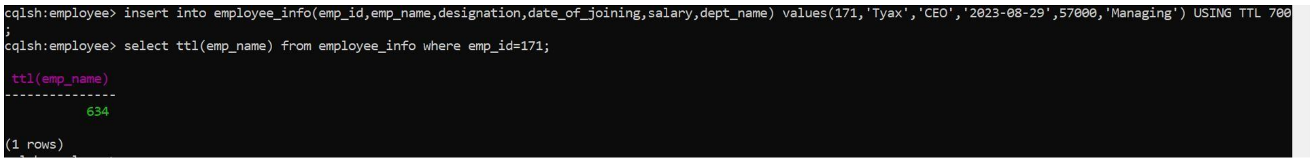
alter table employee_info add project_names set<text>;

7. Update the altered table to add project names.



```
cqlsh:employee> update employee_info  set project_names = project_names + {'Project1','p2'}  where emp_id =1;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name  | project_names     | salary
--------+---------------------------------+-----------+-------------+-----------+-------------------+--------
      1 | 2022-06-22 18:30:00.000000+0000 |   Overall |         CEO |     Prema | {'Project1', 'p2'} |  70000
    121 | 2022-06-24 18:30:00.000000+0000 | Developer |         ABC | Pratiksha |              null |  80000
    112 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |     Pooja |              null |  50000
     12 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |    Sahana |              null |  50000

(4 rows)
cqlsh:employee> update employee_info  set project_names = project_names + {'q1','q2'}  where emp_id =121;
cqlsh:employee> update employee_info  set project_names = project_names + {'s1','s2'}  where emp_id =112;
cqlsh:employee> update employee_info  set project_names = project_names + {'m1','m2'}  where emp_id =12;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name  | project_names     | salary
--------+---------------------------------+-----------+-------------+-----------+-------------------+--------
      1 | 2022-06-22 18:30:00.000000+0000 |   Overall |         CEO |     Prema | {'Project1', 'p2'} |  70000
    121 | 2022-06-24 18:30:00.000000+0000 | Developer |         ABC | Pratiksha |      {'q1', 'q2'} |  80000
    112 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |     Pooja |      {'s1', 's2'} |  50000
     12 | 2022-06-24 18:30:00.000000+0000 | Developer |         CTO |    Sahana |      {'m1', 'm2'} |  50000
```

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



```
cqlsh:employee> insert into employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name) values(171,'Tyax','CEO','2023-08-29',57000,'Managing') USING TTL 700
;
cqlsh:employee> select ttl(emp_name) from employee_info where emp_id=171;

 ttl(emp_name)
---------------
           634

(1 rows)
```

LAB -3

3. Perform the following DB operations using Cassandra.

1. Create a keyspace by name Library

```
CREATE KEYSPACE LIBRARY1 WITH REPLICATION = {
   ... 'class':'SimpleStrategy',
   ... 'replication_factor':1};
```

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

```
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));
```

3. Insert the values into the table in batch

```
update library_info
          ... set counter_value = counter_value +1 where stud_id=121 and
stud_name='Prema' and book_name='cns' and book_id=113 and
date_of_issue='2022-06-29';
select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                   |
counter_value
---------+-----------+-----------+---------+---------------------------------+--------------
     121 |     Prema |       cns |     113 | 2022-06-28 18:30:00.000000+0000 |            1
```

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

```
 update library_info set counter_value = counter_value +1 where stud_id=121 and
stud_name='Prema' and book_name='cns' and book_id=113 and
date_of_issue='2022-06-29';
cqlsh:library1> select * from library_info;
```

```
 stud_id | stud_name | book_name | book_id | date_of_issue                    |
counter_value
-----------+------------+-------------+-----------+------------------------------------+--------------------
    121 |     Prema |       cns |     113 | 2022-06-28 18:30:00.000000+0000 |             2
```

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

```
cqlsh:library1> update library_info set counter_value = counter_value +2 where
stud_id=111 and stud_name='Pooja' and book_name='bda' and book_id=112 and
date_of_issue='202
2-06-29';
select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                    |
counter_value
-----------+------------+-------------+-----------+------------------------------------+--------------------
    111 |      Pooja |        bda |    112 | 2022-06-28 18:30:00.000000+0000 |             2
    121 |      Prema |        cns |     113 | 2022-06-28 18:30:00.000000+0000 |             2
```

6. Export the created column to a csv file

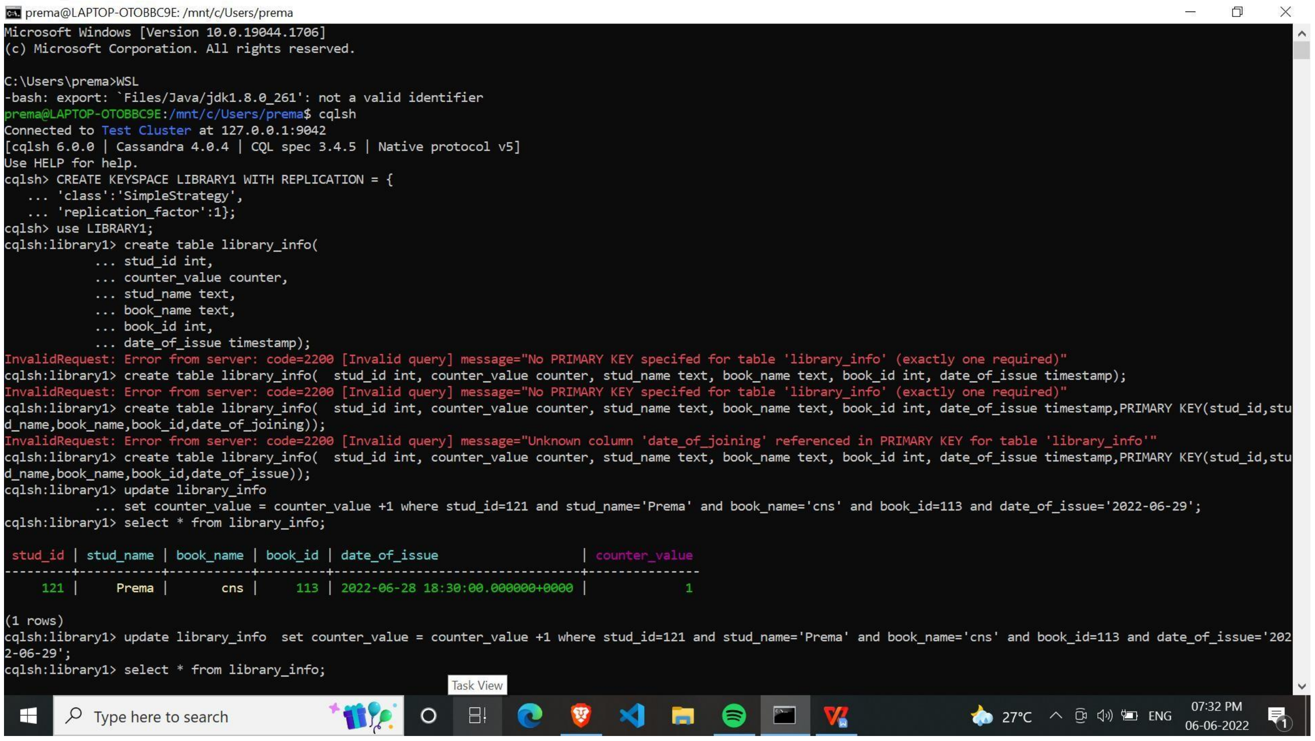
```
COPY
library_info(stud_id,counter_value,stud_name,book_name,book_id,date_of_issue) TO
'lib1.csv'
         ... ;
Using 7 child processes

Starting copy of library1.library_info with columns [stud_id, counter_value,
stud_name, book_name, book_id, date_of_issue].
Processed: 2 rows; Rate:      17 rows/s; Avg. rate:      17 rows/s
2 rows exported to 1 files in 0.143 seconds.
```

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

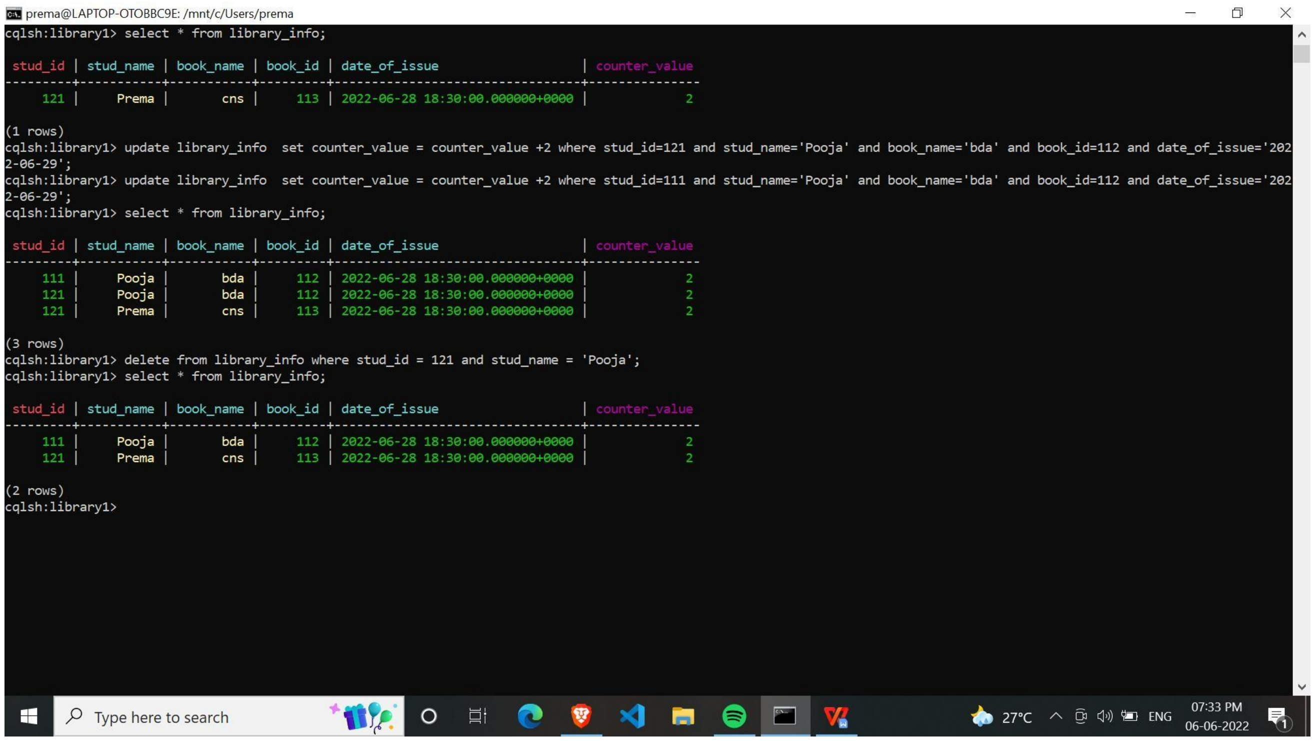
```
familyTRUNCATE library_info;
cqlsh:library1> select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue | counter_value
```

```
----------+------------+-------------+-----------+----------------+-------------------

(0 rows)
cqlsh:library1>
COPY
library_info(stud_id,counter_value,stud_name,book_name,book_id,date_of_issue)
FROM 'lib1.csv' ;
Using 7 child processes

Starting copy of library1.library_info with columns [stud_id, counter_value,
stud_name, book_name, book_id, date_of_issue].
Processed: 2 rows; Rate:      4 rows/s; Avg. rate:      6 rows/s
2 rows imported from 1 files in 0.364 seconds (0 skipped).
cqlsh:library1> select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                   |
counter_value
----------+------------+-------------+-----------+-----------------------------------+-------------------
     111 |      Pooja |       bda |      112 | 2022-06-28 18:30:00.000000+0000 |             2
     121 |      Prema |        cns |      113 | 2022-06-28 18:30:00.000000+0000 |             2
```

Output screenshots:



```
prema@LAPTOP-OTOBBC9E: /mnt/c/Users/prema
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\prema>WSL
-bash: export: `Files/Java/jdk1.8.0_261': not a valid identifier
prema@LAPTOP-OTOBBC9E:/mnt/c/Users/prema$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.0.0 | Cassandra 4.0.4 | CQL spec 3.4.5 | Native protocol v5]
Use HELP for help.
cqlsh> CREATE KEYSPACE LIBRARY1 WITH REPLICATION = {
   ... 'class':'SimpleStrategy',
   ... 'replication_factor':1};
cqlsh> use LIBRARY1;
cqlsh:library1> create table library_info(
            ... stud_id int,
            ... counter_value counter,
            ... stud_name text,
            ... book_name text,
            ... book_id int,
            ... date_of_issue timestamp);
InvalidRequest: Error from server: code=2200 [Invalid query] message="No PRIMARY KEY specifed for table 'library_info' (exactly one required)"
cqlsh:library1> create table library_info(  stud_id int, counter_value counter, stud_name text, book_name text, book_id int, date_of_issue timestamp);
InvalidRequest: Error from server: code=2200 [Invalid query] message="No PRIMARY KEY specifed for table 'library_info' (exactly one required)"
cqlsh:library1> 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,stu
d_name,book_name,book_id,date_of_joining));
InvalidRequest: Error from server: code=2200 [Invalid query] message="Unknown column 'date_of_joining' referenced in PRIMARY KEY for table 'library_info'"
cqlsh:library1> 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,stu
d_name,book_name,book_id,date_of_issue));
cqlsh:library1> update library_info
            ... set counter_value = counter_value +1 where stud_id=121 and stud_name='Prema' and book_name='cns' and book_id=113 and date_of_issue='2022-06-29';
cqlsh:library1> select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                   | counter_value
---------+-----------+-----------+---------+---------------------------------+---------------
     121 |     Prema |       cns |     113 | 2022-06-28 18:30:00.000000+0000 |             1

(1 rows)
cqlsh:library1> update library_info  set counter_value = counter_value +1 where stud_id=121 and stud_name='Prema' and book_name='cns' and book_id=113 and date_of_issue='202
2-06-29';
cqlsh:library1> select * from library_info;
```

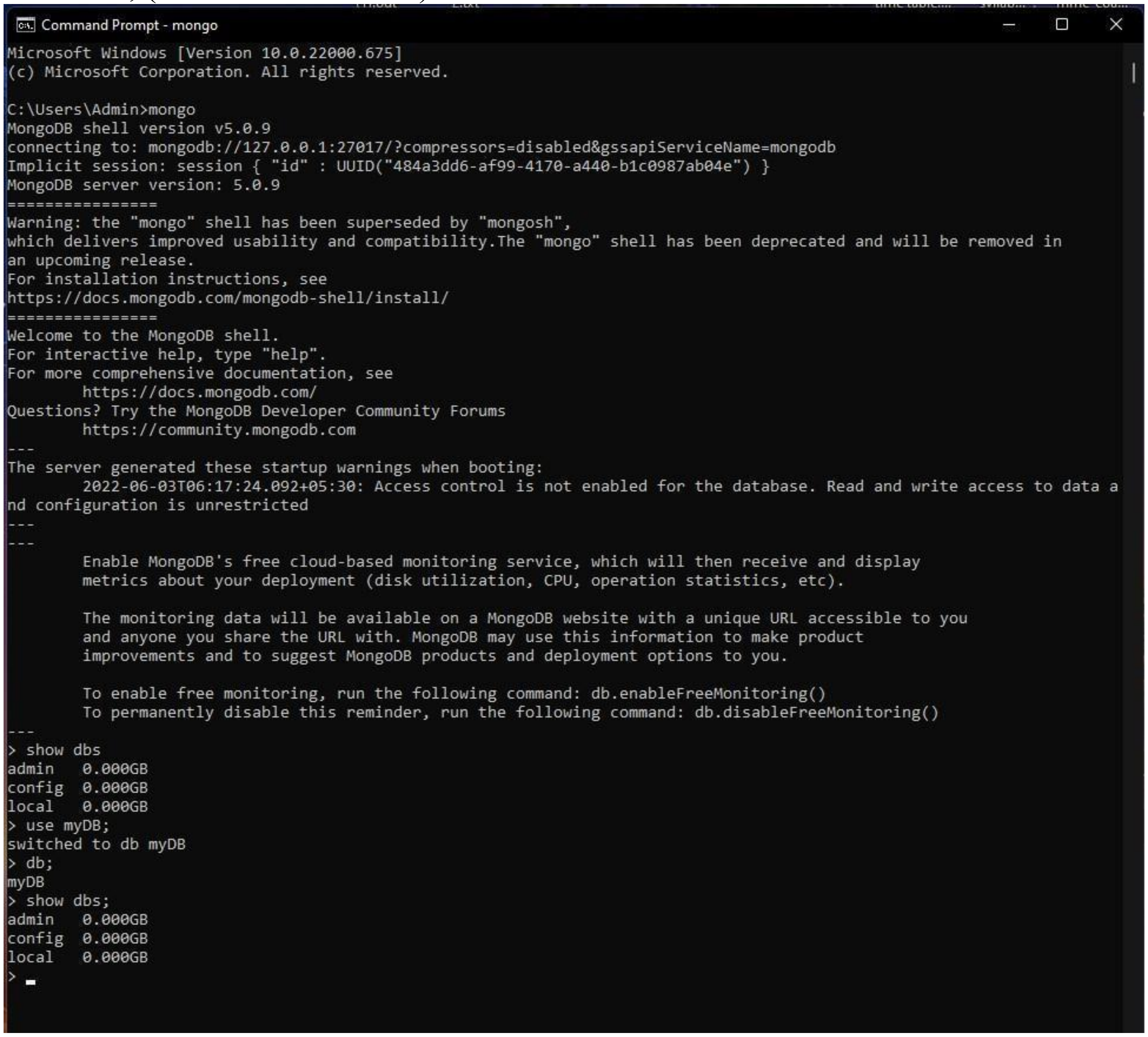


prema@LAPTOP-OTOBBC9E: /mnt/c/Users/prema

```
cqlsh:library1> select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                   | counter_value
---------+-----------+-----------+---------+---------------------------------+---------------
     121 |     Prema |       cns |     113 | 2022-06-28 18:30:00.000000+0000 |             2

(1 rows)
cqlsh:library1> update library_info  set counter_value = counter_value +2 where stud_id=121 and stud_name='Pooja' and book_name='bda' and book_id=112 and date_of_issue='202
2-06-29';
cqlsh:library1> update library_info  set counter_value = counter_value +2 where stud_id=111 and stud_name='Pooja' and book_name='bda' and book_id=112 and date_of_issue='202
2-06-29';
cqlsh:library1> select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                   | counter_value
---------+-----------+-----------+---------+---------------------------------+---------------
     111 |     Pooja |       bda |     112 | 2022-06-28 18:30:00.000000+0000 |             2
     121 |     Pooja |       bda |     112 | 2022-06-28 18:30:00.000000+0000 |             2
     121 |     Prema |       cns |     113 | 2022-06-28 18:30:00.000000+0000 |             2

(3 rows)
cqlsh:library1> delete from library_info where stud_id = 121 and stud_name = 'Pooja';
cqlsh:library1> select * from library_info;

 stud_id | stud_name | book_name | book_id | date_of_issue                   | counter_value
---------+-----------+-----------+---------+---------------------------------+---------------
     111 |     Pooja |       bda |     112 | 2022-06-28 18:30:00.000000+0000 |             2
     121 |     Prema |       cns |     113 | 2022-06-28 18:30:00.000000+0000 |             2

(2 rows)
cqlsh:library1>
```

LAB-1
Mongo db CRUD demonstration:

I. CREATE DATABASE IN
MONGODB.use myDB; db;
(Confirm the existence of your
database)
show dbs; (To list all databases)



```
Microsoft Windows [Version 10.0.22000.675]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>mongo
MongoDB shell version v5.0.9
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("484a3dd6-af99-4170-a440-b1c0987ab04e") }
MongoDB server version: 5.0.9
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
        https://community.mongodb.com
---
The server generated these startup warnings when booting:
        2022-06-03T06:17:24.092+05:30: Access control is not enabled for the database. Read and write access to data a
nd configuration is unrestricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use myDB;
switched to db myDB
> db;
myDB
> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

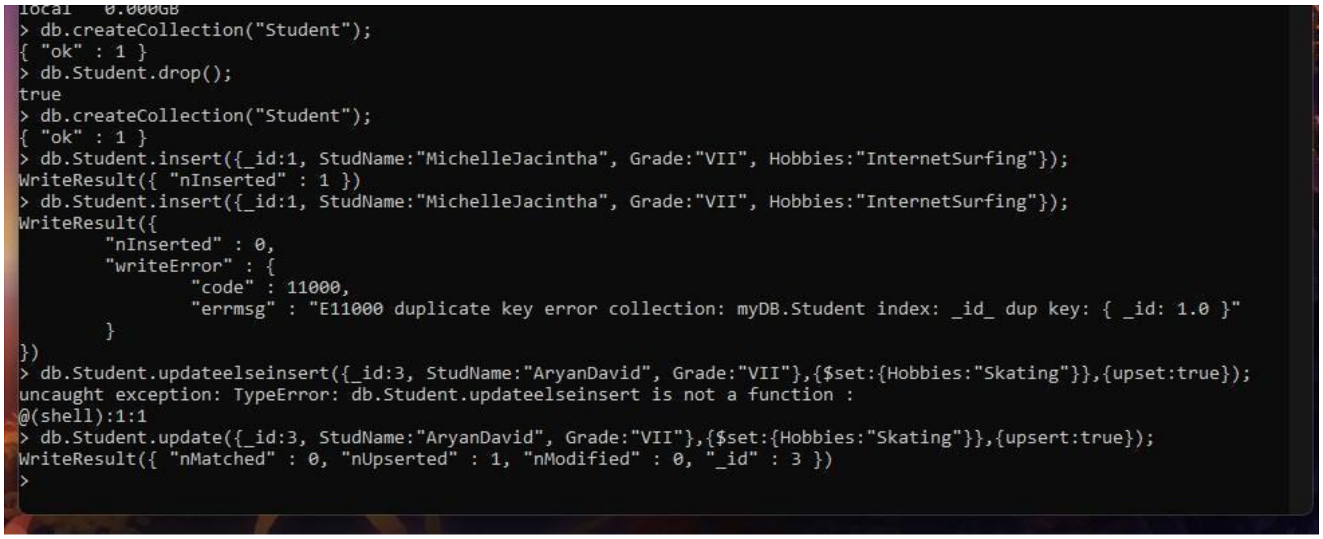
II. CRUD (CREATE, READ, UPDATE, DELETE) OPERATIONS

1. To create a collection by the name “Student”. Let us take a look at the collection list prior to the creation of the new collection “Student”.
db.createCollection(“Student”); => sql equivalent CREATE TABLE STUDENT(…);

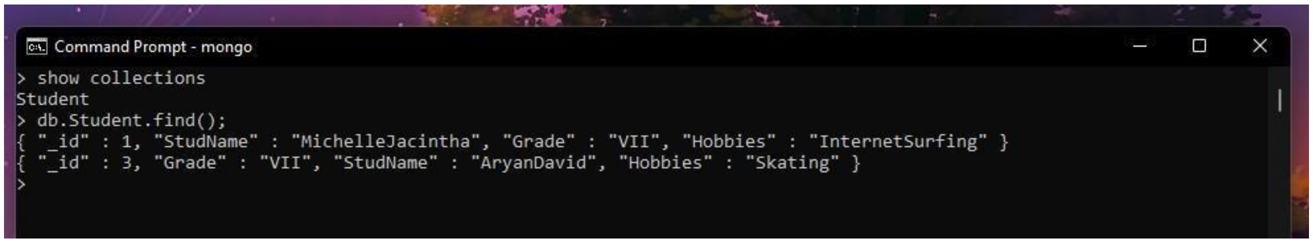
2. To drop a collection by the name “Student”.
db.Student.drop();

3. Create a collection by the name “Students” and store the following data in it.
db.Student.insert({_id:1,StudName:"MichelleJacintha",Grade:"VII",Hobbies:&quot;Int ernetS urfing"});

4. Insert the document for “AryanDavid” in to the Students collection only if it does not already exist in the collection. However, if it is already present in the collection, then update the document with new values. (Update his Hobbies from “Skating” to “Chess”.) Use “Update else insert” (if there is an existing document, it will attempt to update it, if there is no existing document then it will insert it).
db.Student.update({_id:3,StudName:"AryanDavid",Grade:"VII"},{$set:{Hobbies:&quo t;Skatin g"}},{upsert:true});



```
local   0.000GB
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.drop();
true
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.insert({_id:1, StudName:"MichelleJacintha", Grade:"VII", Hobbies:"InternetSurfing"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:1, StudName:"MichelleJacintha", Grade:"VII", Hobbies:"InternetSurfing"});
WriteResult({
        "nInserted" : 0,
        "writeError" : {
                "code" : 11000,
                "errmsg" : "E11000 duplicate key error collection: myDB.Student index: _id_ dup key: { _id: 1.0 }"
        }
})
> db.Student.updateelseinsert({_id:3, StudName:"AryanDavid", Grade:"VII"},{$set:{Hobbies:"Skating"}},{upset:true});
uncaught exception: TypeError: db.Student.updateelseinsert is not a function :
@(shell):1:1
> db.Student.update({_id:3, StudName:"AryanDavid", Grade:"VII"},{$set:{Hobbies:"Skating"}},{upsert:true});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 3 })
>
```



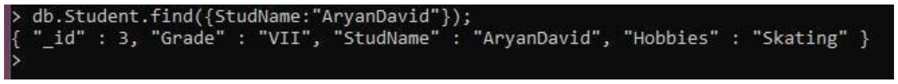
Command Prompt - mongo

```
> show collections
Student
> db.Student.find();
{ "_id" : 1, "StudName" : "MichelleJacintha", "Grade" : "VII", "Hobbies" : "InternetSurfing" }
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : "Skating" }
>
```

5. FIND METHOD

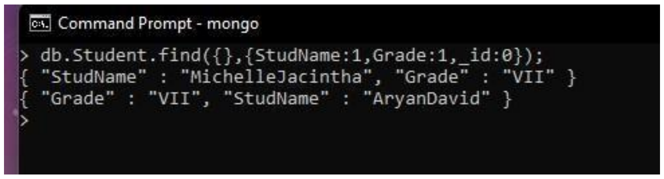
A.

To search for documents from the “Students” collection based on certain search criteria.
db.Student.find({StudName:"Aryan David"});
({cond..},{columns.. column:1, columnname:0})



```
> db.Student.find({StudName:"AryanDavid"});
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : "Skating" }
>
```

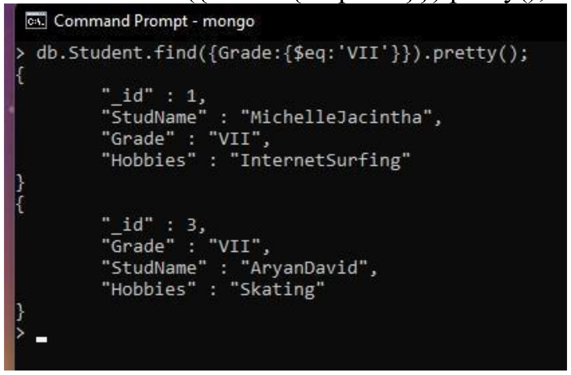
B.
To display only the StudName and Grade from all the documents of the Students collection. The identifier_id should be suppressed and NOT displayed.
db.Student.find({},{StudName:1,Grade:1,_id:0});



Command Prompt - mongo

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

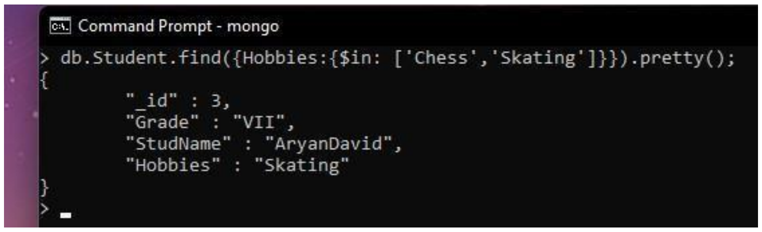
C.
To find those documents where the Grade is set to 'VII'
db.Student.find({Grade:{$eq:'VII'}}).pretty();



Command Prompt - mongo

```
> db.Student.find({Grade:{$eq:'VII'}}).pretty();
{
        "_id" : 1,
        "StudName" : "MichelleJacintha",
        "Grade" : "VII",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
>
```

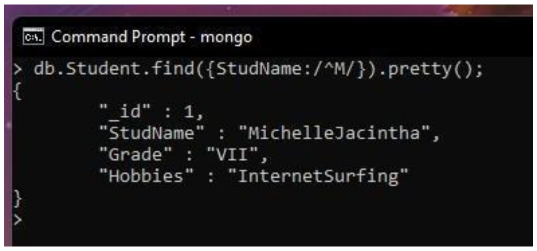
D.
To find those documents from the Students collection where the Hobbies is set to either 'Chess' or is set to 'Skating'. db.Student.find({Hobbies :{ $in: ['Chess','Skating']}}).pretty ();



Command Prompt - mongo

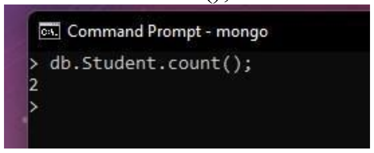
```
> db.Student.find({Hobbies:{$in: ['Chess','Skating']}}).pretty();
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
>
```

E.
To find documents from the Students collection where the StudName begins with "M". db.Student.find({StudName:/^M/}).pretty();



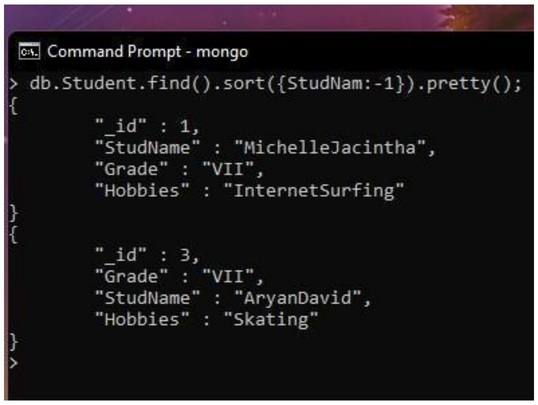
```
> db.Student.find({StudName:/^M/}).pretty();
{
        "_id" : 1,
        "StudName" : "MichelleJacintha",
        "Grade" : "VII",
        "Hobbies" : "InternetSurfing"
}
>
```

G.
To find the number of documents in the Students collection.
db.Student.count();



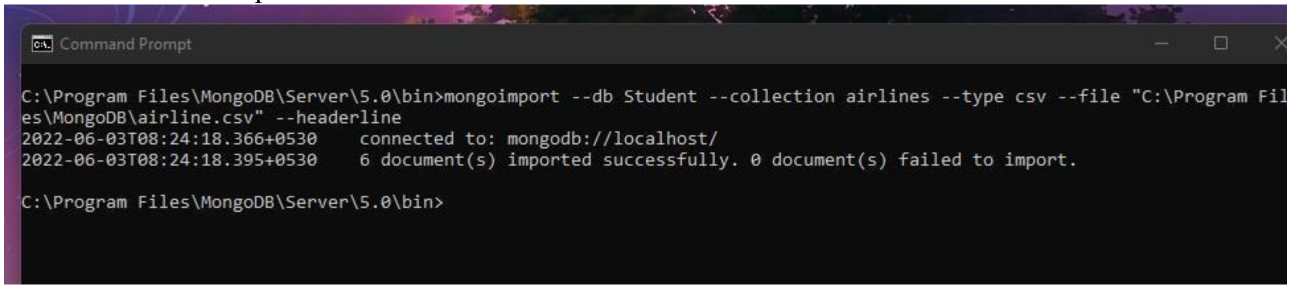
```
> db.Student.count();
2
>
```

H.
To sort the documents from the Students collection in the descending order of StudName. db.Student.find().sort({StudName:-1}).pretty();



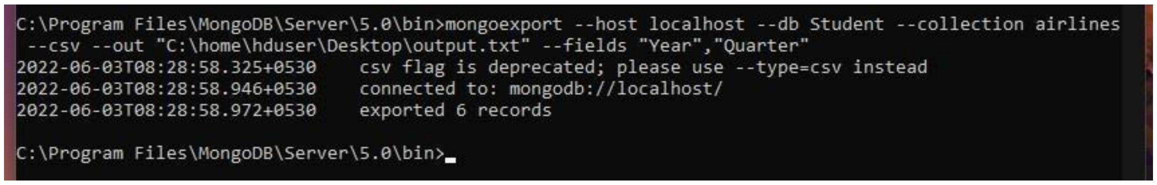
```
> db.Student.find().sort({StudNam:-1}).pretty();
{
        "_id" : 1,
        "StudName" : "MichelleJacintha",
        "Grade" : "VII",
        "Hobbies" : "InternetSurfing"
}
{
        "_id" : 3,
        "Grade" : "VII",
        "StudName" : "AryanDavid",
        "Hobbies" : "Skating"
}
>
```

III. Import data from a CSV file
Given a CSV file "sample.txt" in the D:drive, import the file into the MongoDB collection, "SampleJSON". The collection is in the database "test".
mongoimport --db Student --collection airlines --type csv –headerline --file /home/hduser/Desktop/airline.csv



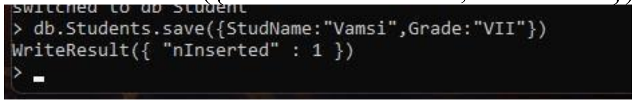
```
C:\Program Files\MongoDB\Server\5.0\bin>mongoimport --db Student --collection airlines --type csv --file "C:\Program Fil
es\MongoDB\airline.csv" --headerline
2022-06-03T08:24:18.366+0530    connected to: mongodb://localhost/
2022-06-03T08:24:18.395+0530    6 document(s) imported successfully. 0 document(s) failed to import.

C:\Program Files\MongoDB\Server\5.0\bin>
```

IV. Export data to a CSV file
This command used at the command prompt exports MongoDB JSON documents from
“Customers” collection in the “test” database into a CSV file “Output.txt” in the D:drive.
mongoexport --host localhost --db Student --collection airlines --csv --out /home/hduser/Desktop/output.txt –fields “Year”,”Quarter”



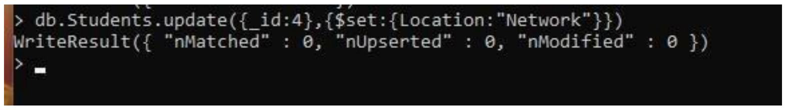
```
C:\Program Files\MongoDB\Server\5.0\bin>mongoexport --host localhost --db Student --collection airlines
 --csv --out "C:\home\hduser\Desktop\output.txt" --fields "Year","Quarter"
2022-06-03T08:28:58.325+0530    csv flag is deprecated; please use --type=csv instead
2022-06-03T08:28:58.946+0530    connected to: mongodb://localhost/
2022-06-03T08:28:58.972+0530    exported 6 records

C:\Program Files\MongoDB\Server\5.0\bin>
```

V. Save Method :
Save() method will insert a new document, if the document with the _id does not exist. If it exists it will replace the exisiting document.
db.Students.save({StudName:”Vamsi”, Grade:”VI”})



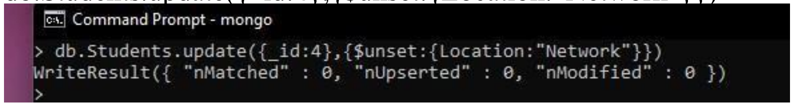
```
switched to db Student
> db.Students.save({StudName:"Vamsi",Grade:"VII"})
WriteResult({ "nInserted" : 1 })
>
```

VI. Add a new field to existing Document:
db.Students.update({_id:4},{$set:{Location:”Network”}})



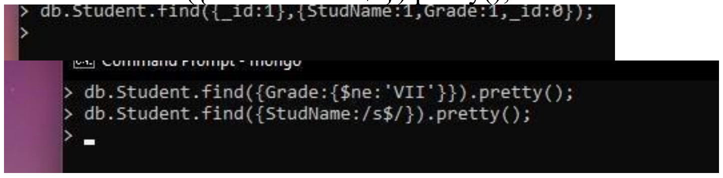
```
> db.Students.update({_id:4},{$set:{Location:"Network"}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

VII. Remove the field in an existing Document
db.Students.update({_id:4},{$unset:{Location:”Network”}})



```
Command Prompt - mongo
> db.Students.update({_id:4},{$unset:{Location:"Network"}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

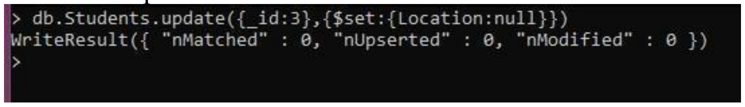
VIII. Finding Document based on search criteria suppressing few fields
db.Student.find({_id:1},{StudName:1,Grade:1,_id:0});
To find those documents where the Grade is not set to ‘VII’
db.Student.find({Grade:{$ne:'VII'}}).pretty();
To find documents from the Students collection where the StudName ends with s.
db.Student.find({StudName:/s$/}).pretty();



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

```
Command Prompt - mongo
> db.Student.find({Grade:{$ne:'VII'}}).pretty();
> db.Student.find({StudName:/s$/}).pretty();
>
```

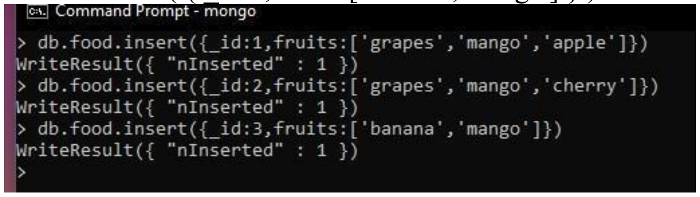
IX. to set a particular field value to NULL



```
> db.Students.update({_id:3},{$set:{Location:null}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

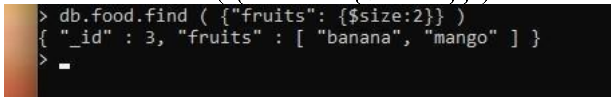
XII. Create a collection by name “food” and add to each document add a “fruits” array db.food.insert({ _id:1,
fruits:['grapes','mango','apple'] })
db.food.insert({ _id:2,

fruits:['grapes','mango','cherry'] })
db.food.insert({ _id:3, fruits:['banana','mango'] })



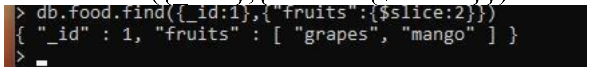
```
Command Prompt - mongo
> db.food.insert({_id:1,fruits:['grapes','mango','apple']})
WriteResult({ "nInserted" : 1 })
> db.food.insert({_id:2,fruits:['grapes','mango','cherry']})
WriteResult({ "nInserted" : 1 })
> db.food.insert({_id:3,fruits:['banana','mango']})
WriteResult({ "nInserted" : 1 })
>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

To find those documents from the "food" collection where the size of the array is two. db.food.find ({"fruits": {$size:2}})



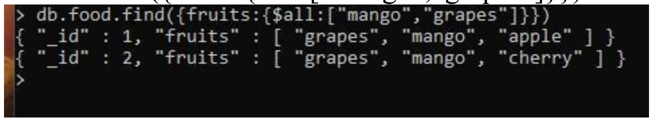
```
> db.food.find ( {"fruits": {$size:2}} )
{ "_id" : 3, "fruits" : [ "banana", "mango" ] }
>
```

To find the document with a particular id and display the first two elements from the array "fruits"
db.food.find({_id:1},{"fruits":{$slice:2}})



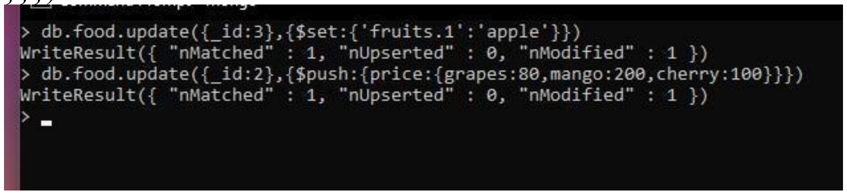
```
> db.food.find({_id:1},{"fruits":{$slice:2}})
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }
>
```

To find all the documets from the food collection which have elements mango and grapes in the array "fruits"
db.food.find({fruits:{$all:["mango","grapes"]}})



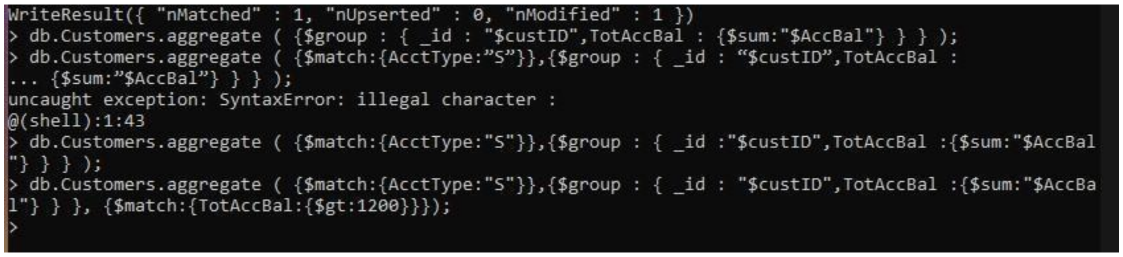
```
> db.food.find({fruits:{$all:["mango","grapes"]}})
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }
>
```

update on Array: using particular id replace the element present in the 1 st index position of the fruits array with apple
db.food.update({_id:3},{$set:{'fruits.1':'apple'}})
insert new key value pairs in the fruits array
db.food.update({_id:2},{$push:{price:{grapes:80,mango:200,cherry:100 }}})



```
> 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:100}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

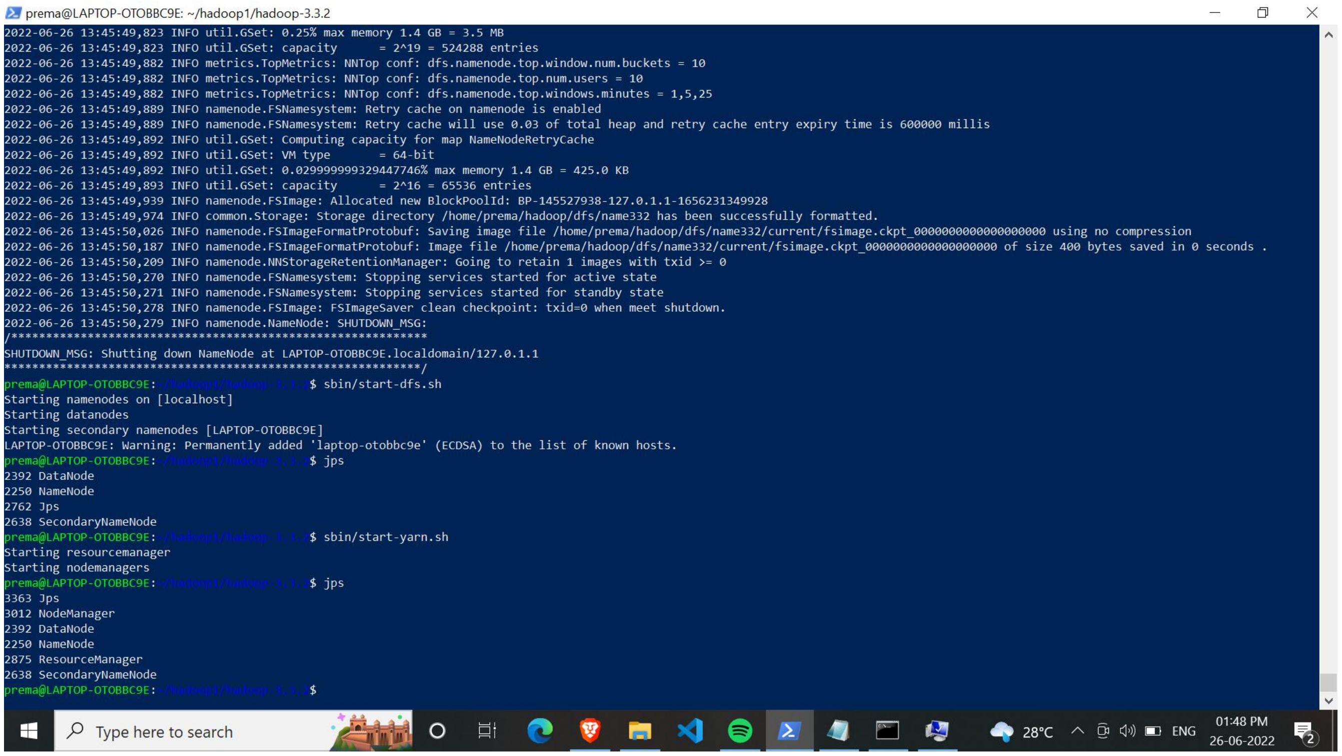
XII. Aggregate Function :
Create a collection Customers with fields custID, AcctBal, AcctType.
Now group on "custID" and compute the sum of "AccBal". db.Customers.aggregate ({$group : { _id : "$custID",TotAccBal : {$sum:"$AccBal"} } }); match on AcctType:"S" then group on "CustID" and compute the sum of "AccBal".
db.Customers.aggregate ({$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :
{$sum:"$AccBal"} } });
match on AcctType:"S" then group on "CustID" and compute the sum of "AccBal" and total balance greater than 1200.
db.Customers.aggregate ({$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :
{$sum:"$AccBal"} } }, {$match:{TotAccBal:{$gt:1200}}});



```
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Customers.aggregate ( {$group : { _id : "$custID",TotAccBal : {$sum:"$AccBal"} } } );
> db.Customers.aggregate ( {$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :
... {$sum:"$AccBal"} } } );
uncaught exception: SyntaxError: illegal character :
@(shell):1:43
> db.Customers.aggregate ( {$match:{AcctType:"S"}},{$group : { _id :"$custID",TotAccBal :{$sum:"$AccBal
"} } } );
> db.Customers.aggregate ( {$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :{$sum:"$AccBa
l"} } }, {$match:{TotAccBal:{$gt:1200}}});
>
```

LAB 4

Screenshot of Hadoop installed



prema@LAPTOP-OTOBBC9E: ~/hadoop1/hadoop-3.3.2

```
2022-06-26 13:45:49,823 INFO util.GSet: 0.25% max memory 1.4 GB = 3.5 MB
2022-06-26 13:45:49,823 INFO util.GSet: capacity      = 2^19 = 524288 entries
2022-06-26 13:45:49,882 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
2022-06-26 13:45:49,882 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
2022-06-26 13:45:49,882 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
2022-06-26 13:45:49,889 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
2022-06-26 13:45:49,889 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
2022-06-26 13:45:49,892 INFO util.GSet: Computing capacity for map NameNodeRetryCache
2022-06-26 13:45:49,892 INFO util.GSet: VM type       = 64-bit
2022-06-26 13:45:49,892 INFO util.GSet: 0.029999999329447746% max memory 1.4 GB = 425.0 KB
2022-06-26 13:45:49,893 INFO util.GSet: capacity      = 2^16 = 65536 entries
2022-06-26 13:45:49,939 INFO namenode.FSImage: Allocated new BlockPoolId: BP-145527938-127.0.1.1-1656231349928
2022-06-26 13:45:49,974 INFO common.Storage: Storage directory /home/prema/hadoop/dfs/name332 has been successfully formatted.
2022-06-26 13:45:50,026 INFO namenode.FSImageFormatProtobuf: Saving image file /home/prema/hadoop/dfs/name332/current/fsimage.ckpt_0000000000000000000 using no compression
2022-06-26 13:45:50,187 INFO namenode.FSImageFormatProtobuf: Image file /home/prema/hadoop/dfs/name332/current/fsimage.ckpt_0000000000000000000 of size 400 bytes saved in 0 seconds .
2022-06-26 13:45:50,209 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
2022-06-26 13:45:50,270 INFO namenode.FSNamesystem: Stopping services started for active state
2022-06-26 13:45:50,271 INFO namenode.FSNamesystem: Stopping services started for standby state
2022-06-26 13:45:50,278 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when meet shutdown.
2022-06-26 13:45:50,279 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at LAPTOP-OTOBBC9E.localdomain/127.0.1.1
************************************************************/
prema@LAPTOP-OTOBBC9E:~/hadoop1/hadoop-3.3.2$ sbin/start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [LAPTOP-OTOBBC9E]
LAPTOP-OTOBBC9E: Warning: Permanently added 'laptop-otobbc9e' (ECDSA) to the list of known hosts.
prema@LAPTOP-OTOBBC9E:~/hadoop1/hadoop-3.3.2$ jps
2392 DataNode
2250 NameNode
2762 Jps
2638 SecondaryNameNode
prema@LAPTOP-OTOBBC9E:~/hadoop1/hadoop-3.3.2$ sbin/start-yarn.sh
Starting resourcemanager
Starting nodemanagers
prema@LAPTOP-OTOBBC9E:~/hadoop1/hadoop-3.3.2$ jps
3363 Jps
3012 NodeManager
2392 DataNode
2250 NameNode
2875 ResourceManager
2638 SecondaryNameNode
prema@LAPTOP-OTOBBC9E:~/hadoop1/hadoop-3.3.2$
```

LAB 5
6. From the following link extract the weather data
https://github.com/tomwhite/hadoop-book/tree/master/input/ncdc/all
Create a Map Reduce program to
a) find average temperature for each year from NCDC data set.
b) find the mean max temperature for every month

```
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);
    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;
```

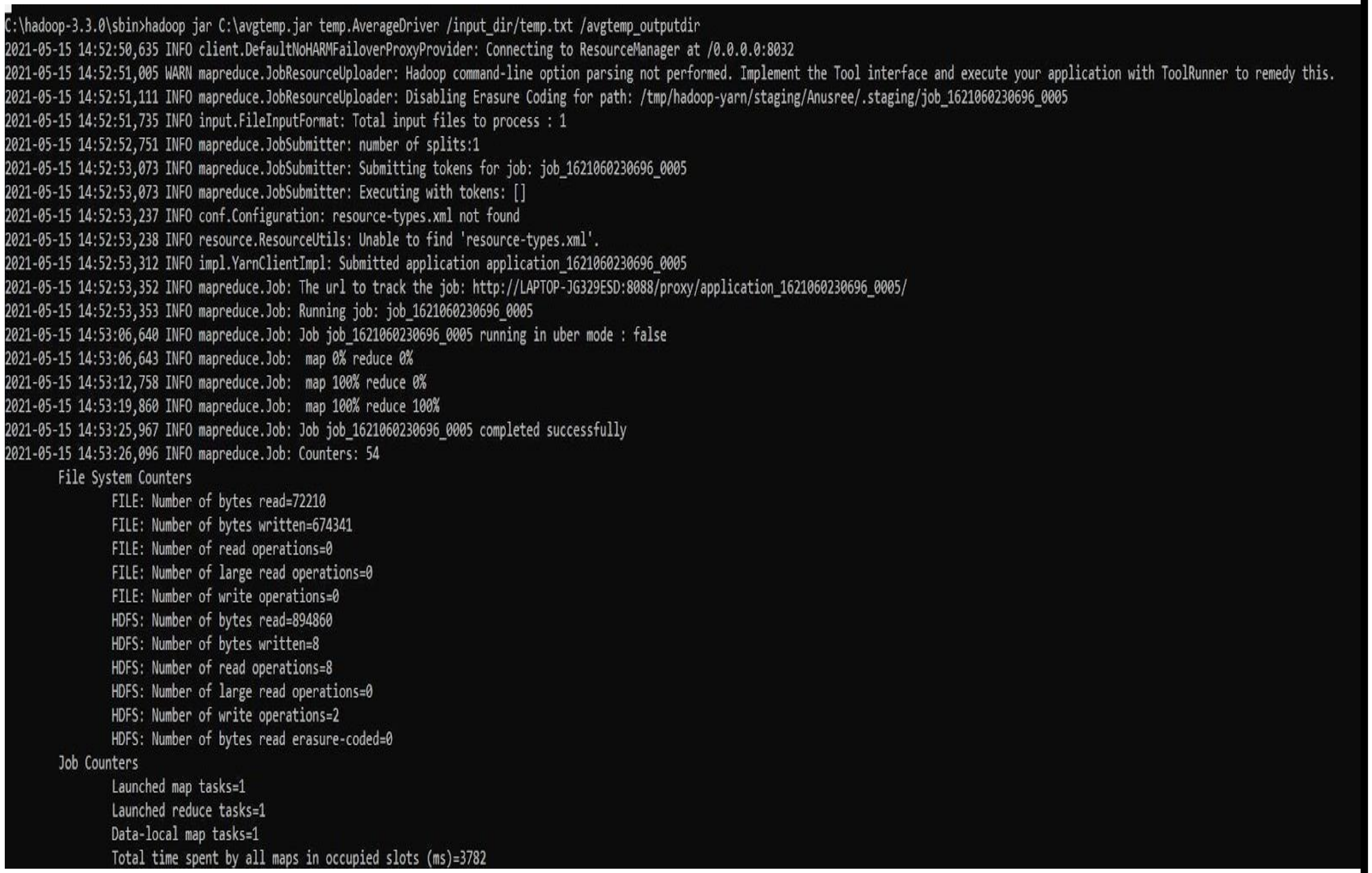
```
public class AverageMapper extends Mapper<LongWritable, Text,
Text, IntWritable> {
  public static final int MISSING = 9999;

  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);
    if (temperature != 9999 && quality.matches("[01459]"))
      context.write(new Text(year), new
IntWritable(temperature));
  }
}
```

AverageReducer

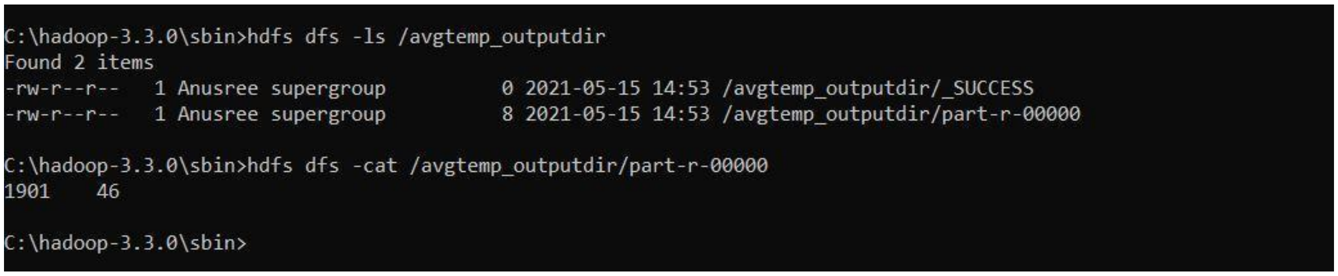
```
package temp;


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));
    }
}
```

SCREENSHOTS:



```
C:\hadoop-3.3.0\sbin>hadoop jar C:\avgtemp.jar temp.AverageDriver /input_dir/temp.txt /avgtemp_outputdir
2021-05-15 14:52:50,635 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-05-15 14:52:51,005 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2021-05-15 14:52:51,111 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1621060230696_0005
2021-05-15 14:52:51,735 INFO input.FileInputFormat: Total input files to process : 1
2021-05-15 14:52:52,751 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-15 14:52:53,073 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1621060230696_0005
2021-05-15 14:52:53,073 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-15 14:52:53,237 INFO conf.Configuration: resource-types.xml not found
2021-05-15 14:52:53,238 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-15 14:52:53,312 INFO impl.YarnClientImpl: Submitted application application_1621060230696_0005
2021-05-15 14:52:53,352 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1621060230696_0005/
2021-05-15 14:52:53,353 INFO mapreduce.Job: Running job: job_1621060230696_0005
2021-05-15 14:53:06,640 INFO mapreduce.Job: Job job_1621060230696_0005 running in uber mode : false
2021-05-15 14:53:06,643 INFO mapreduce.Job:  map 0% reduce 0%
2021-05-15 14:53:12,758 INFO mapreduce.Job:  map 100% reduce 0%
2021-05-15 14:53:19,860 INFO mapreduce.Job:  map 100% reduce 100%
2021-05-15 14:53:25,967 INFO mapreduce.Job: Job job_1621060230696_0005 completed successfully
2021-05-15 14:53:26,096 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=72210
                FILE: Number of bytes written=674341
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=894860
                HDFS: Number of bytes written=8
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=3782
```



```
C:\hadoop-3.3.0\sbin>hdfs dfs -ls /avgtemp_outputdir
Found 2 items
-rw-r--r--   1 Anusree supergroup          0 2021-05-15 14:53 /avgtemp_outputdir/_SUCCESS
-rw-r--r--   1 Anusree supergroup          8 2021-05-15 14:53 /avgtemp_outputdir/part-r-00000

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /avgtemp_outputdir/part-r-00000
1901    46

C:\hadoop-3.3.0\sbin>
```

b) find the mean max temperature for every month

MeanMax

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);
  }
}
```

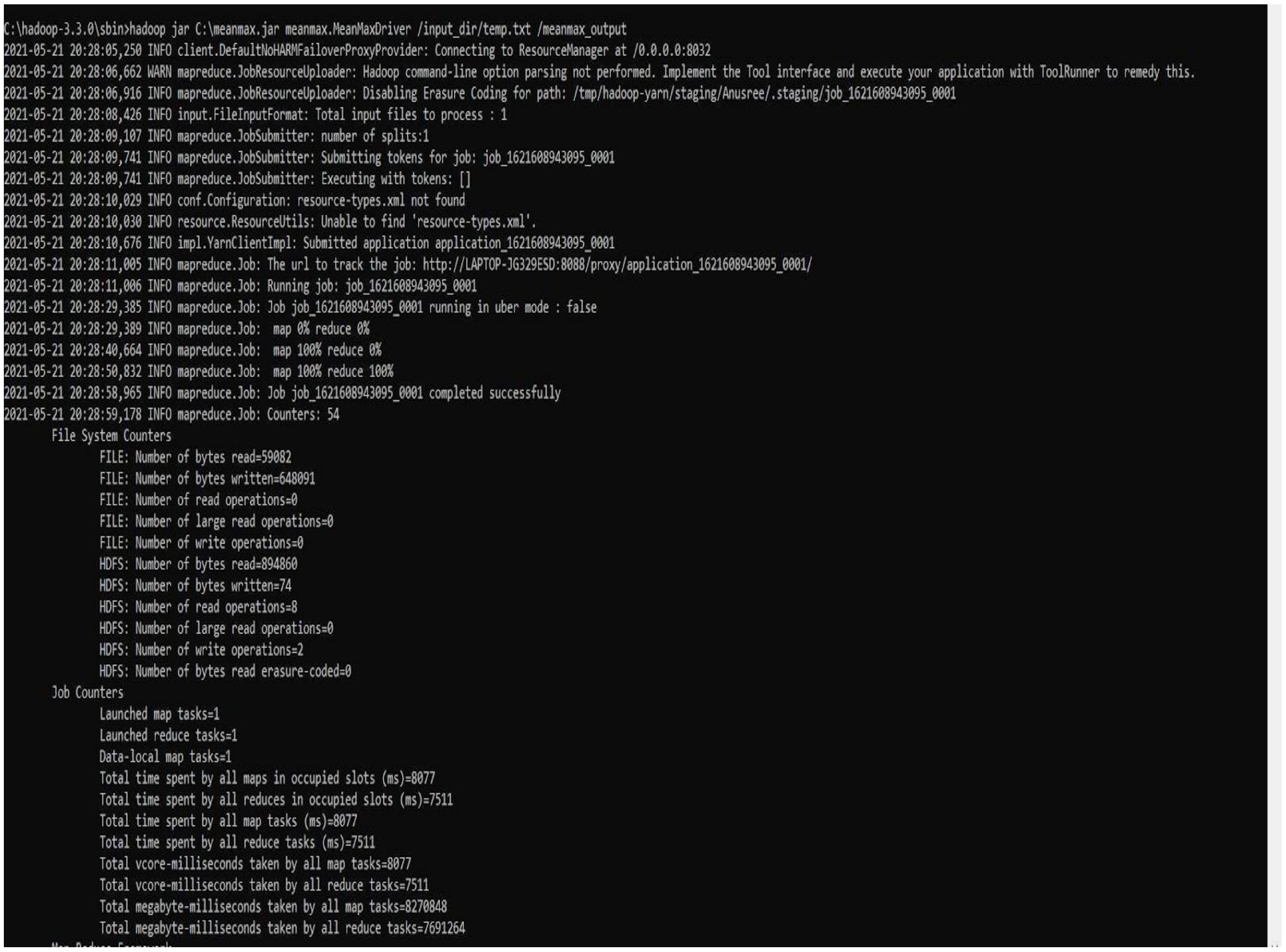
MeanMaxMapper.class

```
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));
  }
}
```

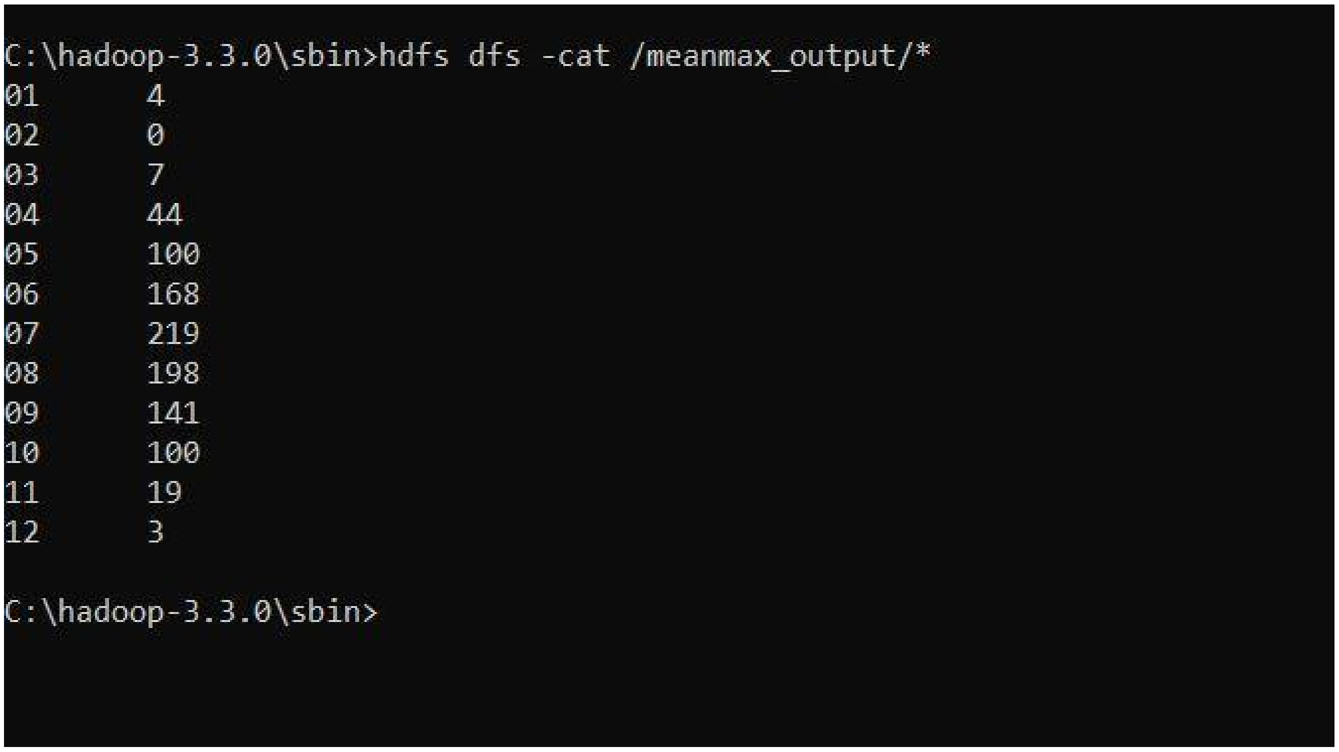
MeanMaxReducer.class

```
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));
  }
}
```



```
C:\hadoop-3.3.0\sbin>hadoop jar C:\meanmax.jar meanmax.MeanMaxDriver /input_dir/temp.txt /meanmax_output
2021-05-21 20:28:05,250 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-05-21 20:28:06,662 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2021-05-21 20:28:06,916 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1621608943095_0001
2021-05-21 20:28:08,426 INFO input.FileInputFormat: Total input files to process : 1
2021-05-21 20:28:09,107 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-21 20:28:09,741 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1621608943095_0001
2021-05-21 20:28:09,741 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-21 20:28:10,029 INFO conf.Configuration: resource-types.xml not found
2021-05-21 20:28:10,030 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-21 20:28:10,676 INFO impl.YarnClientImpl: Submitted application application_1621608943095_0001
2021-05-21 20:28:11,005 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1621608943095_0001/
2021-05-21 20:28:11,006 INFO mapreduce.Job: Running job: job_1621608943095_0001
2021-05-21 20:28:29,385 INFO mapreduce.Job: Job job_1621608943095_0001 running in uber mode : false
2021-05-21 20:28:29,389 INFO mapreduce.Job:  map 0% reduce 0%
2021-05-21 20:28:40,664 INFO mapreduce.Job:  map 100% reduce 0%
2021-05-21 20:28:50,832 INFO mapreduce.Job:  map 100% reduce 100%
2021-05-21 20:28:58,965 INFO mapreduce.Job: Job job_1621608943095_0001 completed successfully
2021-05-21 20:28:59,178 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=59082
                FILE: Number of bytes written=648091
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=894860
                HDFS: Number of bytes written=74
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=8077
                Total time spent by all reduces in occupied slots (ms)=7511
                Total time spent by all map tasks (ms)=8077
                Total time spent by all reduce tasks (ms)=7511
                Total vcore-milliseconds taken by all map tasks=8077
                Total vcore-milliseconds taken by all reduce tasks=7511
                Total megabyte-milliseconds taken by all map tasks=8270848
                Total megabyte-milliseconds taken by all reduce tasks=7691264
```



```
C:\hadoop-3.3.0\sbin>hdfs dfs -cat /meanmax_output/*
01      4
02      0
03      7
04      44
05      100
06      168
07      219
08      198
09      141
10      100
11      19
12      3

C:\hadoop-3.3.0\sbin>
```

LAB 7:

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-TopN.class

```java
package samples.topn;


import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
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.Mapper;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;


public class TopN {
  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    String[] otherArgs = (new GenericOptionsParser(conf,
args)).getRemainingArgs();
    if (otherArgs.length != 2) {
      System.err.println("Usage: TopN <in> <out>");
      System.exit(2);
    }
    Job job = Job.getInstance(conf);
    job.setJobName("Top N");
    job.setJarByClass(TopN.class);
    job.setMapperClass(TopNMapper.class);
    job.setReducerClass(TopNReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }

  public static class TopNMapper extends Mapper<Object, Text,
Text, IntWritable> {
    private static final IntWritable one = new IntWritable(1);

    private Text word = new Text();

    private String tokens = "[_|$#<>\\^=\\[\\]\\*/\\\\,;,.\\-
:()?!\"']";

    public void map(Object key, Text value, Mapper<Object, Text,
Text, IntWritable>.Context context) throws IOException,
InterruptedException {
      String cleanLine =
value.toString().toLowerCase().replaceAll(this.tokens, " ");
      StringTokenizer itr = new StringTokenizer(cleanLine);
      while (itr.hasMoreTokens()) {
        this.word.set(itr.nextToken().trim());
        context.write(this.word, one);
      }
    }
  }
}
```

TopNCombiner.class

```
package samples.topn;


import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;


public class TopNCombiner 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 sum = 0;
    for (IntWritable val : values)
      sum += val.get();
    context.write(key, new IntWritable(sum));
  }
}
```

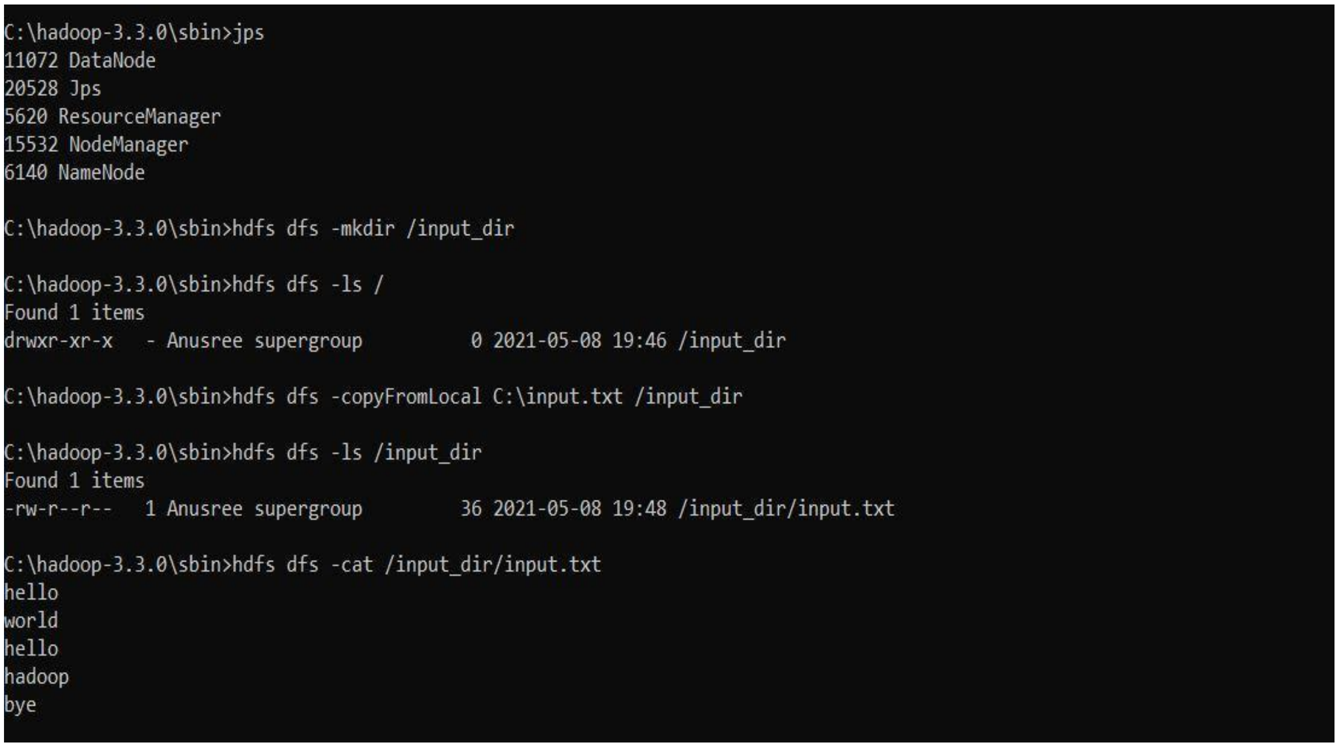
TopNMapper.class

````java
package samples.topn;


import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;


public class TopNMapper extends Mapper<Object, Text, Text,
IntWritable> {
  private static final IntWritable one = new IntWritable(1);

  private Text word = new Text();

  private String tokens = "[_|$#<>\\^=\\[\\]\\*/\\\\,;,.\\-
:()?!\"']";

  public vo```\\id map(Object key, Text value, Mapper<Object,
Text, Text, IntWritable>.Context context) throws IOException,
InterruptedException {
    String cleanLine =
value.toString().toLowerCase().replaceAll(this.tokens, " ");
    StringTokenizer itr = new StringTokenizer(cleanLine);
    while (itr.hasMoreTokens()) {
      this.word.set(itr.nextToken().trim());
      context.write(this.word, one);
    }
  }
}
````

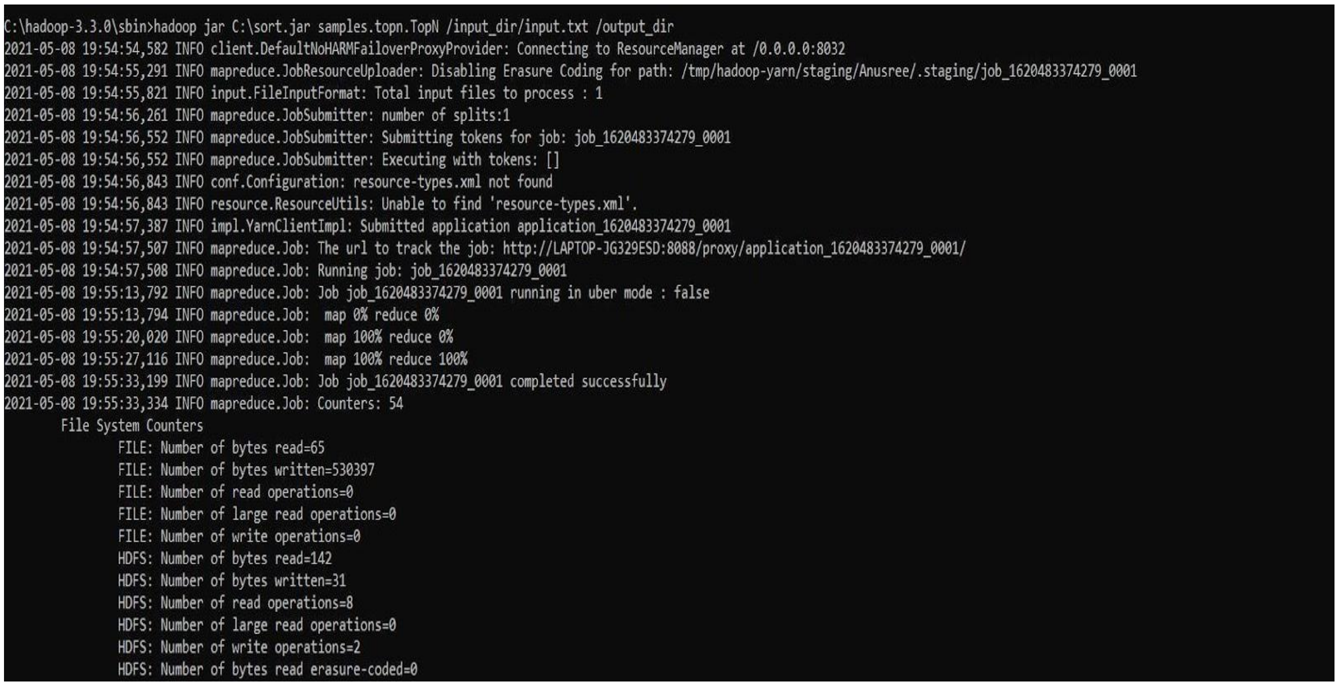
TopNReducer.class

```java
package samples.topn;


import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import utils.MiscUtils;
```

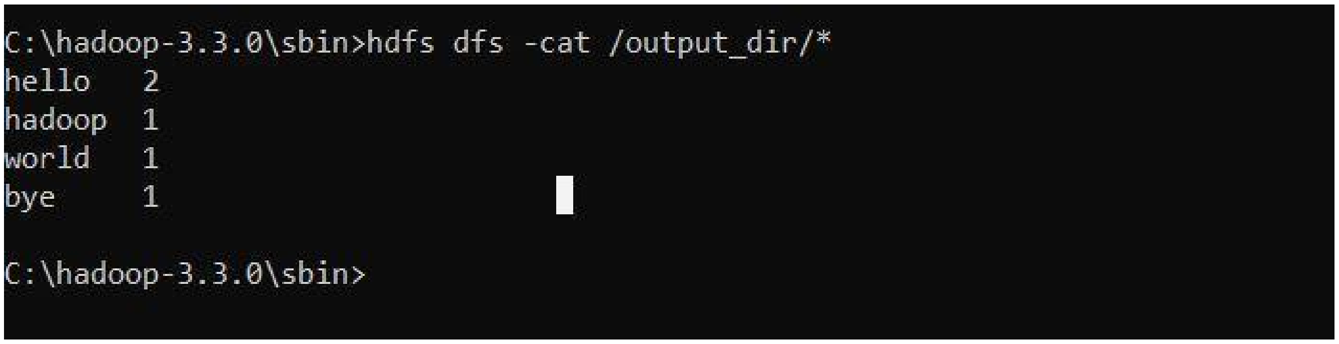
```java
public class TopNReducer extends Reducer<Text, IntWritable, Text,
IntWritable> {
  private Map<Text, IntWritable> countMap = new HashMap<>();

  public void reduce(Text key, Iterable<IntWritable> values,
Reducer<Text, IntWritable, Text, IntWritable>.Context context)
throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values)
      sum += val.get();
    this.countMap.put(new Text(key), new IntWritable(sum));
  }

  protected void cleanup(Reducer<Text, IntWritable, Text,
IntWritable>.Context context) throws IOException,
InterruptedException {
    Map<Text, IntWritable> sortedMap =
MiscUtils.sortByValues(this.countMap);
    int counter = 0;
    for (Text key : sortedMap.keySet()) {
      if (counter++ == 20)
        break;
      context.write(key, sortedMap.get(key));
    }
  }
}
```



```
C:\hadoop-3.3.0\sbin>jps
11072 DataNode
20528 Jps
5620 ResourceManager
15532 NodeManager
6140 NameNode

C:\hadoop-3.3.0\sbin>hdfs dfs -mkdir /input_dir

C:\hadoop-3.3.0\sbin>hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - Anusree supergroup          0 2021-05-08 19:46 /input_dir

C:\hadoop-3.3.0\sbin>hdfs dfs -copyFromLocal C:\input.txt /input_dir

C:\hadoop-3.3.0\sbin>hdfs dfs -ls /input_dir
Found 1 items
-rw-r--r--   1 Anusree supergroup         36 2021-05-08 19:48 /input_dir/input.txt

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /input_dir/input.txt
hello
world
hello
hadoop
bye
```



```
C:\hadoop-3.3.0\sbin>hadoop jar C:\sort.jar samples.topn.TopN /input_dir/input.txt /output_dir
2021-05-08 19:54:54,582 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-05-08 19:54:55,291 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1620483374279_0001
2021-05-08 19:54:55,821 INFO input.FileInputFormat: Total input files to process : 1
2021-05-08 19:54:56,261 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-08 19:54:56,552 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1620483374279_0001
2021-05-08 19:54:56,552 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-08 19:54:56,843 INFO conf.Configuration: resource-types.xml not found
2021-05-08 19:54:56,843 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-08 19:54:57,387 INFO impl.YarnClientImpl: Submitted application application_1620483374279_0001
2021-05-08 19:54:57,507 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1620483374279_0001/
2021-05-08 19:54:57,508 INFO mapreduce.Job: Running job: job_1620483374279_0001
2021-05-08 19:55:13,792 INFO mapreduce.Job: Job job_1620483374279_0001 running in uber mode : false
2021-05-08 19:55:13,794 INFO mapreduce.Job:  map 0% reduce 0%
2021-05-08 19:55:20,020 INFO mapreduce.Job:  map 100% reduce 0%
2021-05-08 19:55:27,116 INFO mapreduce.Job:  map 100% reduce 100%
2021-05-08 19:55:33,199 INFO mapreduce.Job: Job job_1620483374279_0001 completed successfully
2021-05-08 19:55:33,334 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=65
                FILE: Number of bytes written=530397
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=142
                HDFS: Number of bytes written=31
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
```



```
C:\hadoop-3.3.0\sbin>hdfs dfs -cat /output_dir/*
hello   2
hadoop  1
world   1
bye     1

C:\hadoop-3.3.0\sbin>
```

LAB 8: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
```

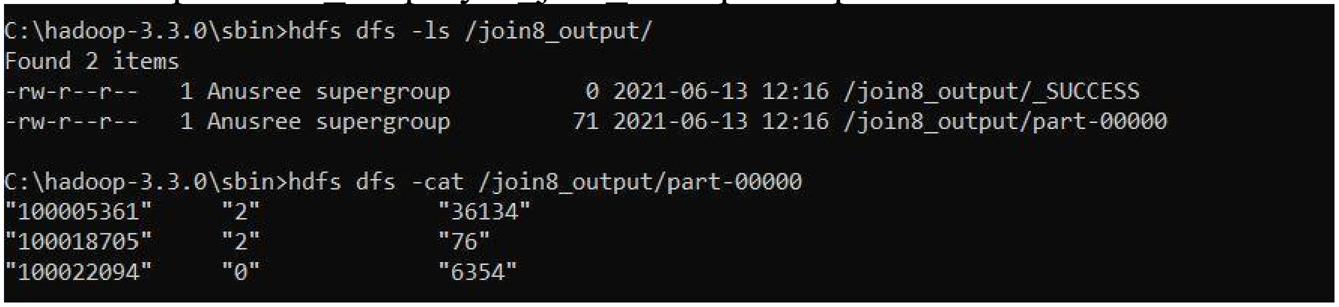
```
public void configure(JobConf job) {}

@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;
}

JobConf conf = new JobConf(getConf(), getClass());

conf.setJobName("Join 'Department Emp Strength input' with 'Department
Name
input'");

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;

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;
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 IOException
{
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;
}
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;
```

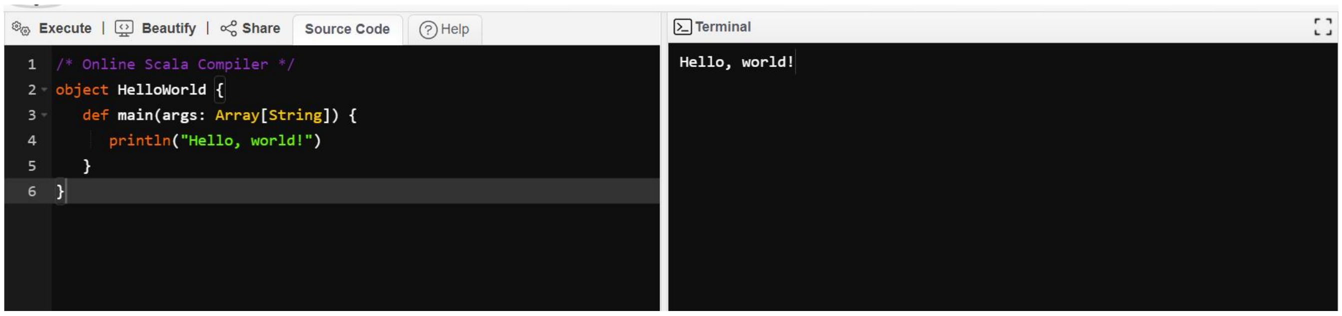
```
}

@Override
public String toString() {
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,
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) {
return ((TextPair) a).first.compareTo(((TextPair) b).first);
}
return super.compare(a, b);
}
}  }
```

LAB8/Department_Employee_join_example/DeptName.txt



```
C:\hadoop-3.3.0\sbin>hdfs dfs -ls /join8_output/
Found 2 items
-rw-r--r--   1 Anusree supergroup          0 2021-06-13 12:16 /join8_output/_SUCCESS
-rw-r--r--   1 Anusree supergroup         71 2021-06-13 12:16 /join8_output/part-00000

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /join8_output/part-00000
"100005361"     "2"             "36134"
"100018705"     "2"             "76"
"100022094"     "0"             "6354"
```

LAB 9
Program to print word count on scala shell and print “Hello world” on scala IDE



```scala
/* Online Scala Compiler */
object HelloWorld {
   def main(args: Array[String]) {
      println("Hello, world!")
   }
}
```

```
Hello, world!
```

LAB 10:

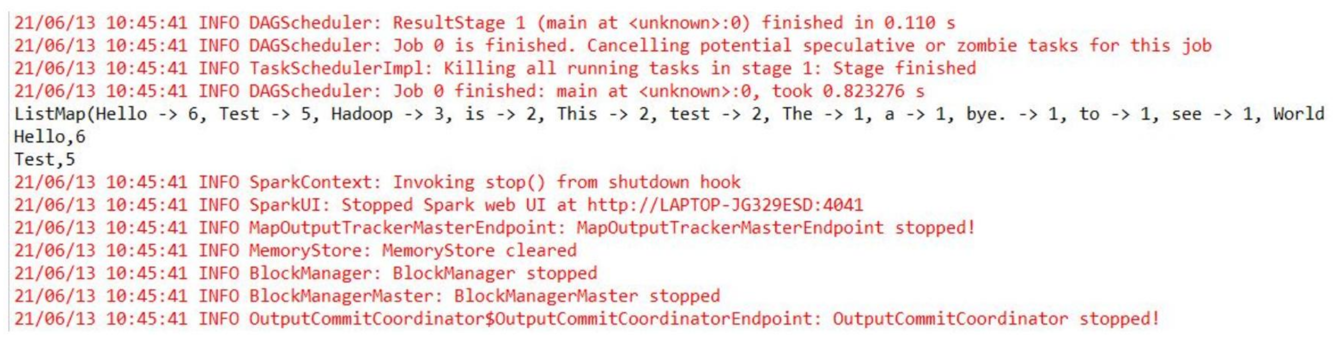
Using RDD and FlaMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark

package scalawordcount

import org.apache.spark.SparkConf
import org.apache.spark.SparkContext
import org.apache.spark.rdd.RDD.rddToPairRDDFunctions
import scala.collection.immutable.ListMap

object wordcount {
  def main (args: Array[String]) {

```
  val conf = new SparkConf().setAppName("WordCount").setMaster("local")
  val sc = new SparkContext(conf)
  val textFile = sc.textFile("input.txt")
  val counts = textFile.flatMap(line => line.split(" ")).map(word => (word,
1)).reduceByKey(_ + _)
  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()
     }
  }
  }
}
```



```
21/06/13 10:45:41 INFO DAGScheduler: ResultStage 1 (main at <unknown>:0) finished in 0.110 s
21/06/13 10:45:41 INFO DAGScheduler: Job 0 is finished. Cancelling potential speculative or zombie tasks for this job
21/06/13 10:45:41 INFO TaskSchedulerImpl: Killing all running tasks in stage 1: Stage finished
21/06/13 10:45:41 INFO DAGScheduler: Job 0 finished: main at <unknown>:0, took 0.823276 s
ListMap(Hello -> 6, Test -> 5, Hadoop -> 3, is -> 2, This -> 2, test -> 2, The -> 1, a -> 1, bye. -> 1, to -> 1, see -> 1, World
Hello,6
Test,5
21/06/13 10:45:41 INFO SparkContext: Invoking stop() from shutdown hook
21/06/13 10:45:41 INFO SparkUI: Stopped Spark web UI at http://LAPTOP-JG329ESD:4041
21/06/13 10:45:41 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
21/06/13 10:45:41 INFO MemoryStore: MemoryStore cleared
21/06/13 10:45:41 INFO BlockManager: BlockManager stopped
21/06/13 10:45:41 INFO BlockManagerMaster: BlockManagerMaster stopped
21/06/13 10:45:41 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
```