VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT on

BIG DATA ANALYTICS
(20CS6PEBDA)

Submitted by

Parvathareddy Sai Dhruv Reddy (1BM20CS104)

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
Mar-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 **Parvathareddy Sai Dhruv Reddy (1BM20CS104),** 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 2023. 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.

Vikranth B.M
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

`

Index Sheet

Course Outcome

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

1 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
        ... ;

 emp_id | salary | date_of_joining              | dept_name  | designation | emp_name
--------+--------+---------------------------------+------------+-------------+----------
    105 |  25000 | 2021-06-24 18:30:00.000000+0000 |    TESTING |   ASSOCIATE |    Sarah
    111 |  25000 | 2021-06-21 18:30:00.000000+0000 | DEVELOPING |   ASSOCIATE |      Tom
    121 |  35000 | 2021-03-29 18:30:00.000000+0000 |         HR |     MANAGER |     Elsa
    115 |  20000 | 2021-12-29 18:30:00.000000+0000 | DEVELOPING |   ASSISTANT |    Chris
    100 |  30000 | 2021-09-10 18:30:00.000000+0000 |    TESTING |     MANAGER |     John

(5 rows)
cqlsh:employee> UPDATE employee_info SET emp_name = 'Jessica', dept_name = 'DEVELOPING' WHERE
emp_id = 121;
cqlsh:employee> UPDATE employee_info SET emp_name = 'Jessica', dept_name = 'DEVELOPING' WHERE
emp_id = 121 AND salary = 35000;
cqlsh:employee> SELECT * FROM employee_info ;

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

```
 105 |  25000 | 2021-06-24 18:30:00.000000+0000 |    TESTING |   ASSOCIATE |    Sarah
 111 |  25000 | 2021-06-21 18:30:00.000000+0000 | DEVELOPING |   ASSOCIATE |      Tom
 121 |  35000 | 2021-03-29 18:30:00.000000+0000 | DEVELOPING |     MANAGER |  Jessica
 115 |  20000 | 2021-12-29 18:30:00.000000+0000 | DEVELOPING |   ASSISTANT |    Chris
 100 |  30000 | 2021-09-10 18:30:00.000000+0000 |    TESTING |     MANAGER |     John

(5 rows)
```

cqlsh:employee> SELECT * FROM employee_info WHERE emp_id in (105, 111, 121, 115, 100) order by salary;

cqlsh:employee> paging off Disabled Query paging.

cqlsh:employee> SELECT * FROM employee_info WHERE emp_id in (105, 111, 121, 115, 100) order by salary;

```
 emp_id | salary | date_of_joining                 | dept_name  | designation | emp_name
--------+--------+---------------------------------+------------+-------------+----------
 115 |  20000 | 2021-12-29 18:30:00.000000+0000 | DEVELOPING |   ASSISTANT |    Chris
 105 |  25000 | 2021-06-24 18:30:00.000000+0000 |    TESTING |   ASSOCIATE |    Sarah
 111 |  25000 | 2021-06-21 18:30:00.000000+0000 | DEVELOPING |   ASSOCIATE |      Tom
 100 |  30000 | 2021-09-10 18:30:00.000000+0000 |    TESTING |     MANAGER |     John
 121 |  35000 | 2021-03-29 18:30:00.000000+0000 | DEVELOPING |     MANAGER |  Jessica

(5 rows)
```

cqlsh:employee> ALTER TABLE employee_info ADD projects text; cqlsh:employee>

UPDATE employee_info SET projects = 'Chat App' WHERE emp_id = 111;

cqlsh:employee> UPDATE employee_info SET projects = 'Chat App' WHERE emp_id = 111 and salary = 25000;
cqlsh:employee> UPDATE employee_info SET projects = 'Discord Bot' WHERE emp_id = 115 and salary = 20000;

cqlsh:employee> UPDATE employee_info SET projects = 'Campus Portal' WHERE emp_id = 105 and salary = 25000;

cqlsh:employee> UPDATE employee_info SET projects = 'YouTube Downloader' WHERE emp_id = 100 and salary = 30000;

```
cqlsh:employee> UPDATE employee_info SET projects = 'Library Management System ' WHERE emp_id = 121
and salary = 35000;

cqlsh:employee> SELECT * FROM employee_infor

           ... ;

cqlsh:employee> SELECT * FROM employee_info ;


 emp_id | salary | date_of_joining                 | dept_name  | designation | emp_name | projects

--------+--------+---------------------------------+------------+-------------+----------+--------------------------

   105 |  25000 | 2021-06-24 18:30:00.000000+0000 |    TESTING |   ASSOCIATE |    Sarah |           Campus
Portal

   111 |  25000 | 2021-06-21 18:30:00.000000+0000 | DEVELOPING |   ASSOCIATE |      Tom |               Chat
App

   121 |  35000 | 2021-03-29 18:30:00.000000+0000 | DEVELOPING |     MANAGER |  Jessica | Library
Management System

   115 |  20000 | 2021-12-29 18:30:00.000000+0000 | DEVELOPING |   ASSISTANT |    Chris |            Discord
Bot

   100 |  30000 | 2021-09-10 18:30:00.000000+0000 |    TESTING |     MANAGER |     John |       YouTube
Downloader


(5 rows)

cqlsh:employee> INSERT INTO

           ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)

           ...

           ... ;

cqlsh:employee> INSERT INTO

           ... employee_info(emp_id,emp_name,designation,date_of_joining,salary,dept_name)

... VALUES(110,'SAM','ASSOCIATE','2021-01-11',28000,'TESTING') USING TTL 15;

cqlsh:employee> SELECT TTL(emp_name) from employee_info WHERE emp_id = 110;


 ttl(emp_name)
```

3

(1 rows)

cqlsh:employee> SELECT * FROM employee_info;

 emp_id | salary | date_of_joining | dept_name | designation | emp_name | projects

--------+--------+---------------------------------+-----------+-------------+----------+--------------------------

 105 | 25000 | 2021-06-24 18:30:00.000000+0000 | TESTING | ASSOCIATE | Sarah | Campus Portal

 111 | 25000 | 2021-06-21 18:30:00.000000+0000 | DEVELOPING | ASSOCIATE | Tom | Chat App

 121 | 35000 | 2021-03-29 18:30:00.000000+0000 | DEVELOPING | MANAGER | Jessica | Library Management System

 115 | 20000 | 2021-12-29 18:30:00.000000+0000 | DEVELOPING | ASSISTANT | Chris | Discord Bot

 100 | 30000 | 2021-09-10 18:30:00.000000+0000 | TESTING | MANAGER | John | YouTube Downloader

(5 rows)

2.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

```
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 ;


 stud_id | stud_name | book_name              | book_id | date_of_issue               | counter_value

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

```
 114 |      Raj |      Engineering Drawing |   123ER | 2021-04-02 18:30:00.000000+0000 |           1
 111 |    Manoj |      Operations Research |   56TXT | 2021-09-11 18:30:00.000000+0000 |           1
 113 |   Mahesh |          Robinson Crusoe |   34EDC | 2021-01-31 18:30:00.000000+0000 |           1
 112 |    Kamal | Engineering Mathematics-3 |   5ERW4 | 2021-04-09 18:30:00.000000+0000 |           1

(4 rows)

cqlsh:library> 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 = '2021-04-09';
cqlsh:library> SELECT * FROM library_info ;

 stud_id | stud_name | book_name               | book_id | date_of_issue                | counter_value
---------+-----------+-------------------------+---------+------------------------------+---------------
 114 |      Raj |      Engineering Drawing |   123ER | 2021-04-02 18:30:00.000000+0000 |           1
111 |    Manoj |      Operations Research |   56TXT | 2021-09-11 18:30:00.000000+0000 |           1
 113 |   Mahesh |          Robinson Crusoe |   34EDC | 2021-01-31 18:30:00.000000+0000 |           1
112 |    Kamal | Engineering Mathematics-3 |   5ERW4 | 2021-04-09 18:30:00.000000+0000 |           2

cqlsh:library> copy library_info(stud_id,stud_name, book_name, book_id, date_of_issue,counter_value) to
'library_info.csv' ;
Using 11 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_name, book_id, date_of_issue,
counter_value].
Processed: 6 rows; Rate:      39 rows/s; Avg. rate:      39 rows/s
6 rows exported to 1 files in 0.165 seconds.

cqlsh:library> copy library_info(stud_id,stud_name, book_name, book_id, date_of_issue,counter_value) from
'library_info.csv' ;
```

Using 11 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_name, book_id, date_of_issue, counter_value].

Processed: 6 rows; Rate: 10 rows/s; Avg. rate: 15 rows/s

6 rows imported from 1 files in 0.392 seconds (0 skipped).

3.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:"InternetSurfing"});
WriteResult({ "nInserted" : 1 })
> db.Student.update({_id:3,StudName:"Ayan",Grade:"vii"},{$set:{Hobbies:"skating"}},{upsert: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.count();
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.count()
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.count()
3
> db.Student.count({Grade:"vii"})
2
> db.Student.find({Grade:"vii"}).limit(3).pretty();
{
        "_id" : 1,
        "StudName" : "Megha",
        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
```

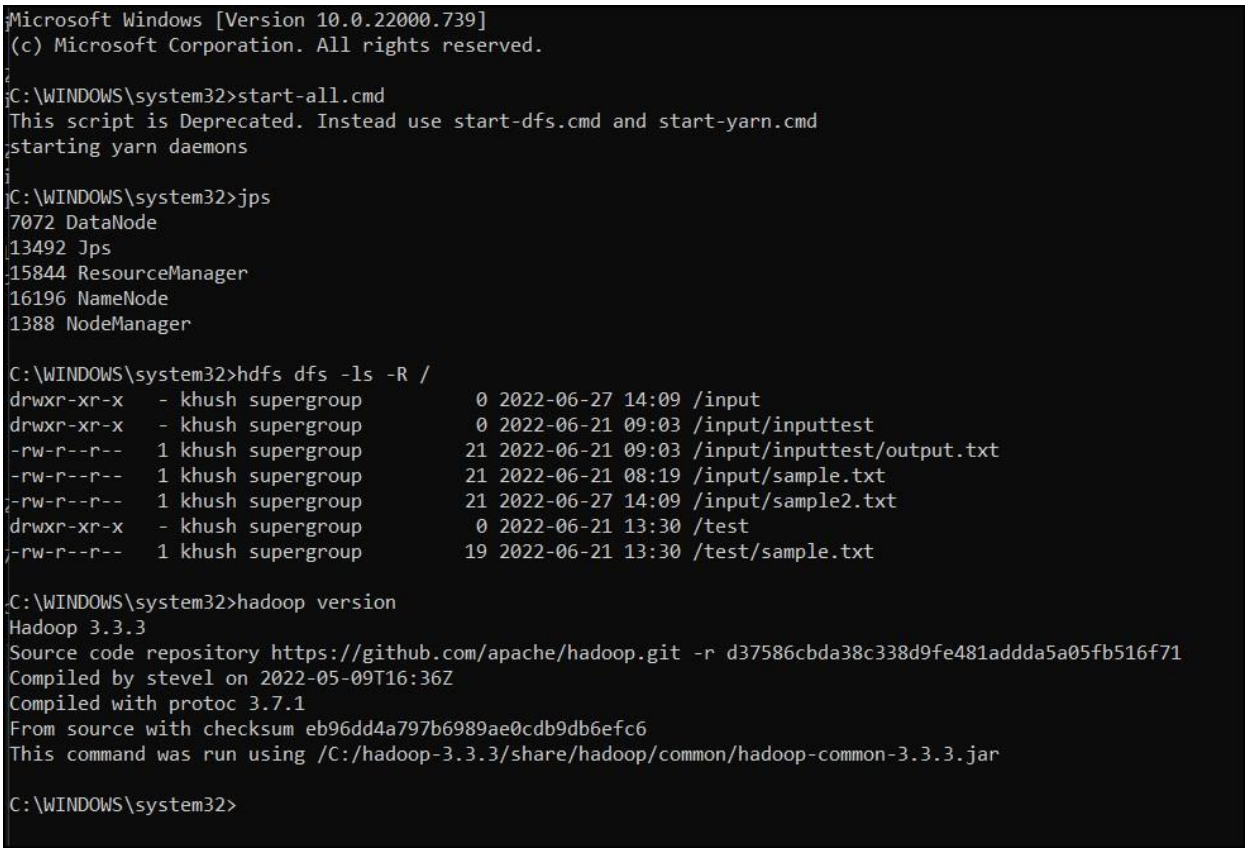
```
}{
        "_id" : 3,
        "Grade" : "vii",
        "StudName" : "Ayan",
        "Hobbies" : "skating",
        "Location" : null
}
> db.Student.find().sort({StudName:1}).pretty();
{
        "_id" : 3,
        "Grade" : "vii",
        "StudName" : "Ayan",
        "Hobbies" : "skating",
        "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','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 })
> 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:100}}})
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}}});
```

4. Screenshot of Hadoop installed



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

hduser@bmsce-Precision-T1700:~$ start-all.sh

hduser@bmsce-Precision-T1700:~$ jps

7184 NodeManager

6851 ResourceManager

6692 SecondaryNameNode

6313 NameNode

7306 Jps

6479 DataNode

hduser@bmsce-Precision-T1700:~$ hdfs dfs -mkdir /1BM19CS167

hduser@bmsce-Precision-T1700:~$ hadoop fs -ls / Found

5 items

drwxr-xr-x - hduser supergroup 0 2022-06-01 09:30 /1BM19CS167

drwxr-xr-x - hduser supergroup 0 2022-05-31 09:58 /abcde drwxr-xr-

x - hduser supergroup 0 2022-05-31 10:04 /abcdef drwxrwxr-x -

hduser supergroup 0 2019-08-01 16:19 /tmp drwxr-xr-x - hduser

supergroup 0 2019-08-01 16:03 /user

hduser@bmsce-Precision-T1700:~$ hdfs dfs -put /home/hduser/Desktop/Welcome.txt /1BM19CS167/WC.txt

hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /1BM19CS167/WC.txt

Science in our Daily Lives

As I have mentioned earlier Science has got many changes in our lives. First of all, transportation is easier now. With the help of Science it now easier to travel long distances. Moreover, the time of traveling is also reduced. Various high-speed vehicles are available these days. These vehicles have totally changed. The phase of our society. Science upgraded steam engines to electric engines. In earlier times people were traveling with cycles.

But now everybody travels on motorcycles and cars. This saves time and effort. And this is all possible with the help of Science.

hduser@bmsce-Precision-T1700:~$ hdfs dfs -get /1BM19CS167/WC.txt /home/hduser/Desktop/WWC.txt

hduser@bmsce-Precision-T1700:~$ hdfs dfs -put /home/hduser/Desktop/Welcome.txt /1BM19CS167/WC2.txt

hduser@bmsce-Precision-T1700:~$ hdfs dfs -getmerge /1BM19CS167/WC.txt /1BM19CS167/WC2.txt /home/hduser/Desktop/Merge.txt

hduser@bmsce-Precision-T1700:~$ hadoop fs -getfacl /1BM19CS167/

file: /1BM19CS167

owner: hduser

group: supergroup

user::rwx group::r-x

other::r-x

hduser@bmsce-Precision-T1700:~$ hdfs dfs -copyToLocal /1BM19CS167/WC.txt /home/hduser/Desktop

hduser@bmsce-Precision-T1700:~$ hdfs dfs -mv /1BM19CS167 /1bm19cs167

hduser@bmsce-Precision-T1700:~$ hadoop fs -ls / Found

5 items

drwxr-xr-x - hduser supergroup 0 2022-06-01 10:03 /1bm19cs167 drwxr-xr-x

- hduser supergroup 0 2022-05-31 09:58 /abcde drwxr-xr-x - hduser

supergroup 0 2022-05-31 10:04 /abcdef drwxrwxr-x - hduser supergroup

0 2019-08-01 16:19 /tmp drwxr-xr-x - hduser supergroup 0 2019-08-01

16:03 /user

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /1bm19cs167
Found 2 items
-rw-r--r--   1 hduser supergroup       1812 2022-06-01 09:39 /1bm19cs167/WC.txt
-rw-r--r--   1 hduser supergroup        607 2022-06-01 10:03 /1bm19cs167/WC2.txt

hduser@bmsce-Precision-T1700:~$ hdfs dfs -cp /1bm19cs167 /1BM19CS167

hduser@bmsce-Precision-T1700:~$ hadoop fs -ls / Found 6 items drwxr-
xr-x   - hduser supergroup          0 2022-06-01 10:15 /1BM19CS167
drwxr-xr-x   - hduser supergroup          0 2022-06-01 10:03 /1bm19cs167
drwxr-xr-x   - hduser supergroup          0 2022-05-31 09:58 /abcde drwxr-
xr-x   - hduser supergroup          0 2022-05-31 10:04 /abcdef drwxrwxr-x
- hduser supergroup          0 2019-08-01 16:19 /tmp drwxr-xr-x   - hduser
supergroup          0 2019-08-01 16:03 /user

hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /1BM19CS167
Found 2 items
-rw-r--r--   1 hduser supergroup       1812 2022-06-01 10:15 /1BM19CS167/WC.txt
-rw-r--r--   1 hduser supergroup        607 2022-06-01 10:15 /1BM19CS167/WC2.txt
```

6. Create a Map Reduce program to

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

b) find the mean max temperature for every month

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);
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));

 }

}
```

```
c:\hadoop_new\sbin>hdfs dfs -cat /tempAverageOutput/part-r-00000
1901    46
1949    94
1950    3
```

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_new\sbin>hdfs dfs -cat /tempMaxOutput/part-r-00000
01      44
02      17
03      111
04      194
05      256
06      278
07      317
08      283
09      211
10      156
11      89
12      117
```

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 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 IOException
        {
                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; 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));
        }
}

//Hadoop Commands


hduser@bmsce-Precision-T1700:~$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [localhost] hduser@localhost's password:
localhost: namenode running as process 10473. Stop it first.
hduser@localhost's password:
localhost: datanode running as process 10644. Stop it first.
Starting secondary namenodes [0.0.0.0] hduser@0.0.0.0's
password:
```

0.0.0.0: secondarynamenode running as process 10857. Stop it first.

starting yarn daemons resourcemanager running as process 9796.

Stop it first. hduser@localhost's password: localhost: nodemanager

running as process 10160. Stop it first.

hduser@bmsce-Precision-T1700:~$ jps

10160 NodeManager

7441 org.eclipse.equinox.launcher_1.5.600.v20191014-2022.jar

9796 ResourceManager

12692 org.eclipse.equinox.launcher_1.5.600.v20191014-2022.jar

10644 DataNode

10857 SecondaryNameNode

10473 NameNode

15100 Jps

hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /

Found 10 items

drwxr-xr-x   - hduser supergroup   0 2019-10-23 09:52 /gou drwxr-xr-x

- hduser supergroup   0 2019-10-23 10:33 /har drwxr-xr-x   - hduser

supergroup   0 2022-06-14 10:50 /input drwxr-xr-x   - hduser

supergroup   0 2019-10-23 09:58 /output1 drwxr-xr-x   - hduser

supergroup   0 2019-10-23 15:57 /output2 drwxr-xr-x   - hduser

supergroup   0 2022-06-15 10:27 /rgs drwxr-xr-x   - hduser

supergroup   0 2019-10-23 11:09 /stud drwxr-xr-x   - hduser

supergroup   0 2019-10-23 15:50 /testing drwxrwxr-x   - hduser

supergroup   0 2019-10-23 11:24 /tmp drwxr-xr-x   - hduser

supergroup   0 2019-08-01 16:03 /user

hduser@bmsce-Precision-T1700:~$ hadoop fs -mkdir /1BM19CS167

hduser@bmsce-Precision-T1700:~$ hadoop fs -copyFromLocal /home/hduser/Desktop/sample.txt /1BM19CS167/test.txt

hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /1BM19CS167/test.txt

hi how are you how is your job how is your family how is your brother

how is your sister

hduser@bmsce-Precision-T1700:~$ hadoop jar /home/hduser/Documents/wordCount.jar wordCount.WCDriver /1BM19CS167/test.txt /1BM19CS167/output

22/06/15 10:27:53 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.sessionid

22/06/15 10:27:53 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
22/06/15 10:27:53 INFO jvm.JvmMetrics: Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized

22/06/15 10:27:53 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.

22/06/15 10:27:53 INFO mapred.FileInputFormat: Total input paths to process : 1

22/06/15 10:27:53 INFO mapreduce.JobSubmitter: number of splits:1

22/06/15 10:27:53 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1115189753_0001

22/06/15 10:27:53 INFO mapreduce.Job: The url to track the job: http://localhost:8080/

22/06/15 10:27:53 INFO mapred.LocalJobRunner: OutputCommitter set in config null

22/06/15 10:27:53 INFO mapreduce.Job: Running job: job_local1115189753_0001

22/06/15 10:27:53 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter

22/06/15 10:27:53 INFO mapred.LocalJobRunner: Waiting for map tasks

22/06/15 10:27:53 INFO mapred.LocalJobRunner: Starting task: attempt_local1115189753_0001_m_000000_0

22/06/15 10:27:53 INFO mapred.Task: Using ResourceCalculatorProcessTree : [ ]

22/06/15 10:27:53 INFO mapred.MapTask: Processing split: hdfs://localhost:54310/rgs/test.txt:0+89

22/06/15 10:27:53 INFO mapred.MapTask: numReduceTasks: 1

22/06/15 10:27:54 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)

22/06/15 10:27:54 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100

22/06/15 10:27:54 INFO mapred.MapTask: soft limit at 83886080

22/06/15 10:27:54 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600

22/06/15 10:27:54 INFO mapred.MapTask: kvstart = 26214396; length = 6553600

22/06/15 10:27:54 INFO mapred.MapTask: Map output collector class =
org.apache.hadoop.mapred.MapTask$MapOutputBuffer 22/06/15
10:27:54 INFO mapred.LocalJobRunner:

22/06/15 10:27:54 INFO mapred.MapTask: Starting flush of map output

22/06/15 10:27:54 INFO mapred.MapTask: Spilling map output

22/06/15 10:27:54 INFO mapred.MapTask: bufstart = 0; bufend = 169; bufvoid = 104857600

22/06/15 10:27:54 INFO mapred.MapTask: kvstart = 26214396(104857584); kvend = 26214320(104857280); length = 77/6553600

22/06/15 10:27:54 INFO mapred.MapTask: Finished spill 0

22/06/15 10:27:54 INFO mapred.Task: Task:attempt_local1115189753_0001_m_000000_0 is done. And is in the process of committing

22/06/15 10:27:54 INFO mapred.LocalJobRunner: hdfs://localhost:54310/rgs/test.txt:0+89 22/06/15

10:27:54 INFO mapred.Task: Task 'attempt_local1115189753_0001_m_000000_0' done.

22/06/15 10:27:54 INFO mapred.LocalJobRunner: Finishing task:
attempt_local1115189753_0001_m_000000_0

22/06/15 10:27:54 INFO mapred.LocalJobRunner: map task executor complete.

22/06/15 10:27:54 INFO mapred.LocalJobRunner: Waiting for reduce tasks

22/06/15 10:27:54 INFO mapred.LocalJobRunner: Starting task: attempt_local1115189753_0001_r_000000_0

22/06/15 10:27:54 INFO mapred.Task: Using ResourceCalculatorProcessTree : [ ]

22/06/15 10:27:54 INFO mapred.ReduceTask: Using ShuffleConsumerPlugin:
org.apache.hadoop.mapreduce.task.reduce.Shuffle@1bc68cd5

22/06/15 10:27:54 INFO reduce.MergeManagerImpl: MergerManager: memoryLimit=334338464, maxSingleShuffleLimit=83584616, mergeThreshold=220663392, ioSortFactor=10, memToMemMergeOutputsThreshold=10

22/06/15 10:27:54 INFO reduce.EventFetcher: attempt_local1115189753_0001_r_000000_0 Thread started: EventFetcher for fetching Map Completion Events

22/06/15 10:27:54 INFO reduce.LocalFetcher: localfetcher#1 about to shuffle output of map attempt_local1115189753_0001_m_000000_0 decomp: 211 len: 215 to MEMORY

22/06/15 10:27:54 INFO reduce.InMemoryMapOutput: Read 211 bytes from map-output for attempt_local1115189753_0001_m_000000_0

22/06/15 10:27:54 INFO reduce.MergeManagerImpl: closeInMemoryFile -> map-output of size: 211, inMemoryMapOutputs.size() -> 1, commitMemory -> 0, usedMemory ->211

22/06/15 10:27:54 INFO reduce.EventFetcher: EventFetcher is interrupted.. Returning 22/06/15

10:27:54 INFO mapred.LocalJobRunner: 1 / 1 copied.

22/06/15 10:27:54 INFO reduce.MergeManagerImpl: finalMerge called with 1 in-memory map-outputs and 0 on-disk map-outputs

22/06/15 10:27:54 INFO mapred.Merger: Merging 1 sorted segments

22/06/15 10:27:54 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 205 bytes

22/06/15 10:27:54 INFO reduce.MergeManagerImpl: Merged 1 segments, 211 bytes to disk to satisfy reduce memory limit

22/06/15 10:27:54 INFO reduce.MergeManagerImpl: Merging 1 files, 215 bytes from disk

22/06/15 10:27:54 INFO reduce.MergeManagerImpl: Merging 0 segments, 0 bytes from memory into reduce

22/06/15 10:27:54 INFO mapred.Merger: Merging 1 sorted segments

22/06/15 10:27:54 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 205 bytes

22/06/15 10:27:54 INFO mapred.LocalJobRunner: 1 / 1 copied.

22/06/15 10:27:54 INFO mapred.Task: Task:attempt_local1115189753_0001_r_000000_0 is done. And is in the process of committing

22/06/15 10:27:54 INFO mapred.LocalJobRunner: 1 / 1 copied.

22/06/15 10:27:54 INFO mapred.Task: Task attempt_local1115189753_0001_r_000000_0 is allowed to commit now

22/06/15 10:27:54 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1115189753_0001_r_000000_0' to hdfs://localhost:54310/rgs/output/_temporary/0/task_local1115189753_0001_r_000000

22/06/15 10:27:54 INFO mapred.LocalJobRunner: reduce > reduce

22/06/15 10:27:54 INFO mapred.Task: Task 'attempt_local1115189753_0001_r_000000_0' done.

22/06/15 10:27:54 INFO mapred.LocalJobRunner: Finishing task: attempt_local1115189753_0001_r_000000_0

22/06/15 10:27:54 INFO mapred.LocalJobRunner: reduce task executor complete.

22/06/15 10:27:54 INFO mapreduce.Job: Job job_local1115189753_0001 running in uber mode : false

22/06/15 10:27:54 INFO mapreduce.Job: map 100% reduce 100%

22/06/15 10:27:54 INFO mapreduce.Job: Job job_local1115189753_0001 completed successfully 22/06/15 10:27:54 INFO mapreduce.Job: Counters: 38

File System Counters

FILE: Number of bytes read=8614

FILE: Number of bytes written=510599

FILE: Number of read operations=0

FILE: Number of large read operations=0

FILE: Number of write operations=0

HDFS: Number of bytes read=178

HDFS: Number of bytes written=69

HDFS: Number of read operations=13

HDFS: Number of large read operations=0

HDFS: Number of write operations=4

Map-Reduce Framework

Map input records=5

Map output records=20

Map output bytes=169

Map output materialized bytes=215

Input split bytes=87

Combine input records=0

Combine output records=0

Reduce input groups=10

Reduce shuffle bytes=215

Reduce input records=20

Reduce output records=10

```
		Spilled Records=40
		Shuffled Maps =1
		Failed Shuffles=0
		Merged Map outputs=1
		GC time elapsed (ms)=1
		CPU time spent (ms)=0
		Physical memory (bytes) snapshot=0
		Virtual memory (bytes) snapshot=0
		Total committed heap usage (bytes)=471859200	Shuffle Errors
		BAD_ID=0
		CONNECTION=0
		IO_ERROR=0
		WRONG_LENGTH=0
WRONG_MAP=0
		WRONG_REDUCE=0
	File Input Format Counters
		Bytes Read=89
	File Output Format Counters
		Bytes Written=69
0

hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /1BM19CS167/output/part-00000
are	1
brother	1
family 1
hi	1
how	5
```

is 4

job 1

sister 1

you 1

your 4

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 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; import
org.apache.hadoop.io.Text;
import
org.apache.hadoop.mapre
d.*;
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);
}
} }
```

```
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
```

9. 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(" "));

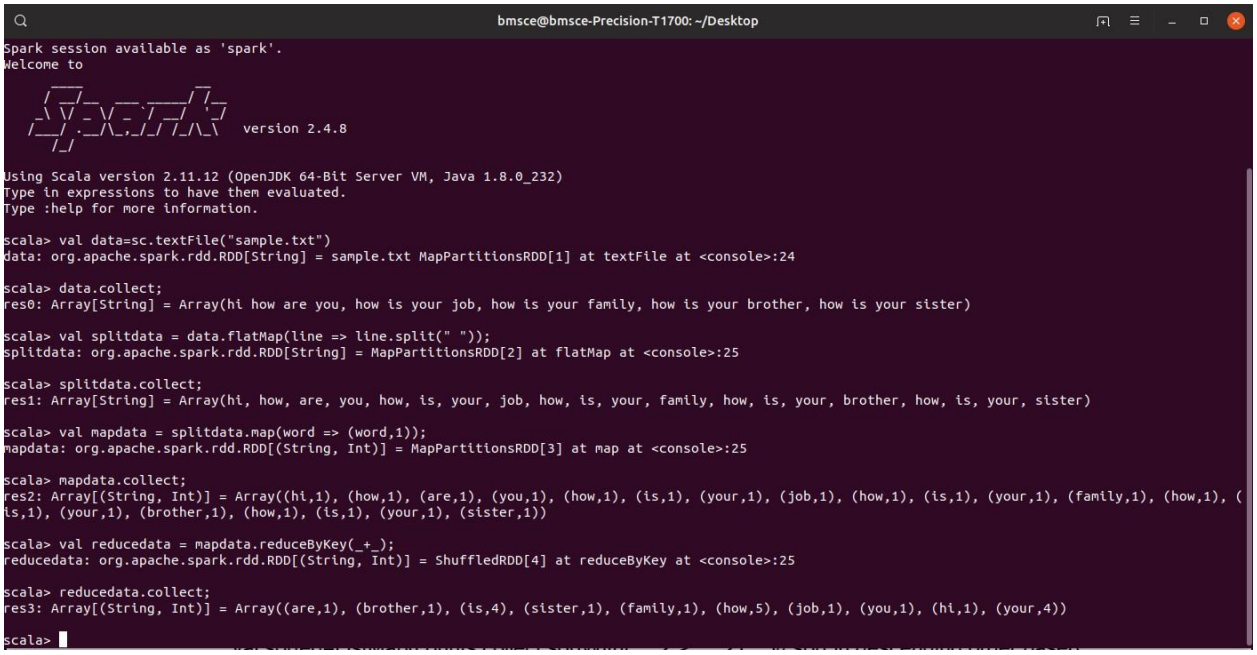
splitdata.collect;

val mapdata = splitdata.map(word => (word,1));

mapdata.collect; val reducedata =

mapdata.reduceByKey(_+_);

reducedata.collect;



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

```
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()

}
}
```