B.M.S. COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

Lab Record

BIG DATA ANALYTICS

Submitted in partial fulfilment for the 6th Semester Laboratory

Bachelor of Technology

in

Computer Science and Engineering

Submitted by:

PRITHVI J

1BM19CS122

Department of Computer Science and Engineering

B.M.S. College of Engineering

Bull Temple Road, Basavanagudi, Bangalore 560 019

Mar-July 2022

B.M.S. COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Big Data Analytics(20CS6PEBDA) laboratory has been carried out by PRITHVI J (1BM19CS122) during the 6 th Semester Mar-July-2022

Signature of the Faculty In charge:

ANTARA ROY CHOUDHURY(Assistant Professor)
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

TABLE OF CONTENTS

Employee database (CASSANDRA)

Date - 29/03/2021

Question -

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

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

4. Sort the details of Employee records based on salary

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

6. Update the altered table to add project names.

7.Create a TTL of 30 seconds to display the values of Employees.

```
cqlsh:employee_info> begin batch
            ... insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(101,'Sakshi','manag
er','2020-09-08',35000,'testing')
            ... insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(201,'Sneha','manag
er','2020-08-08',85000,'development')
            ... insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(201,'Shreya','associ
ate','2020-07-08',75000,'HR')
            ... apply batch;
cqlsh:employee_info> select *from employee_details;

 emp_id | salary | dept_name   | designation | doj                      | emp_name
```

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

201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya

201 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha

101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi

(3 rows)

cqlsh:employee_info> delete from employee_details where emp_id=201;

cqlsh:employee_info> select *from employee_details;

emp_id | salary | dept_name | designation | doj | emp_name

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

101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi

(1 rows)

cqlsh:employee_info> begin batch insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(301,'Sneha','manager','2020-08-08',85000,'development') insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(201,'Shreya','associate','2020-07-08',75000,'HR') apply batch;

cqlsh:employee_info> select *from employee_details;

emp_id | salary | dept_name | designation | doj | emp_name

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

201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya

101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi

301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha

(3 rows)

cqlsh:employee_info> alter table employee_details add project text;

cqlsh:employee_info> update employee_details set project='library app' where emp_id=201 and salary=75000;

cqlsh:employee_info> update employee_details set project='medicine app' where emp_id=301 and salary=85000;

cqlsh:employee_info> update employee_details set project='fitness app' where emp_id=101 and salary=85000;

cqlsh:employee_info> select *from employee_details;

```
 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+--------------
    201 |  75000 |         HR |   associate | 2020-07-08 07:00:00.000000+0000 |   Shreya |  library app
    101 |  35000 |     testing |     manager | 2020-09-08 07:00:00.000000+0000 |   Sakshi |         null
    101 |  85000 |        null |        null |                            null |     null |  fitness app
    301 |  85000 | development |     manager | 2020-08-08 07:00:00.000000+0000 |    Sneha |  medicine app

(4 rows)
```

cqlsh:employee_info> update employee_details set project='fitness app' where emp_id=101 and salary=35000;

cqlsh:employee_info> select *from employee_details;

```
 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+--------------
    201 |  75000 |         HR |   associate | 2020-07-08 07:00:00.000000+0000 |   Shreya |  library app
    101 |  35000 |     testing |     manager | 2020-09-08 07:00:00.000000+0000 |   Sakshi |  fitness app
    101 |  85000 |        null |        null |                            null |     null |  fitness app
    301 |  85000 | development |     manager | 2020-08-08 07:00:00.000000+0000 |    Sneha |  medicine app

(4 rows)
```

cqlsh:employee_info> delete from employee_details where emp_id=1 and salary=85000;

cqlsh:employee_info> select *from employee_details;

```
 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+--------------
    201 |  75000 |         HR |   associate | 2020-07-08 07:00:00.000000+0000 |   Shreya |  library app
```

101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app

101 | 85000 | null | null | null | null | fitness app

301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app

(4 rows)

cqlsh:employee_info> select *from employee_details;

emp_id | salary | dept_name | designation | doj | emp_name | project

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

201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya | library app

101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app

401 | 65000 | testing | manager | 2020-05-08 07:00:00.000000+0000 | Resh | null

301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app

(4 rows)

cqlsh:employee_info> insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(501,'wesh','manager','2020-04-08',95000,'testing') using ttl 30;

cqlsh:employee_info> select ttl(emp_name)from employee_details where emp_id=501 and salary=95000;

ttl(emp_name)

24

(1 rows)

cqlsh:employee_info> select *from employee_details;

emp_id | salary | dept_name | designation | doj | emp_name | project

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

201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya | library app

101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app

401 | 65000 | testing | manager | 2020-05-08 07:00:00.000000+0000 | Resh | null

301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app

(4 rows)

cqlsh:employee_info> paging off

Disabled Query paging.

cqlsh:employee_info> select *from employee_details where emp_id in(201,101,301) order by salary;

emp_id | salary | dept_name | designation | doj | emp_name | project

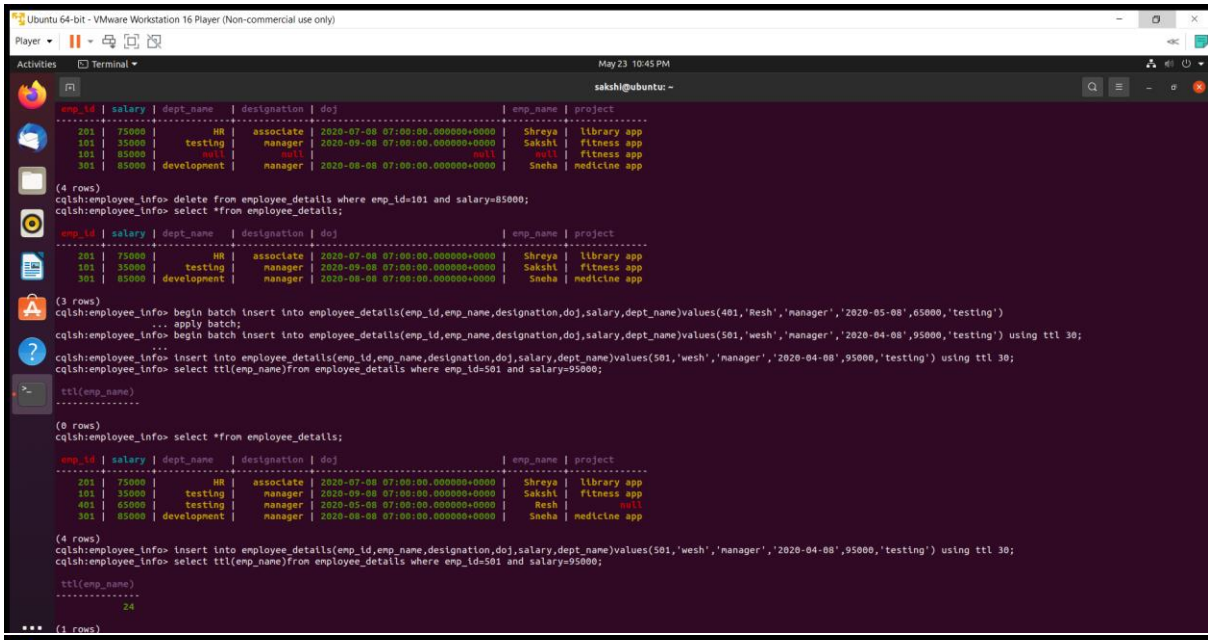
--------+--------+-------------+-------------+---------------------------------+----------+--------------

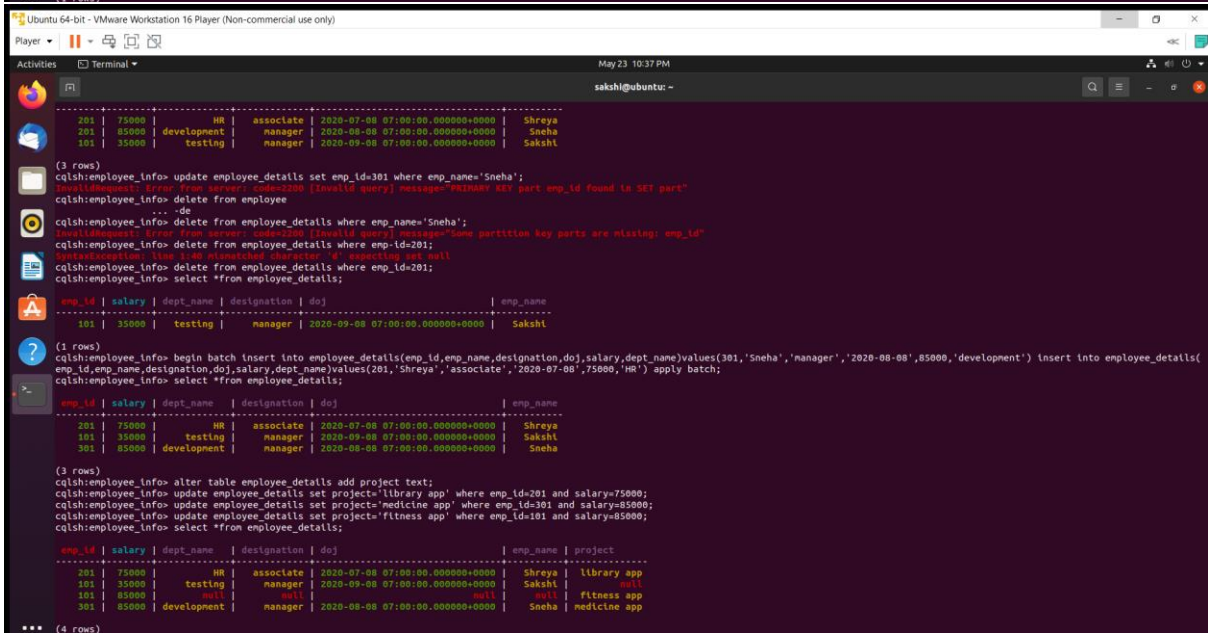
101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app

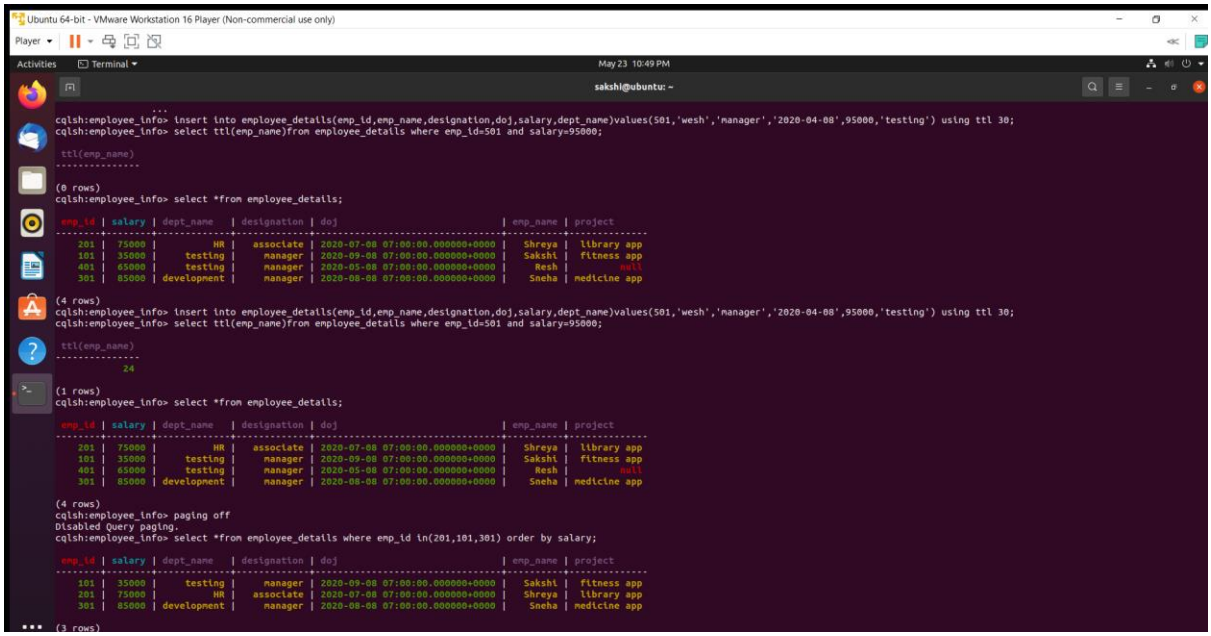
201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya | library app

301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app

SCREENSHOTS






Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
May 23 10:49 PM
sakshi@ubuntu: ~
cqlsh:employee_info> insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(501,'wesh','manager','2020-04-08',95000,'testing') using ttl 30;
cqlsh:employee_info> select ttl(emp_name)from employee_details where emp_id=501 and salary=95000;
ttl(emp_name)
(0 rows)
cqlsh:employee_info> select *from employee_details;
emp_id | salary | dept_name | designation | doj | emp_name | project
201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya | library app
101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app
401 | 65000 | testing | manager | 2020-05-08 07:00:00.000000+0000 | Resh | null
301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app
(4 rows)
cqlsh:employee_info> insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)values(501,'wesh','manager','2020-04-08',95000,'testing') using ttl 30;
cqlsh:employee_info> select ttl(emp_name)from employee_details where emp_id=501 and salary=95000;
ttl(emp_name)
24
(1 rows)
cqlsh:employee_info> select *from employee_details;
emp_id | salary | dept_name | designation | doj | emp_name | project
201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya | library app
101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app
401 | 65000 | testing | manager | 2020-05-08 07:00:00.000000+0000 | Resh | null
301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app
(4 rows)
cqlsh:employee_info> paging off
Disabled Query paging.
cqlsh:employee_info> select *from employee_details where emp_id in(201,101,301) order by salary;
emp_id | salary | dept_name | designation | doj | emp_name | project
101 | 35000 | testing | manager | 2020-09-08 07:00:00.000000+0000 | Sakshi | fitness app
201 | 75000 | HR | associate | 2020-07-08 07:00:00.000000+0000 | Shreya | library app
301 | 85000 | development | manager | 2020-08-08 07:00:00.000000+0000 | Sneha | medicine app
(3 rows)

LIBRARY DATABASE (CASSANDRA)

Date - 29/03/2021

Question -

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

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

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

5. Export the created column to a csv file

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

```
cqlsh> use library_info;
cqlsh:library_info> create table library_details(stud_id int,counter_value
           ... counter,stud_name text,book_name text,date_of_issue timestamp,book_id
           ... int,primary key(stud_id,stud_name,book_name,date_of_issue,book_id));
cqlsh:library_info> update library_details set counter_value=counter_value+1
           ... where stud_id=111 and stud_name='sam' and book_name='ML' and
           ... date_of_issue='2020-11-09' and book_id=200;
cqlsh:library_info> select *from library_details;

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

```
 111 |      sam |       ML | 2020-11-09 08:00:00.000000+0000 |     200 |           1


(1 rows)
```

cqlsh:library_info> update library_details set counter_value=counter_value+1 where stud_id=112 and stud_name='sakshi' and book_name='BDA' and date_of_issue='2020-01-01' and book_id=300;

cqlsh:library_info> update library_details set counter_value=counter_value+1 where stud_id=115 and stud_name='aditya' and book_name='OOMD' and date_of_issue='2020-06-01' and book_id=400;

cqlsh:library_info> select *from library_details;

```
 stud_id | stud_name | book_name | date_of_issue                | book_id | counter_value
---------+-----------+-----------+----------------------------------+---------+---------------
    111 |      sam |       ML | 2020-11-09 08:00:00.000000+0000 |     200 |           1
    112 |    sakshi |      BDA | 2020-01-01 08:00:00.000000+0000 |     300 |           1
    115 |    aditya |     OOMD | 2020-06-01 07:00:00.000000+0000 |     400 |           1
```

qlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) to 'C:\Desktop\sample.csv';

Using 1 child processes

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

cqlshlib.copyutil.ExportProcess.write_rows_to_csv(): writing row

cqlshlib.copyutil.ExportProcess.write_rows_to_csv(): writing row

cqlshlib.copyutil.ExportProcess.write_rows_to_csv(): writing row/s

Processed: 3 rows; Rate: 37 rows/s; Avg. rate: 6 rows/s

3 rows exported to 1 files in 0.500 seconds.

cqlsh:library_info> truncate library_details;

cqlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) from 'C:\Desktop\sample.csv';

Using 1 child processes

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

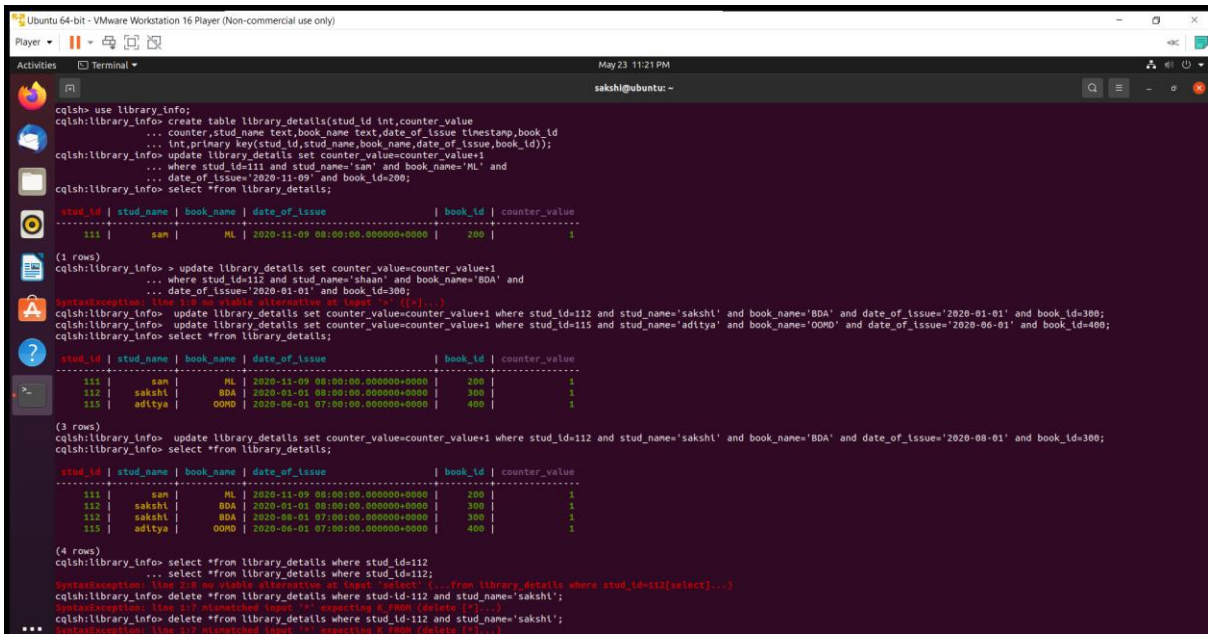
Processed: 3 rows; Rate: 3 rows/s; Avg. rate: 5 rows/s

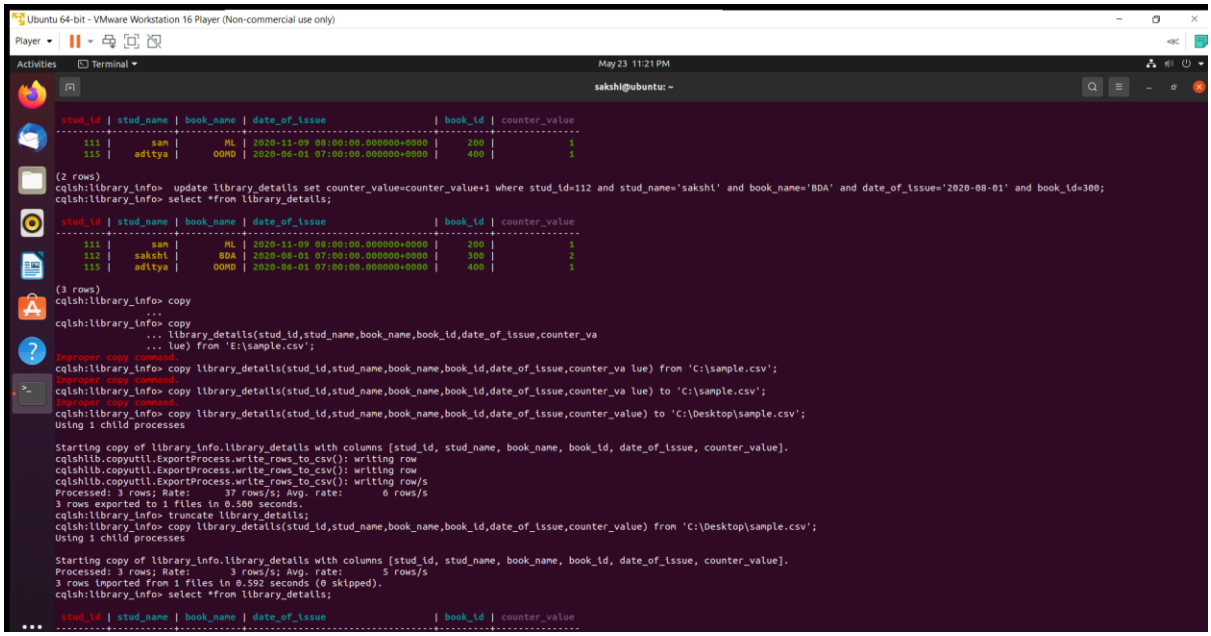
3 rows imported from 1 files in 0.592 seconds (0 skipped).

cqlsh:library_info> select *from library_details;

stud_id	stud_name	book_name	date_of_issue	book_id	counter_value
111	sam	ML	2020-11-09 08:00:00.000000+0000	200	1
112	sakshi	BDA	2020-08-01 07:00:00.000000+0000	300	2
115	aditya	OOMD	2020-06-01 07:00:00.000000+0000	400	1

SCREENSHOTS:




Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
May 23 11:21 PM
sakshi@ubuntu: ~
stud_id | stud_name | book_name | date_of_issue | book_id | counter_value
111 | sam | ML | 2020-11-09 08:00:00.000000+0000 | 200 | 1
115 | aditya | OOMD | 2020-06-01 07:00:00.000000+0000 | 400 | 1
(2 rows)
cqlsh:library_info> update library_details set counter_value=counter_value+1 where stud_id=112 and stud_name='sakshi' and book_name='BDA' and date_of_issue='2020-08-01' and book_id=300;
cqlsh:library_info> select *from library_details;
stud_id | stud_name | book_name | date_of_issue | book_id | counter_value
111 | sam | ML | 2020-11-09 08:00:00.000000+0000 | 200 | 1
112 | sakshi | BDA | 2020-08-01 07:00:00.000000+0000 | 300 | 2
115 | aditya | OOMD | 2020-06-01 07:00:00.000000+0000 | 400 | 1
(3 rows)
cqlsh:library_info> copy
... ...
cqlsh:library_info> copy
... library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_va
... lue) from 'E:\sample.csv';
Improper copy command.
cqlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_va lue) from 'C:\sample.csv';
Improper copy command.
cqlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_va lue) to 'C:\sample.csv';
Improper copy command.
cqlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) to 'C:\Desktop\sample.csv';
Using 1 child processes
Starting copy of library_info.library_details with columns [stud_id, stud_name, book_name, book_id, date_of_issue, counter_value].
cqlshlib.copyutil.ExportProcess.write_rows_to_csv(): writing row
cqlshlib.copyutil.ExportProcess.write_rows_to_csv(): writing row
cqlshlib.copyutil.ExportProcess.write_rows_to_csv(): writing row/s
Processed: 3 rows; Rate: 37 rows/s; Avg. rate: 6 rows/s
3 rows exported to 1 files in 0.500 seconds.
cqlsh:library_info> truncate library_details;
cqlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) from 'C:\Desktop\sample.csv';
Using 1 child processes
Starting copy of library_info.library_details with columns [stud_id, stud_name, book_name, book_id, date_of_issue, counter_value].
Processed: 3 rows; Rate: 3 rows/s; Avg. rate: 5 rows/s
3 rows imported from 1 files in 0.592 seconds (0 skipped).
cqlsh:library_info> select *from library_details;
stud_id | stud_name | book_name | date_of_issue | book_id | counter_value

MONGODB SAMPLE

Date - 05/04/2021

Question -

Perform the following DB operations using MongoDB.

1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.

2. Insert appropriate values

3. Write a query to update Email-Id of a student with rollno 10.

4. Replace the student name from “ABC” to “FEM” of rollno 11.

5. Export the created table into local file system

6. Drop the table

7. Import a given csv dataset from the local file system into mongodb collection.

```
> use students
switched to db students
> db.createCollection("stud_details")
{ "ok" : 1 }
>
db.stud_details.insert({'name':'Sakshi','rollno':1,'age':19,'contactno':'8670794779','email':'sakshi@bmsce.ac.in'})
WriteResult({ "nInserted" : 1 })
>
db.stud_details.insert({'name':'Sneha','rollno':2,'age':20,'contactno':'8670794789','email':'sneha@bmsce.ac.in'})
WriteResult({ "nInserted" : 1 })
>
db.stud_details.insert({'name':'Shruti','rollno':3,'age':21,'contactno':'8630394789','email':'shruti@bmsce.ac.in'})
WriteResult({ "nInserted" : 1 })
```

> db.stud_details.find({})

{ "_id" : ObjectId("60aaabf1b1aea56bb97beef8"), "name" : "Sakshi", "rollno" : 1, "age" : 19, "contactno" : "8670794779", "email" : "sakshi@bmsce.ac.in" }

{ "_id" : ObjectId("60aaac16b1aea56bb97beef9"), "name" : "Sneha", "rollno" : 2, "age" : 20, "contactno" : "8670794789", "email" : "sneha@bmsce.ac.in" }

{ "_id" : ObjectId("60aaac41b1aea56bb97beefa"), "name" : "Shruti", "rollno" : 3, "age" : 21, "contactno" : "8630394789", "email" : "shruti@bmsce.ac.in" }

> db.student_details.update({'rollno':3},{$set:{'email':'update@lab.com'}})

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

> db.stud_details.find({'rollno':3})

{ "_id" : ObjectId("60aaac41b1aea56bb97beefa"), "name" : "Shruti", "rollno" : 3, "age" : 21, "contactno" : "8630394789", "email" : "shruti@bmsce.ac.in" }

mongoexport --db students --collection stud_details --out C:\Desktop\sample.json

2021-05-22T10:43:30.687+0530 connected to: mongodb://localhost/ 2021-05-22T10:43:31.026+0530 exported 3 records

mongoimport --db students --collection stud_details --type=json --file= C:\Desktop\sample.json

2021-05-22T10:46:49.898+0530 connected to: mongodb://localhost/ 2021-05-22T10:46:50.044+0530 3 document(s) imported successfully. 0 document(s) failed to import.

db.stud_details.find({})

{ "_id" : ObjectId("60aaabf1b1aea56bb97beef8"), "name" : "Sakshi", "rollno" : 1, "age" : 19, "contactno" : "8670794779", "email" : "sakshi@bmsce.ac.in" }

{ "_id" : ObjectId("60aaac16b1aea56bb97beef9"), "name" : "Sneha", "rollno" : 2, "age" : 20, "contactno" : "8670794789", "email" : "sneha@bmsce.ac.in" }

{ "_id" : ObjectId("60aaac41b1aea56bb97beefa"), "name" : "Shruti", "rollno" : 3, "age" : 21, "contactno" : "8630394789", "email" : "shruti@bmsce.ac.in" }

> db.student_details.remove({age:{$gt:20}})

WriteResult({ "nRemoved" : 0 })

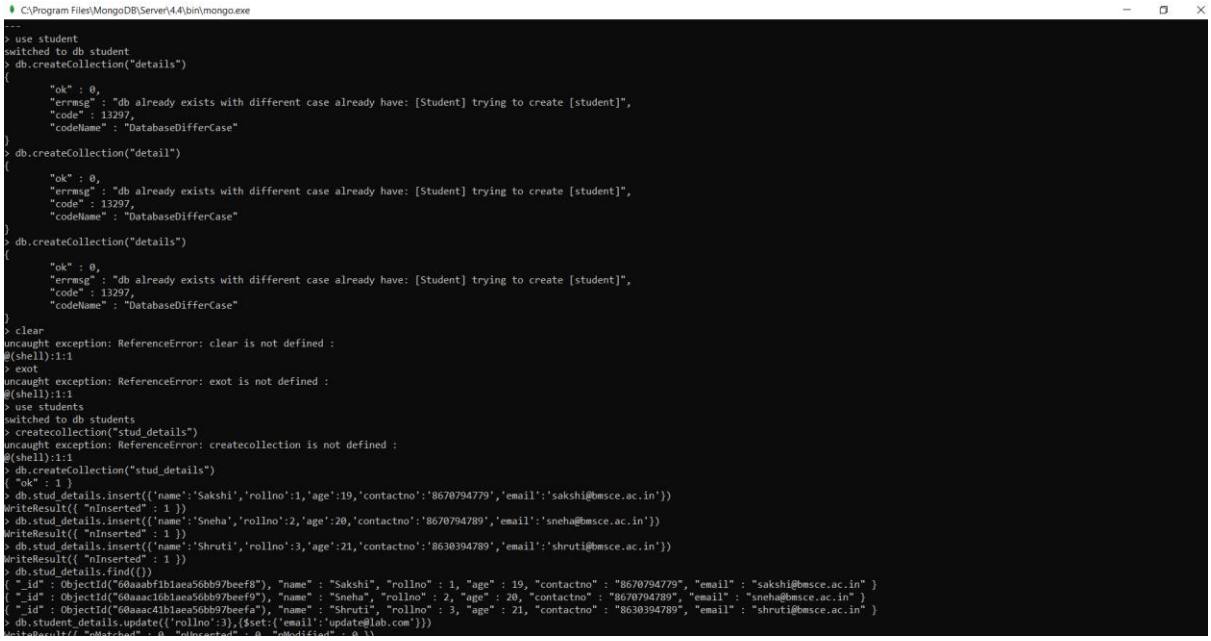
> db.stud_details.find({})

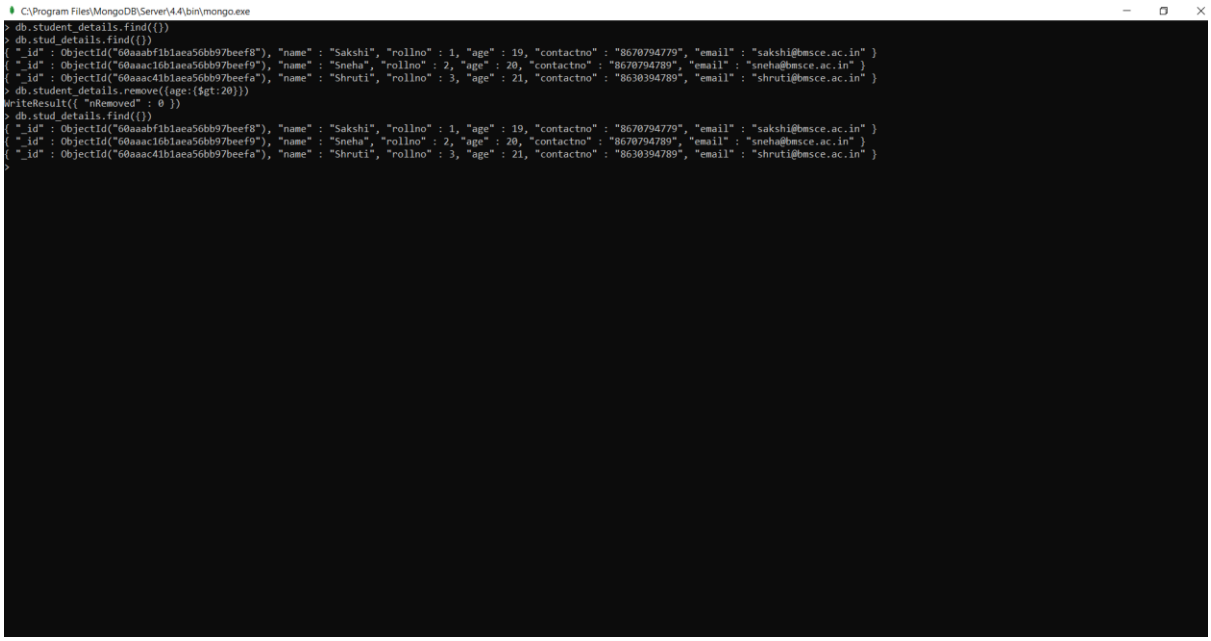
{ "_id" : ObjectId("60aaabf1b1aea56bb97beef8"), "name" : "Sakshi", "rollno" : 1, "age" : 19, "contactno" : "8670794779", "email" : "sakshi@bmsce.ac.in" }

{ "_id" : ObjectId("60aaac16b1aea56bb97beef9"), "name" : "Sneha", "rollno" : 2, "age" : 20, "contactno" : "8670794789", "email" : "sneha@bmsce.ac.in" }

{ "_id" : ObjectId("60aaac41b1aea56bb97beefa"), "name" : "Shruti", "rollno" : 3, "age" : 21, "contactno" : "8630394789", "email" : "shruti@bmsce.ac.in" }

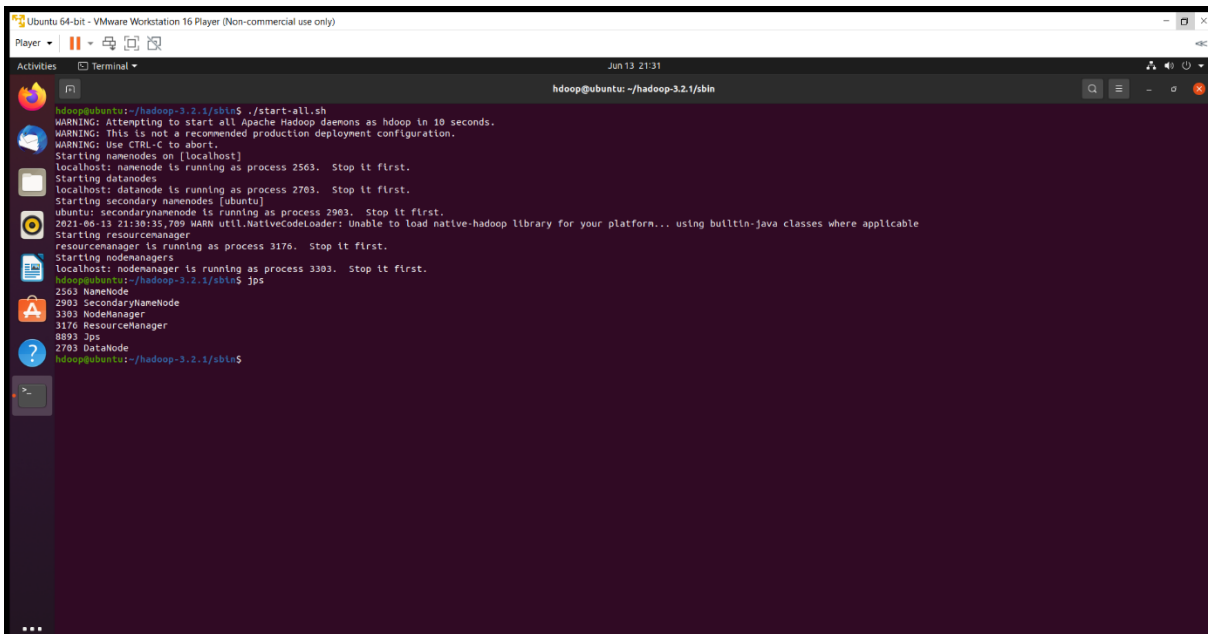
SCREENSHOTS:





SCREENSHOT OF HADOOP INSTALLATION

Date - 12/04/2021



HADOOP SAMPLE

Date - 19/04/2021

Execution of HDFS Commands for interaction with Hadoop Environment. (Minimum 10 commands to be executed)

c:\hadoop_new\sbin>hdfs dfs -mkdir /temp

c:\hadoop_new\sbin>hdfs dfs -copyFromLocal E:\Desktop\sample.txt \temp

c:\hadoop_new\sbin>hdfs dfs -ls \temp

Found 1 items

-rw-r--r-- 1 Admin supergroup 11 2021-06-11 21:12 /temp/sample.txt

```
c:\hadoop_new\sbin>hdfs dfs -cat \temp\sample.txt
hello world

c:\hadoop_new\sbin>hdfs dfs -get \temp\sample.txt E:\Desktop\temp

c:\hadoop_new\sbin>hdfs dfs -put E:\Desktop\temp \temp

c:\hadoop_new\sbin>hdfs dfs -ls \temp
Found 2 items
-rw-r--r--   1 Admin supergroup         11 2021-06-11 21:12 /temp/sample.txt
drwxr-xr-x   - Admin supergroup          0 2021-06-11 21:15 /temp/temp

c:\hadoop_new\sbin>hdfs dfs -mv \lab1 \temp

c:\hadoop_new\sbin>hdfs dfs -ls \temp
Found 3 items
drwxr-xr-x   - Admin supergroup          0 2021-04-19 15:07 /temp/lab1
-rw-r--r--   1 Admin supergroup         11 2021-06-11 21:12 /temp/sample.txt
drwxr-xr-x   - Admin supergroup          0 2021-06-11 21:15 /temp/temp

c:\hadoop_new\sbin>hdfs dfs -rm /temp/sample.txt
Deleted /temp/sample.txt

c:\hadoop_new\sbin>hdfs dfs -ls \temp
```

Found 2 items

drwxr-xr-x   - Admin supergroup   0 2021-04-19 15:07 /temp/lab1

drwxr-xr-x   - Admin supergroup   0 2021-06-11 21:15 /temp/temp

c:\hadoop_new\sbin>hdfs dfs -copyFromLocal E:\Desktop\sample.txt \temp

c:\hadoop_new\sbin>hdfs dfs -ls \temp

Found 3 items

drwxr-xr-x   - Admin supergroup   0 2021-04-19 15:07 /temp/lab1

-rw-r--r--   1 Admin supergroup   11 2021-06-11 21:17 /temp/sample.txt

drwxr-xr-x   - Admin supergroup   0 2021-06-11 21:15 /temp/temp

c:\hadoop_new\sbin>hdfs dfs -copyToLocal \temp\sample.txt E:\Desktop\sample.txt

SCREENSHOTS:

```
c:\hadoop_new\sbin>hdfs dfs -mkdir /temp

c:\hadoop_new\sbin>hdfs dfs -copyFromLocal E:\Desktop\sample.txt \temp

c:\hadoop_new\sbin>hdfs dfs -ls \temp
Found 1 items
-rw-r--r--   1 Admin supergroup         11 2021-06-11 21:12 /temp/sample.txt

c:\hadoop_new\sbin>hdfs dfs -cat \temp\sample.txt
hello world
c:\hadoop_new\sbin>hdfs dfs -get \temp\sample.txt E:\Desktop\temp

c:\hadoop_new\sbin>hdfs dfs -put E:\Desktop\temp \temp

c:\hadoop_new\sbin>hdfs dfs -ls \temp
Found 2 items
-rw-r--r--   1 Admin supergroup         11 2021-06-11 21:12 /temp/sample.txt
drwxr-xr-x   - Admin supergroup          0 2021-06-11 21:15 /temp/temp
```

MAPREDUCE TEMPERATURE

Date - 10/05/2021

For the given file, Create a Map Reduce program to

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

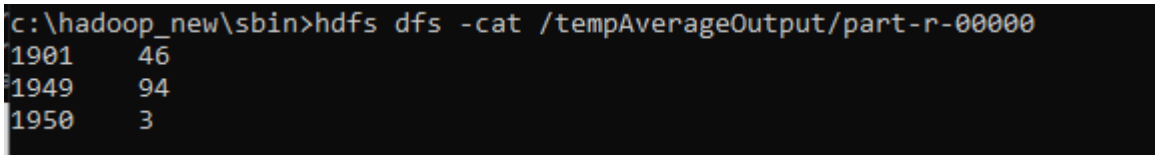
```
// AverageDriver.java
package temperature;

import org.apache.hadoop.io.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.mapreduce.*;
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.java
package temperature;

import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import java.io.IOException;

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

```
public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException
{
        String line = value.toString();
        String year = line.substring(15,19);
        int temperature;
        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 != MISSING && quality.matches("[01459]"))
                context.write(new Text(year),new IntWritable(temperature));
        }
}

//AverageReducer.java
package temperature;

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

```
public class AverageReducer extends Reducer <Text, IntWritable,Text, IntWritable>
{
	public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,InterruptedException
	{
		int max_temp = 0;
		int count = 0;
		for (IntWritable value : values)
		{
			max_temp += value.get();
			count+=1;
		}
		context.write(key, new IntWritable(max_temp/count));
	}
}
```

SCREENSHOT –



b) Find the mean max temperature for every month.

//TempDriver.java

```
package temperatureMax;
```

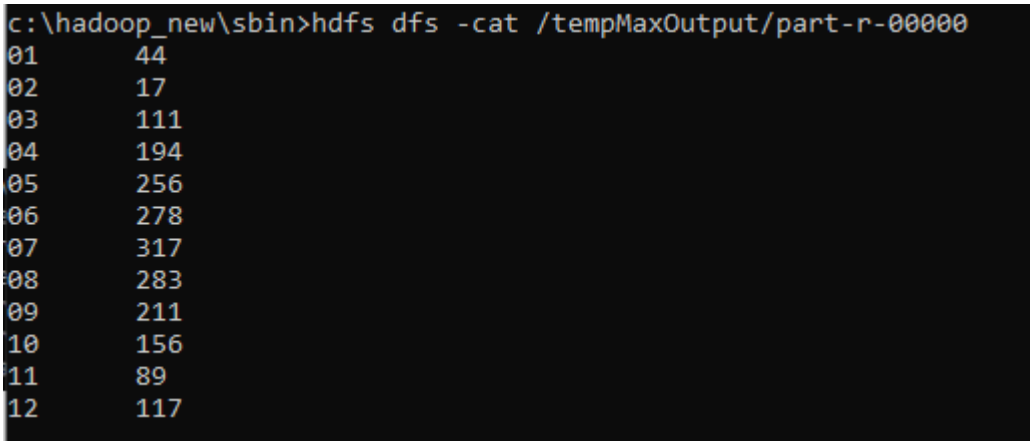
```
import org.apache.hadoop.io.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class TempDriver
{
	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(TempDriver.class);
		job.setJobName("Max temperature");
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path (args[1]));

		job.setMapperClass(TempMapper.class);
		job.setReducerClass(TempReducer.class);
```

```
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);
                System.exit(job.waitForCompletion(true)?0:1);
        }
}

//TempMapper.java
package temperatureMax;

import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import java.io.IOException;

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

public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException
{
        String line = value.toString();
        String month = line.substring(19,21);
        int temperature;
        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 != MISSING && quality.matches("[01459]"))
                context.write(new Text(month),new IntWritable(temperature));
        }
}

//TempReducer.java
package temperatureMax;

import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import java.io.IOException;

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

public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException
{
        String line = value.toString();
        String month = line.substring(19,21);
        int temperature;
```

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

SCREENSHOT -



MAPREDUCE TOPN

Date - 03/05/2021

For a given Text file, create a Map Reduce program to sort the content in an alphabetic order listing only top ‘n’ maximum occurrence of words.

```
// TopN.java

package sortWords;


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.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.util.GenericOptionsParser;

import utils.MiscUtils;


import java.io.IOException;

import java.util.*;


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.setCombinerClass(TopNReducer.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);
    }

    /**
     * The mapper reads one line at the time, splits it into an array of single words and emits
every
     * word to the reducers with the value of 1.
```

```
 */
public static class TopNMapper extends Mapper<Object, Text, Text, IntWritable> {

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    private String tokens = "[_|$#<>\\^=\\[\\]\\*/\\\\,;.\\-:()?!\"']";

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

/**
 * The reducer retrieves every word and puts it into a Map: if the word already exists in the
 * map, increments its value, otherwise sets it to 1.
 */
public static class TopNReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
```

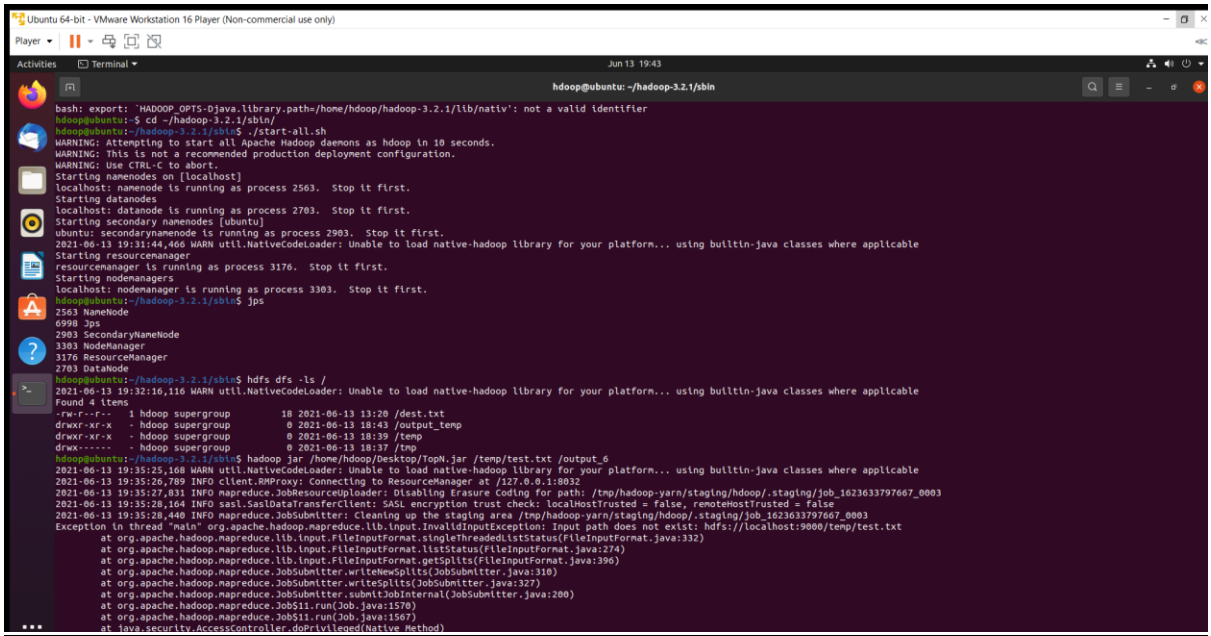
```
    private Map<Text, IntWritable> countMap = new HashMap<>();

    @Override
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws
IOException, InterruptedException {

        // computes the number of occurrences of a single word
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }

        // puts the number of occurrences of this word into the map.
        // We need to create another Text object because the Text instance
        // we receive is the same for all the words
        countMap.put(new Text(key), new IntWritable(sum));
    }

    @Override
    protected void cleanup(Context context) throws IOException, InterruptedException {

        Map<Text, IntWritable> sortedMap = MiscUtils.sortByValues(countMap);

        int counter = 0;
        for (Text key : sortedMap.keySet()) {
```

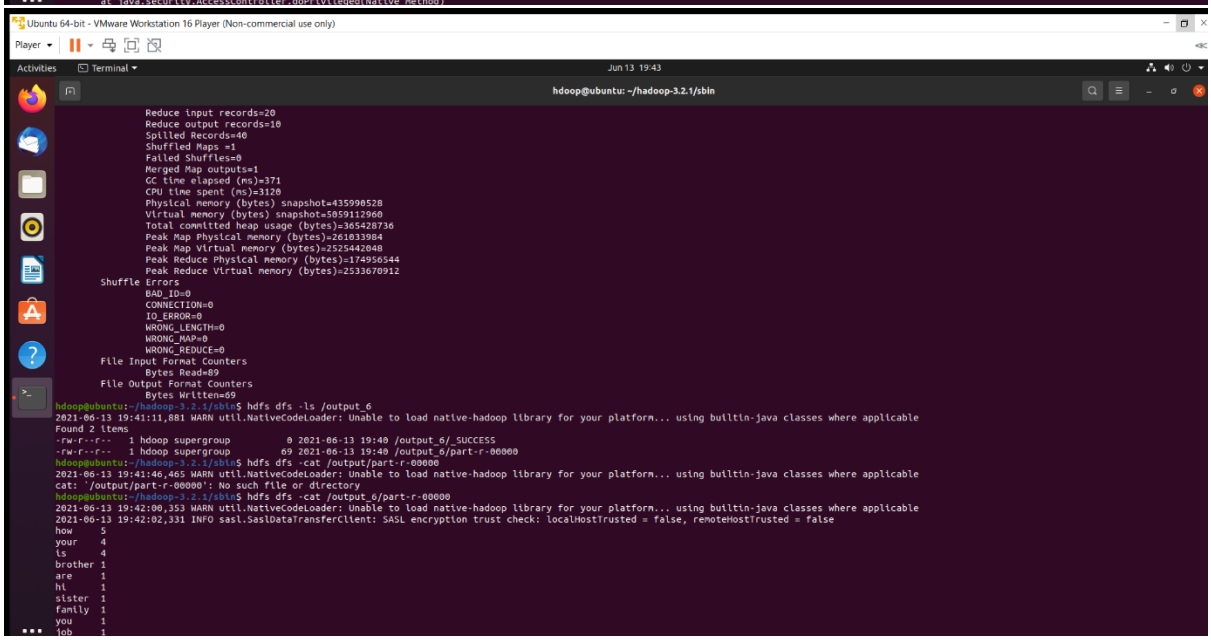
```
            if (counter++ == 3) {
                break;
            }
            context.write(key, sortedMap.get(key));
        }
    }
}

/**
 * The combiner retrieves every word and puts it into a Map: if the word already exists in the
 * map, increments its value, otherwise sets it to 1.
 */
public static class TopNCombiner extends Reducer<Text, IntWritable, Text, IntWritable> {

    @Override
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {

        // computes the number of occurrences of a single word
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
```

```
            context.write(key, new IntWritable(sum));
        }
    }
}

// MiscUtils.java
package utils;

import java.util.*;

public class MiscUtils {

    /**
     * sorts the map by values. Taken from:
     * http://javarevisited.blogspot.it/2012/12/how-to-sort-hashmap-java-by-key-and-value.html
     */
    public static <K extends Comparable, V extends Comparable> Map<K, V> sortByValues(Map<K, V> map) {
        List<Map.Entry<K, V>> entries = new LinkedList<Map.Entry<K, V>>(map.entrySet());

        Collections.sort(entries, new Comparator<Map.Entry<K, V>>() {

            @Override
```

```
            public int compare(Map.Entry<K, V> o1, Map.Entry<K, V> o2) {
                return o2.getValue().compareTo(o1.getValue());
            }
        });

        //LinkedHashMap will keep the keys in the order they are inserted
        //which is currently sorted on natural ordering
        Map<K, V> sortedMap = new LinkedHashMap<K, V>();

        for (Map.Entry<K, V> entry : entries) {
            sortedMap.put(entry.getKey(), entry.getValue());
        }

        return sortedMap;
    }
}
```

SCREENSHOTS:





MAPREDUCE JOIN

Date - 31/05/2021

Create a Hadoop Map Reduce program to combine information from the users file along with Information from the posts file by using the concept of join and display user_id, Reputation and Score.

// 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);
    }
  }
}
```

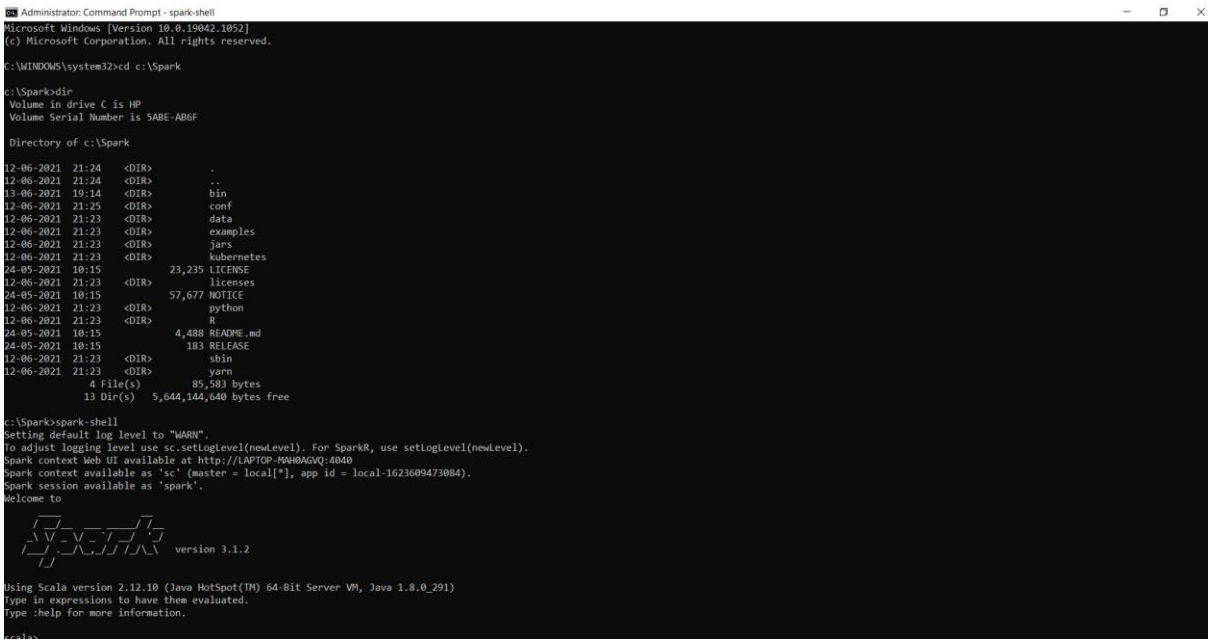
```
static {
  WritableComparator.define(TextPair.class, new Comparator());
}
public static class FirstComparator extends WritableComparator {

  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);
    }
  }
}
```

SCREENSHOTS –

```
hduser@ubuntu:~/hadoop-3.2.1/sbin$ hdfs dfs -cat /output_join/part-00000
2021-06-13 09:01:24,785 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
2021-06-13 09:01:26,736 INFO sasl.SaslDataTransferClient: SASL encryption trust
 check: localHostTrusted = false, remoteHostTrusted = false
A11      50              Finance
B12      100             HR
C13      250             Manufacturing
Dept_ID Total_Employee          Dept_Name
hduser@ubuntu:~/hadoop-3.2.1/sbin$
```

SCALA INSTALLATION SCREENSHOT



SCALA WORDCOUNT

Date - 07/06/2021

```
package count
import org.apache.spark.SparkConf


import org.apache.spark.SparkContext


import org.apache.spark.rdd.RDD.rddToPairRDDFunctions


object count {



def
main(args: Array[String]) = {




//Start the Spark context


val conf = new SparkConf()


.setAppName("count")


.setMaster("local")


val sc = new SparkContext(conf)




//Read some example file to a test RDD
```

```
val test = sc.textFile("C:\\Spark\\spark-2.4.8-bin-hadoop2.7\\bin\\testdata\\sparkdata.txt")



test.flatMap {
line => //for each line


line.split(" ") //split the line in word by word.


}


.map {
word => //for each word

      (word, 1) //Return a key/value tuple, with the word as key and 1 as value


}


.reduceByKey(_ + _) //Sum all of the value same key


.saveAsTextFile("C:\\Spark\\spark-2.4.8-bin-hadoop2.7\\bin\\testdata\\output2.txt") //Save to a
text file



//Stop the Spark context


sc.stop}}
```

SCREENSHOTS: