B.M.S. COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

Lab Record

BIG DATA ANALYTICS

Submitted in partial fulfillment for the 6th Semester Laboratory

Bachelor of Technology
in
Computer Science and Engineering

Submitted by:

SHAAN SUBBAIAH B C
1BM18CS096

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
Mar-June 2021

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 Shaan Subbaiah B C (1BM18CS096) during the 6th Semester Mar-June-2021.

Signature of the Faculty In-charge:

Sowmya V
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 15 seconds to display the values of Employees.

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

cqlsh> use employee_info;

cqlsh:employee_info> create table employee_details(emp_id int,emp_name text,designation
text,doj timestamp,salary double,dept_name text,primary key(emp_id,salary));

cqlsh:employee_info> begin batch
        ... insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)
values (100,'tanya','manager','2020-09-11',30000,'testing')
        ... insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)
values (111,'sriram','associate','2020-06-11',25000,'development')
        ... insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name)
values (121,'shiva','manager','2020-01-03',35000,'hr')
```

```
                ... apply batch;

cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name
--------+--------+-------------+-------------+---------------------------------+----------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shiva
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya

(3 rows)


cqlsh:employee_info> update employee_details set emp_name='shaan' where emp_id=121 and
salary=35000;

cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name
--------+--------+-------------+-------------+---------------------------------+----------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya

(3 rows)

cqlsh:employee_info> alter table employee_details add project text;

cqlsh:employee_info> update employee_details set project='chat app' where emp_id=111 and
salary=25000;
```

```
cqlsh:employee_info> update employee_details set project='campusx' where emp_id=121 and
salary=35000;

cqlsh:employee_info> update employee_details set project='canteen app' where emp_id=100 and
salary=30000;

cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat
app
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen
app

(3 rows)

cqlsh:employee_info> insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name)
values(113,'sam','manager','2020-09-09',30000,'testing') using ttl 30;

cqlsh:employee_info> select ttl(emp_name) from employee_details where emp_id=113 and
salary=30000;

 ttl(emp_name)
---------------
           29

(1 rows)
```

```
cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat
app
    113 |  30000 |     testing |     manager | 2020-09-08 18:30:00.000000+0000 |      sam |        null
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen
app

(4 rows)

cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat
app
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen
app

(3 rows)

cqlsh:employee_info> paging off;
Disabled Query paging.

cqlsh:employee_info> select * from employee_details where emp_id  in (111,121,100) order by
salary;
```

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

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

111 | 25000 | development | associate | 2020-06-10 18:30:00.000000+0000 | sriram | chat app

100 | 30000 | testing | manager | 2020-09-10 18:30:00.000000+0000 | tanya | canteen app

121 | 35000 | hr | manager | 2020-01-02 18:30:00.000000+0000 | shaan | campusx

(3 rows)

SCREENSHOTS -

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

```
cqlsh:employee_info> create table employee_details(emp_id int,emp_name text,designation text,doj timestamp,salary double,dept_name text,primary key(emp_id,salary));
cqlsh:employee_info> begin batch
                   ... insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values (100,'tanya','manager','2020-09-11',30000,'testing')
                   ... insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values (111,'sriram','associate','2020-06-11',25000,'development')
                   ... insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values (121,'shiva','manager','2020-01-03',35000,'hr')
                   ... apply batch;
cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name
--------+--------+-------------+-------------+---------------------------------+----------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shiva
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya

(3 rows)
```

```
cqlsh:employee_info> update employee_details set emp_name='shaan' where emp_id=121 and salary=35000;
cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name
--------+--------+-------------+-------------+---------------------------------+----------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya

(3 rows)
```

```
cqlsh:employee_info> alter table employee_details add project text;
```

```
cqlsh:employee_info> update employee_details set project='chat app' where emp_id=111 and salary=25000;
cqlsh:employee_info> update employee_details set project='campusx' where emp_id=121 and salary=35000;
cqlsh:employee_info> update employee_details set project='canteen app' where emp_id=100 and salary=30000;
cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat app
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen app

(3 rows)
```

```
cqlsh:employee_info> insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values(113,'sam','manager','2020-09-09',30000,'testing') using ttl 3
0;
cqlsh:employee_info> select ttl(emp_name) from employee_details where emp_id=113 and salary=30000;

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

(0 rows)
cqlsh:employee_info> insert into employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values(113,'sam','manager','2020-09-09',30000,'testing') using ttl 3
0;
cqlsh:employee_info> select ttl(emp_name) from employee_details where emp_id=113 and salary=30000;

 ttl(emp_name)
---------------
            29

(1 rows)
cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat app
    113 |  30000 |     testing |     manager | 2020-09-08 18:30:00.000000+0000 |      sam |        null
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen app

(4 rows)
cqlsh:employee_info> select * from employee_details;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat app
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen app

(3 rows)
```

```
cqlsh:employee_info> paging off;
Disabled Query paging.
cqlsh:employee_info> select * from employee_details where emp_id  in (111,121,100) order by salary;

 emp_id | salary | dept_name   | designation | doj                             | emp_name | project
--------+--------+-------------+-------------+---------------------------------+----------+-------------
    111 |  25000 | development |   associate | 2020-06-10 18:30:00.000000+0000 |   sriram |    chat app
    100 |  30000 |     testing |     manager | 2020-09-10 18:30:00.000000+0000 |    tanya | canteen app
    121 |  35000 |          hr |     manager | 2020-01-02 18:30:00.000000+0000 |    shaan |     campusx

(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> create keyspace library_info with replication =
{'class':'SimpleStrategy','replication_factor':1};

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> update library_details set counter_value=counter_value+1 where
stud_id=112 and stud_name='shaan' 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=113 and stud_name='ayman' 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 | 2020-11-08 18:30:00.000000+0000 |     200 |             1
     113 |     ayman |      OOMD | 2020-05-31 18:30:00.000000+0000 |     400 |             1
     112 |     shaan |       BDA | 2019-12-31 18:30:00.000000+0000 |     300 |             1

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

cqlsh:library_info> select * from library_details where stud_id=112;

 stud_id | stud_name | book_name | date_of_issue                  | book_id | counter_value
---------+-----------+-----------+--------------------------------+---------+---------------
     112 |     shaan |       BDA | 2019-12-31 18:30:00.000000+0000 |     300 |             2

(1 rows)
```

```
cqlsh:library_info> copy
library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) to
'E:\sample.csv';
Using 3 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:       1 rows/s; Avg. rate:       1 rows/s
3 rows exported to 1 files in 3.684 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
'E:\sample.csv';
Using 3 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:       1 rows/s; Avg. rate:       1 rows/s
3 rows imported from 1 files in 2.602 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 | 2020-11-08 18:30:00.000000+0000 |     200 |             1
     113 |     ayman |      OOMD | 2020-05-31 18:30:00.000000+0000 |     400 |             1
     112 |     shaan |       BDA | 2019-12-31 18:30:00.000000+0000 |     300 |             2

(3 rows)
```

SCREENSHOTS -

```
cqlsh> create keyspace library_info with replication = {'class':'SimpleStrategy','replication_factor':1};
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> update library_details set counter_value=counter_value+1 where stud_id=112 and stud_name='shaan' 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=113 and stud_name='ayman' 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 | 2020-11-08 18:30:00.000000+0000 |     200 |             1
     113 |     ayman |      OOMD | 2020-05-31 18:30:00.000000+0000 |     400 |             1
     112 |     shaan |       BDA | 2019-12-31 18:30:00.000000+0000 |     300 |             1

(3 rows)
cqlsh:library_info> update library_details set counter_value=counter_value+1 where stud_id=112 and stud_name='shaan' and book_name='BDA' and date_of_issue='2020-01-01'
and book_id=300;
cqlsh:library_info> select * from library_details where stud_id=112;

 stud_id | stud_name | book_name | date_of_issue                   | book_id | counter_value
---------+-----------+-----------+---------------------------------+---------+---------------
     112 |     shaan |       BDA | 2019-12-31 18:30:00.000000+0000 |     300 |             2

(1 rows)
```

```
cqlsh:library_info> copy library_details(stud_id,stud_name,book_name,book_id,date_of_issue,counter_value) to 'E:\sample.csv';
Using 3 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:       1 rows/s; Avg. rate:       1 rows/s
3 rows exported to 1 files in 3.684 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 'E:\sample.csv';
Using 3 child processes

Starting copy of library_info.library_details with columns [stud_id, stud_name, book_name, book_id, date_of_issue, counter_value].
Process ImportProcess-6:       1 rows/s; Avg. rate:       1 rows/s
```

```
Processed: 3 rows; Rate:       1 rows/s; Avg. rate:       1 rows/s
3 rows imported from 1 files in 2.602 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 | 2020-11-08 18:30:00.000000+0000 |     200 |             1
     113 |     ayman |      OOMD | 2020-05-31 18:30:00.000000+0000 |     400 |             1
     112 |     shaan |       BDA | 2019-12-31 18:30:00.000000+0000 |     300 |             2

(3 rows)
```

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 studentdb
switched to db studentdb

db.createCollection("student_details")
{ "ok" : 1 }

db.student_details.insert({'name':'abc','rollno':1,'age':19,'contactno':9090909090,'email':'abc@lab.com'})
WriteResult({ "nInserted" : 1 })

db.student_details.insert({'name':'mno','rollno':2,'age':20,'contactno':9999900000,'email':'mno@lab.com'})
WriteResult({ "nInserted" : 1 })

db.student_details.insert({'name':'xyz','rollno':3,'age':21,'contactno':9999911111,'email':'xyz@lab.com'})
WriteResult({ "nInserted" : 1 })
```

```
db.student_details.find({})
{ "_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19,
"contactno" : 9090909090, "email" : "abc@lab.com" }
{ "_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20,
"contactno" : 9999900000, "email" : "mno@lab.com" }
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "xyz", "rollno" : 3, "age" : 21,
"contactno" : 9999911111, "email" : "xyz@lab.com" }

db.student_details.update({'rollno':3},{$set:{'email':'update@lab.com'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.student_details.find({'rollno':3})
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "xyz", "rollno" : 3, "age" : 21,
"contactno" : 9999911111, "email" : "update@lab.com" }

db.student_details.update({'name':'xyz'},{$set:{'name':'pqr'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
db.student_details.find({'name':'pqr'})
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "pqr", "rollno" : 3, "age" : 21,
"contactno" : 9999911111, "email" : "update@lab.com" }

mongoexport --db studentdb --collection student_details --out E:\Desktop\sample.json
2021-05-22T10:43:30.687+0530    connected to: mongodb://localhost/
2021-05-22T10:43:31.026+0530    exported 3 records

db.getCollection('student_details').drop()
true

mongoimport --db studentdb --collection student_details --type=json --file=
E:\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.student_details.find({})
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "pqr", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "update@lab.com" }
{ "_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "abc@lab.com" }
{ "_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : "mno@lab.com" }

db.student_details.remove({age:{$gt:20}})
WriteResult({ "nRemoved" : 1 })

db.student_details.find({})
{ "_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "abc@lab.com" }
{ "_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : "mno@lab.com" }
```

SCREENSHOTS -

```
> use studentdb
switched to db studentdb
> db.createCollection("student_details")
{ "ok" : 1 }
> db.student_details.insert({'name':'abc','rollno':1,'age':19,'contactno':9090909090,'email':'abc@lab.com'})
WriteResult({ "nInserted" : 1 })
> db.student_details.insert({'name':'mno','rollno':2,'age':20,'contactno':9999900000,'email':'mno@lab.com'})
WriteResult({ "nInserted" : 1 })
> db.student_details.insert({'name':'xyz','rollno':3,'age':21,'contactno':9999911111,'email':'xyz@lab.com'})
WriteResult({ "nInserted" : 1 })
> db.student_details.find({})
{ "_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "abc@lab.com" }
{ "_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : "mno@lab.com" }
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "xyz", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "xyz@lab.com" }
> db.student_details.update({'rollno':3},{$set:{'email':'update@lab.com'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student_details.find({'rollno':3})
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "xyz", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "update@lab.com" }
> db.student_details.update({'name':'xyz'},{$set:{'name':'pqr'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student_details.find({'name':'pqr'})
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "pqr", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "update@lab.com" }
```

```
C:\Program Files\MongoDB\Server\4.4\bin>mongoexport --db studentdb --collection student_details --out E:\Desktop\sample.json
2021-05-22T10:43:30.687+0530    connected to: mongodb://localhost/
2021-05-22T10:43:31.026+0530    exported 3 records
```

```
> db.getCollection('student_details').drop()
true
```

```
C:\Program Files\MongoDB\Server\4.4\bin>mongoimport --db studentdb --collection student_details --type=json --file= E:\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.student_details.find({})
{ "_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "pqr", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "update@lab.com" }
{ "_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "abc@lab.com" }
{ "_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : "mno@lab.com" }
> db.student_details.remove({age:{$gt:20}})
WriteResult({ "nRemoved" : 1 })
> db.student_details.find({})
{ "_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "abc@lab.com" }
{ "_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : "mno@lab.com" }
```

SCREENSHOT OF HADOOP INSTALLATION

Date - 12/04/2021

```
C:\Users\Admin>hadoop version
Hadoop 3.1.0
Source code repository https://github.com/apache/hadoop -r 16b70619a24cdcf5d3b0fcf4b58ca77238ccbe6d
Compiled by centos on 2018-03-30T00:00Z
Compiled with protoc 2.5.0
From source with checksum 14182d20c972b3e2105580a1ad6990
This command was run using /C:/hadoop_new/share/hadoop/common/hadoop-common-3.1.0.jar

C:\Users\Admin>cd c:\hadoop_new\sbin

c:\hadoop_new\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
```

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
```

```
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
```

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 -

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

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 -

```
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
```

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 -

```
C:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \sortwords\input.txt
deer bear river
car car river
deer car bear
car deer deer
car car car
bear deer deer
```

```
C:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \sortwordsOutput\part-r-00000
car     7
deer    6
bear    3
```

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 -

```
c:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \posts\sampleposts.tsv
"2312"  "Feedback on Audio Quality"     "cs101 production audio"        "100005361"     "<p>We are looking for feedback on the audio in our videos. Tell us what you thi
nk and try to be as <em>specific</em> as possible.</p>" "question"
        "\N"    "\N"    "2012-02-23 00:28:02.321344+00" "2"     ""      "\N"    "201398145"     "2014-01-14 17:18:35.613939+00" "2960"  "\N"    "\N"    "524"   "f"
"2014856"       ""      "cs101 "        "100022094"     "<p>I also would like to know the answer to this question. An 'open exam' sounds great, but on the other hand it
 also seems pretty easy to cheat now: solutions have been posted and anybody only interested in a certificate wouldn't have much of a problem getting the highest distin
ction. So where is the catch??</p>"     "answer"        "2014706"       "2014706"       "2012-07-01 10:32:36.302782+00" "0"     ""      "\N"
        "100022094"     "2012-07-01 10:32:36.302782+00" "2020501"       "\N"    "\N"    "0"     "f"
"2004004"       ""      "cs101 "        "100018705"     "<p>But then why even the new variable q? Why not just modify the variable p?</p>"     "comment"       "2003997
"       "2003993"       "2012-05-03 21:07:52.028935+00" "2"     ""
        "\N"    "100018705"     "2012-05-03 21:07:52.028935+00" "2005150"       "\N"    "\N"    "0"     "f"

c:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \users\sampleusers.tsv
"100006402"     "18"    "0"     "0"     "0"
"100022094"     "6354"  "4"     "12"    "50"
"100018705"     "76"    "0"     "3"     "4"
"100005361"     "36134" "73"    "220"   "333"
```

```
c:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \joinOutput\part-00000
"100005361"     "2"             "36134"
"100018705"     "2"             "76"
"100022094"     "0"             "6354"
```

SCALA INSTALLATION SCREENSHOT

```
sam@ubuntu:~$ start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /opt/spark/logs/spark-sam-org.apache.spark.deploy.mas
ter.Master-1-ubuntu.out
sam@ubuntu:~$ start-slave.sh spark://ubuntu:7077
This script is deprecated, use start-worker.sh
starting org.apache.spark.deploy.worker.Worker, logging to /opt/spark/logs/spark-sam-org.apache.spark.deploy.wor
ker.Worker-1-ubuntu.out
```

```
sam@ubuntu:~$ spark-shell
21/06/13 07:19:08 WARN Utils: Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 192.168.18.
128 instead (on interface ens33)
21/06/13 07:19:08 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/opt/spark/jars/spark-unsafe_2.12-3
.1.1.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
21/06/13 07:19:10 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin
-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://192.168.18.128:4040
Spark context available as 'sc' (master = local[*], app id = local-1623593969342).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.1.1
      /_/

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 11.0.11)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

SCALA WORDCOUNT

Date - 07/06/2021

```
// scala shell
scala> val textfile = sc.textFile("/home/sam/Desktop/abc.txt")
textfile: org.apache.spark.rdd.RDD[String] = /home/sam/Desktop/abc.txt MapPartitionsRDD[1]
at textFile at <console>:24

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

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

scala> val sorted = ListMap(counts.collect.sortWith(._2>.2):*)

scala> println(sorted)
ListMap(car -> 7, deer -> 5, bear -> 3, river -> 3,  -> 1)

scala> for((k,v)<-sorted)
     | {
     |   if(v>4)
     |   {
     |     println(k+"-"+v)
     |   }
     | }
car-7
deer-5
```

```
scala> val textfile = sc.textFile("/home/sam/Desktop/abc.txt")
textfile: org.apache.spark.rdd.RDD[String] = /home/sam/Desktop/abc.txt MapPartitionsRDD[8] at textFile at <conso
le>:25

scala> val counts = textfile.flatMap(line => line.split(" ")).map(word => (word,1)).reduceByKey(_+_)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[11] at reduceByKey at <console>:26

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

scala> val sorted = ListMap(counts.collect.sortWith(_._2>_._2):_*)
sorted: scala.collection.immutable.ListMap[String,Int] = ListMap(hello -> 3, apple -> 2, unicorn -> 1, world ->
1)

scala> println(sorted)
ListMap(hello -> 3, apple -> 2, unicorn -> 1, world -> 1)
```

```
scala> for((k,v)<-sorted)
     | {
     | if(v>2)
     | {
     | println(k+"-"+v)
     | }
     | }
hello-3
```