

# **VISVESVARAYATECHNOLOGICALUNIVERSITY**

“JnanaSangama”, Belgaum -590014, Karnataka.



**LAB REPORT**  
**on**

## **BIG DATA ANALYTICS**

*Submitted by*

**SRI DEVI K P NAIK**  
**(1BM20CS162)**

*in partial fulfillment for the award of the degree of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**



**B.M.S. COLLEGE OF ENGINEERING**

(Autonomous Institution under VTU)

**BENGALURU-560019**

**Mar-2023 to July-2023**

**B. M. S. College of Engineering**  
**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the Lab work entitled "LAB COURSE **BIG DATA ANALYTICS** " was **carried** out by **SRI DEVI K P NAIK (1BM20CS162)**, who is a bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics - (20CS6PEBDA)** work prescribed for the said degree.

Dr. Manjunath D R  
Assistant Professor

**Dr. Jyothi S Nayak**  
Professor and Head of Department

### **Index Sheet**

<b>Sl. No.</b>	<b>Experiment Title</b>
<b>01</b>	<b>MongoDB Commands</b>
<b>02</b>	<b>Cassandra program for Employee details</b>
<b>03</b>	<b>Cassandra Library Database</b>
<b>04</b>	<b>Hadoop Commands</b>
<b>05</b>	<b>Word Count program in Hadoop</b>
<b>06</b>	<b>Average Temperature in Hadoop</b>
<b>07</b>	<b>Mean Max Temperature in Hadoop</b>
<b>08</b>	<b>Map Reduce Program in Hadoop using Joins</b>
<b>09</b>	<b>Spark program for Word Count</b>
<b>10</b>	<b>Spark program for Word Count greater than 4.</b>

## Program 01: MongoDB commands

**To execute create, insert, update, find and count commands of MongoDB**

```
$mongosh
```

```
test> show dbs; admin
```

```
40.00 KiB config 60.00
```

```
KiB local 72.00 KiB test>
```

```
use database1
```

```
database1> db.createCollection("student"); database1>
db.student.insert({_id:1,StudName:"student1",Sem:6});
{ acknowledged: true, insertedIds: { '0': 1 } } database1>
db.student.insert({_id:2,StudName:"student2",Sem:6});
{ acknowledged: true, insertedIds: { '0': 2 } } database1>
db.student.insert({_id:3,StudName:"student3",Sem:6});
{ acknowledged: true, insertedIds: { '0': 3 } } database1>
db.student.insert({_id:4,StudName:"student4",Sem:6});
{ acknowledged: true, insertedIds: { '0': 4 } } database1>
db.student.insert({_id:5,StudName:"student5",Sem:6});
{ acknowledged: true, insertedIds: { '0': 5 } } database1>
db.student.insert({_id:6,StudName:"student6",Sem:6});
{ acknowledged: true, insertedIds: { '0': 6 } }
```

```
database1> show collections student
```

```
database1> db.student.find()
```

```
[
  { _id: 1, StudName: 'student1', Sem: 6 },
  { _id: 2, StudName: 'student2', Sem: 6 },
  { _id: 3, StudName: 'student3', Sem: 6 },
  { _id: 4, StudName: 'student4', Sem: 6 },
  { _id: 5, StudName: 'student5', Sem: 6 },
  { _id: 6, StudName: 'student6', Sem: 6 } ]
```

```
database1> db.student.find({StudName:"student1"}); [ { _id: 1,
StudName: 'student1', Sem: 6 } ]
```

```
database1> db.student.count()
```

```
6
```

```
` database1> db.student.find({Sem:6});
```

```
[
```

```
  { _id: 1, StudName: 'student1', Sem: 6 },
```

```
  { _id: 2, StudName: 'student2', Sem: 6 },
```

```
  { _id: 3, StudName: 'student3', Sem: 6 },
```

```
  { _id: 4, StudName: 'student4', Sem: 6 },
```

```
  { _id: 5, StudName: 'student5', Sem: 6 },
```

```
  { _id: 6, StudName: 'student6', Sem: 6 }
```

```
]
```

```
database1> db.student.update({_id:4,StudName:"student4"},{$set:{Sem:7}},{upsert:
true});
```

```
database1> db.student.find()
```

```
[
```

```
  { _id: 1, StudName: 'student1', Sem: 6 },
```

```
  { _id: 2, StudName: 'student2', Sem: 6 },
```

```
  { _id: 3, StudName: 'student3', Sem: 6 },
```

```
  { _id: 4, StudName: 'student4', Sem: 7 },
```

```
  { _id: 5, StudName: 'student5', Sem: 6 },
```

```
  { _id: 6, StudName: 'student6', Sem: 6 }
```

```
]
```

```
database1> db.student.find().pretty()
```

```
[
```

```
  { _id: 1, StudName: 'student1', Sem: 6 },
```

```
  { _id: 2, StudName: 'student2', Sem: 6 },
```

```
  { _id: 3, StudName: 'student3', Sem: 6 },
```

```
  { _id: 4, StudName: 'student4', Sem: 7 },
```

```
  { _id: 5, StudName: 'student5', Sem: 6 },
```

```
  { _id: 6, StudName: 'student6', Sem: 6 }
```

```
]
```

```
database1> db.student.update({_id:5,StudName:"student5"},{$unset:{Sem:6}},{upser
```

```

t:true});
database1> db.student.find().pretty()
[
  { _id: 1, StudName: 'student1', Sem: 6 },
  { _id: 2, StudName: 'student2', Sem: 6 },
  { _id: 3, StudName: 'student3', Sem: 6 },
  { _id: 4, StudName: 'student4', Sem: 7 },
  { _id: 5, StudName: 'student5' },
  { _id: 6, StudName: 'student6', Sem: 6 }
]

database1> db.student.update({_id:6},{ $set:{OE:"OR"}},{upsert:true}); database1>
db.student.find()
[
  { _id: 1, StudName: 'student1', Sem: 6 },
  { _id: 2, StudName: 'student2', Sem: 6 },
  { _id: 3, StudName: 'student3', Sem: 6 },
  { _id: 4, StudName: 'student4', Sem: 7 },
  { _id: 5, StudName: 'student5' },
  { _id: 6, StudName: 'student6', Sem: 6, OE: 'OR' }
]

database1> db.student.find({OE:"OR"});
[ { _id: 6, StudName: 'student6', Sem: 6, OE: 'OR' } ]

database1> db.student.count({Sem:6});
4

database1> db.student.find({Sem:6}).limit(4);
[
  { _id: 1, StudName: 'student1', Sem: 6 },
  { _id: 2, StudName: 'student2', Sem: 6 },
  { _id: 3, StudName: 'student3', Sem: 6 },
  { _id: 6, StudName: 'student6', Sem: 6, OE: 'OR' }
] database1> db.student.find({StudName:"student2",Sem:6}); [ { _id: 2,
StudName: 'student2', Sem: 6 } ]

database1> db.student.find().sort({StudName:1}).pretty();

```

```
[
  { _id: 1, StudName: 'student1', Sem: 6 },
  { _id: 2, StudName: 'student2', Sem: 6 },
  { _id: 3, StudName: 'student3', Sem: 6 },
  { _id: 4, StudName: 'student4', Sem: 7 },
  { _id: 5, StudName: 'student5' },
  { _id: 6, StudName: 'student6', Sem: 6, OE: 'OR' }
] database1> db.student.find().sort({StudName:-1}).pretty();
```

```
[
  { _id: 6, StudName: 'student6', Sem: 6, OE: 'OR' },
  { _id: 5, StudName: 'student5' },
  { _id: 4, StudName: 'student4', Sem: 7 },
  { _id: 3, StudName: 'student3', Sem: 6 },
  { _id: 2, StudName: 'student2', Sem: 6 },
  { _id: 1, StudName: 'student1', Sem: 6 }
]
```

```
database1> db.student.find().skip(3).pretty()
```

```
[
  { _id: 4, StudName: 'student4', Sem: 7 },
  { _id: 5, StudName: 'student5' },
  { _id: 6, StudName: 'student6', Sem: 6, OE: 'OR' }
]
```

```
database1> db.student.count({Sem:7});
```

```
1
```

## Program 02: Cassandra Commands

### Perform the following DB operations using Cassandra

1. Create a keyspace by name Employee

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

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

```
create table EmployeeInfo (  
    ... EmplID int PRIMARY KEY,  
    ... EmplName text,  
    ... Designation text,  
    ... DateOfJoining timestamp,  
    ... Salary int,  
    ... DeptName text  
    ... );
```

3. Insert the values into the table in batch begin

batch

```
insert into EmployeeInfo (EmplID, EmplName, Designation,  
DateOfJoining, Salary, DeptName) values (101, 'employee1',  
'designation1', '2020-03-29', 40000, 'dept1')
```

```
insert into EmployeeInfo (EmplID, EmplName, Designation,  
DateOfJoining, Salary, DeptName) values (102, 'employee2',
```



'designation2', '2020-06-04', 60000, 'dept1') insert into EmployeeInfo

(EmplID, EmplName, Designation, DateOfJoining, Salary, DeptName)

values (103,

'employee3',

'designation3', '2020-04-21', 75000, 'dept1')

insert into EmployeeInfo (EmplID, EmplName, Designation,  
DateOfJoining, Salary, DeptName) values (104, 'employee4',  
'designation4', '2020-12-02', 90000, 'dept2')

insert into EmployeeInfo (EmplID, EmplName, Designation,  
DateOfJoining, Salary, DeptName) values (105, 'employee5',  
'designation5', '2020-09-11', 15000, 'dept2') apply batch;

emplid	dateofjoining	deptname	designation	emplname	salary
105	2020-09-10 18:30:00.000000+0000	dept2	designation5	employee5	15000
104	2020-12-01 18:30:00.000000+0000	dept2	designation4	employee4	90000
102	2020-06-03 18:30:00.000000+0000	dept1	designation2	employee2	60000
101	2020-03-28 18:30:00.000000+0000	dept1	designation1	employee1	40000
103	2020-04-20 18:30:00.000000+0000	dept1	designation3	employee3	75000

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

insert into EmployeeInfo (EmplID, EmplName, Designation,  
DateOfJoining, Salary, DeptName) values (121, 'employee6',  
'designation6', '2020-10-18', 45000, 'dept1'); select \* from

EmployeeInfo;

emplid	dateofjoining	deptname	designation	emplname	salary
105	2020-09-10 18:30:00.000000+0000	dept2	designation5	employee5	15000
121	2020-10-17 18:30:00.000000+0000	dept1	designation6	employee6	45000
104	2020-12-01 18:30:00.000000+0000	dept2	designation4	employee4	90000
102					

```

2020-06-03 18:30:00.000000+0000 | dept1 | designation2 | employee2 | 60000 101 |
2020-03-28 18:30:00.000000+0000 | dept1 | designation1 | employee1 | 40000 103 | 2020-04-20
18:30:00.000000+0000 | dept1 | designation3 | employee3 | 75000
update EmployeeInfo SET EmplName='employee7', DeptName='dept2' where EmplID=121;
select * from EmployeeInfo;

```

	emplid	dateofjoining	deptname	designation	emplname	salary
105	2020-09-10 18:30:00.000000+0000	dept2	designation5	employee5	15000 121	2020-10-17
	18:30:00.000000+0000	dept2	designation6	employee7	45000 104	
	2020-12-01 18:30:00.000000+0000	dept2	designation4	employee4	90000 102	
	2020-06-03 18:30:00.000000+0000	dept1	designation2	employee2	60000 101	
	2020-03-28 18:30:00.000000+0000	dept1	designation1	employee1	40000 103	2020-04-20
	18:30:00.000000+0000	dept1	designation3	employee3	75000	

### 5. Sort the details of Employee records based on salary

```

select * from Employee_info where Emp_id in(101,102,103,104,121,105)
order by salary desc;

```

	emplid	dateofjoining	deptname	designation	emplname	salary
105	2020-09-10 18:30:00.000000+0000	dept2	designation5	employee5	15000 121	2020-10-17
	18:30:00.000000+0000	dept2	designation6	employee7	45000 104	
	2020-12-01 18:30:00.000000+0000	dept2	designation4	employee4	90000 102	
	2020-06-03 18:30:00.000000+0000	dept1	designation2	employee2	60000 101	
	2020-03-28 18:30:00.000000+0000	dept1	designation1	employee1	40000 103	2020-04-20
	18:30:00.000000+0000	dept1	designation3	employee3	75000	

### 6. Alter the schema of the table Employee\_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.

```

alter table EmployeeInfo add Projects text; select * from

```

```

EmployeeInfo;

```

	emplid	dateofjoining	deptname	designation	emplname	projects	salary
105	2020-09-10 18:30:00.000000+0000	dept2	designation5	employee5	null	15000 121	
	2020-10-17 18:30:00.000000+0000	dept2	designation6	employee7	null	45000 104	2020-12-01 18:30:00.000000+0000   dept2
			designation4	employee4	null	90000 102	2020-06-03

```
18:30:00.000000+0000 | dept1 | designation2 | employee2 | null | 60000 101 | 2020-03-28 18:30:00.000000+0000 | dept1 | designation1 |  
employee1 | null | 40000 103 | 2020-04-20 18:30:00.000000+0000 | dept1 | designation3 | employee3 | null | 75000
```

7. Create a TTL of 15 seconds to display the values of Employees.

```
insert into EmployeeInfo (Emp_id, Emp_name, Designation, DOJ, salary, Dept_name) values  
(161,'Ryan','Associate professor','2022-05-11',95000,'ISE') using ttl 60;
```

```
select ttl(Emp_name) from Employee_info where Emp_id = 161 and salary =  
95000;
```

```
ttl(emp_name)
```

```
-----
```

```
53
```

```
(1 rows)
```

```
11
```

## Program 03: Cassandra Library Database

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

1. Create a keyspace by name Library create keyspace

```
libInfo with replication = {  
    ... 'class': 'SimpleStrategy',  
    ... 'replication_factor': 1 ... }; use  
libInfo;
```

2. Create a column family by name Library-Info with attributes Stud\_Id Primary

```
Key, Counter_value of type Counter create table libInfo (  
    ... studID int,  
    ... studName text, ... bookID  
    int,  
    ... bookName text,  
    ... dateOfIssue timestamp,  
    ... counterValue counter,  
    ... primary key ((studID, bookID), studName, bookName,  
dateOfIssue)  
    ... );
```

3. Insert the values into the table in batch update

```
libInfo  
  
    ... set counterValue=counterValue+1  
    ... where studID = 001 and studName = 'Raj' and bookID  
= 101 and bookName = 'The Midnight Library' and dateOfIssue =  
'2023-05-08';  
  
update libInfo  
    ... set counterValue=counterValue+1  
    ... where studID = 002 and studName = 'Krishna' and bookID = 102 and  
bookName = 'The Little Coffee Shop of Kabul' and dateOfIssue = '2023-03-07';  
update libInfo
```

```

        ... set counterValue=counterValue+1
        ... where studID = 003 and studName = 'Trupti' and bookID
= 103 and bookName = 'Tokyo Ueno Station' and dateOfIssue =
'2022-12-26';

```

update libInfo

```

        ... set counterValue=counterValue+1
        ... where studID = 004 and studName = 'Arya' and bookID =
104 and bookName = 'A Thousand Splendid Suns' and dateOfIssue =
'2022-10-03';

```

update libInfo

```

        ... set counterValue=counterValue+1
        ... where studID = 005 and studName = 'Karan' and bookID =
105 and bookName = 'Portrait of an Unknown Woman' and dateOfIssue =
'2023-01-28';

```

4. Display the details of the table created and increase the value of the counter select  
\* from libInfo;

```

studid | bookid | studname | bookname | dateofissue | countervalue
-----+-----+-----+-----+-----+-----
1 | 101 | Raj | The Midnight Library | 2023-05-07 18:30:00.000000+0000 | 1
3 | 103 | Trupti | Tokyo Ueno Station | 2022-12-25 18:30:00.000000+0000 | 1
2 | 102 | Krishna | The Little Coffee Shop of Kabul | 2023-03-06 18:30:00.000000+0000 | 1
5 | 105 | Karan | Portrait of an Unknown Woman | 2023-01-27 18:30:00.000000+0000 | 1
4 | 104 | Arya | A Thousand Splendid Suns | 2022-10-02 18:30:00.000000+0000 | 1

```

update libInfo

```

        ... set counterValue=counterValue+1
        ... where studID = 005 and studName = 'Karan' and bookID =
105 and bookName = 'Portrait of an Unknown Woman' and dateOfIssue =
'2023-01-28'; select * from
libInfo;

```

```

studid | bookid | studname | bookname | dateofissue | countervalue -----+-----+-----+-----+-----+-----
-----+-----

```

```

1 | 101 | Raj | The Midnight Library | 2023-05-07 18:30:00.000000+0000 | 1
3 | 103 | Trupti | Tokyo Ueno Station | 2022-12-25 18:30:00.000000+0000 | 1
2 | 102 | Krishna | The Little Coffee Shop of Kabul | 2023-03-06 18:30:00.000000+0000 | 1
5 | 105 | Karan | Portrait of an Unknown Woman | 2023-01-27 18:30:00.000000+0000 | 2
4 | 104 | Arya | A Thousand Splendid Suns | 2022-10-02 18:30:00.000000+0000 | 1

```

5. Write a query to show that a student with id 114 has taken a book “UNIX” 2 times.

```

select studID from libInfo where bookName = 'Portrait of an Unknown Woman' and
counterValue = 2 allow filtering;

```

```

studid
-----
      5

```

6. Export the created column to a csv file copy libInfo(studID, studName, bookID, bookName, dateOfIssue, counterValue) to 'c:\libInfo.csv'; Using 3 child processes

Starting copy of libinfo.libinfo with columns [studid, studname, bookid, bookname, dateofissue, countervalue].

Processed: 5 rows; Rate: 2 rows/s; Avg. rate: 1 rows/s 5 rows exported to 1 files in 9.163 seconds.

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

```

truncate library_info;

```

```

select * from library_info;

```

```

studid | bookid | studname | bookname | dateofissue | countervalue
-----+-----+-----+-----+-----+-----
(0 rows)

```

```

copy libInfo(studID, studName, bookID, bookName, dateOfIssue,
counterValue) to 'c:\libInfo.csv'; Using 3 child
processes

```

Starting copy of libinfo.libinfo with columns [studid, studname, bookid, bookname, dateofissue, countervalue].

Processed: 5 rows; Rate: 2 rows/s; Avg. rate: 1 rows/s 5 rows exported to 1 files in 9.163 seconds.

## Program 04: Hadoop Commands

```
$start-all.sh
```

WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.

WARNING: This is not a recommended production deployment configuration.

WARNING: Use CTRL-C to abort.

Starting namenodes on [localhost]

Starting datanodes

Starting secondary namenodes

[bmscece-HP-Elite-Tower-600-G9-Desktop-PC]

Starting resourcemanager Starting  
nodemanagers

```
#to check all daemons have loaded successfully $jps
```

```
9056 Jps
```

```
7475 ResourceManager
```

```
6709 NameNode
```

```
7160 SecondaryNameNode
```

```
7659 NodeManager
```

```
6875 DataNode
```

```
#mkdir command hdfs dfs
```

```
-mkdir /bda
```

```
# ls command hadoop fs -ls / Found 4 items drwxr-xr-x - hadoop supergroup 0 2023-05-08 09:40 /abc drwxr-xr-x - hadoop supergroup 0 2023-05-11 13:57 /bda drwxr-xr-x - hadoop supergroup 0 2023-05-04 12:49 /inputbda
```

```
drwxr-xr-x - hadoop supergroup 0 2023-04-27 11:44 /sridevi
```

```
# to append text in a file in hdfs echo "<Text to append>" | hdfs dfs -  
appendToFile /user/hduser/myfile.txt OR
```

hdfs dfs -appendToFile - /user/hduser/myfile.txt and then type the text on the terminal. Once you are done typing then hit 'Ctrl+D'



```
#cat command echo "hello world bda lab" | hdfs dfs -appendToFile - /bda/hello.txt
```

```
hdfs dfs -cat /bda/hello.txt hello world bda  
lab
```

```
#put & copyFromLocal command hdfs dfs -put Desktop/hadooplocal.txt  
/bda/hadoop.txt hdfs dfs -copyFromLocal Desktop/hadooplocal.txt /bda/hadoop.txt
```

```
hdfs dfs -cat /bda/hadoop.txt local file created  
in the desktop
```

```
# get command hdfs dfs -touchz  
/bda/labfile.txt
```

```
echo "copying hdfs file to a local file using get command" | hdfs dfs  
-appendToFile - /bda/labfile.txt
```

```
hdfs dfs -cat /bda/labfile.txt copying hdfs file to a local file using get  
command
```

```
hdfs dfs -get /bda/labfile.txt Desktop/getcmd.txt #Contents of getcmd.txt file in  
Desktop is:
```

```
copying hdfs file to a local file using get command
```

```
#copyToLocal command hdfs dfs -touchz
```

```
/bda/ghost.txt echo "new hdfs file in hdfs
```

```
folder" | hdfs dfs -appendToFile -
```

```
/bda/ghost.txt
```

```
hdfs dfs -cat /bda/ghost.txt new hdfs file in hdfs folder hdfs dfs -  
copyToLocal /bda/ghost.txt Desktop/bigdata.txt
```

#Contents of bigdata.txt file in desktop is: new hdfs file in hdfs

folder

#mv command hdfs

dfs -ls /bda

Found 4 items

-rw-r--r-- 1 hadoop supergroup 29 2023-05-11 14:39 /bda/ghost.txt

-rw-r--r-- 1 hadoop supergroup 34 2023-05-11 14:26 /bda/hadoop.txt

-rw-r--r-- 1 hadoop supergroup 20 2023-05-11 14:11 /bda/hello.txt

-rw-r--r-- 1 hadoop supergroup 52 2023-05-11 14:32

/bda/labfile.txt hadoop fs -mv /bda/hello.txt

/dir

hdfs dfs -ls /bda

Found 3 items

-rw-r--r-- 1 hadoop supergroup 29 2023-05-11 14:39 /bda/ghost.txt

-rw-r--r-- 1 hadoop supergroup 34 2023-05-11 14:26 /bda/hadoop.txt

-rw-r--r-- 1 hadoop supergroup 52 2023-05-11 14:32

/bda/labfile.txt hdfs dfs -ls

/dir -rw-r--r-- 1 hadoop

supergroup 20 2023-05-11

14:11 /dir

#cp command hadoop fs -cp

/bda /rest

hdfs dfs -ls /bda

Found 3 items

-rw-r--r-- 1 hadoop supergroup 29 2023-05-11 14:39 /bda/ghost.txt

-rw-r--r-- 1 hadoop supergroup 34 2023-05-11 14:26 /bda/hadoop.txt

```
-rw-r--r-- 1 hadoop supergroup 52 2023-05-11 14:32  
/bda/labfile.txt
```

```
hdfs dfs -ls /rest
```

```
Found 3 items
```

```
-rw-r--r-- 1 hadoop supergroup 29 2023-05-11 14:50 /rest/ghost.txt  
-rw-r--r-- 1 hadoop supergroup 34 2023-05-11 14:50 /rest/hadoop.txt  
-rw-r--r-- 1 hadoop supergroup 52 2023-05-11 14:50  
/rest/labfile.txt
```

## Program 05: Word Count Program in Hadoop

WCDriver.java

```
//Importing libraries
```

```
import java.io.IOException; import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path; import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text; import
org.apache.hadoop.mapred.FileInputFormat; import
org.apache.hadoop.mapred.FileOutputFormat; import
org.apache.hadoop.mapred.JobClient; import
org.apache.hadoop.mapred.JobConf; import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner; public class WCDriver extends
Configured implements Tool {

    public int run(String args[]) throws IOException
    { if (args.length < 2)
        {
            System.out.println("Please give valid inputs"); return -1;
        }

        JobConf conf = new JobConf(WCDriver.class);
        FileInputFormat.setInputPaths(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(WCMapper.class);
        conf.setReducerClass(WCReducer.class);
        conf.setMapOutputKeyClass(Text.class);
        conf.setMapOutputValueClass(IntWritable.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        JobClient.runJob(conf); return 0;
    }

    // Main Method public static void main(String args[]) throws Exception
    { int exitCode = ToolRunner.run(new WCDriver(), args);
      System.out.println(exitCode);
    }
}
```

```
    }  
}
```

### WCMapper.java

```
// Importing libraries import java.io.IOException; import  
org.apache.hadoop.io.IntWritable; import  
org.apache.hadoop.io.LongWritable; import  
org.apache.hadoop.io.Text; import  
org.apache.hadoop.mapred.MapReduceBase; import  
org.apache.hadoop.mapred.Mapper; import  
org.apache.hadoop.mapred.OutputCollector; import  
org.apache.hadoop.mapred.Reporter;  
  
public class WCMapper extends MapReduceBase implements  
Mapper<LongWritable,Text, Text, IntWritable> {  
    // Map function public void map(LongWritable key, Text value,  
    OutputCollector<Text,  
        IntWritable> output, Reporter rep) throws IOException {  
  
        String line = value.toString();  
        // Splitting the line on spaces for (String word : line.split(" "))  
        { if (word.length() > 0)  
            { output.collect(new Text(word), new  
IntWritable(1));  
            }  
        }  
    }  
}
```

### WCReducer.java

```
// Importing libraries import java.io.IOException; import  
java.util.Iterator; import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.Text; import  
org.apache.hadoop.mapred.MapReduceBase; import  
org.apache.hadoop.mapred.OutputCollector; import
```

```
org.apache.hadoop.mapred.Reducer; import
org.apache.hadoop.mapred.Reporter;
```

```
public class WCReducer extends MapReduceBase implements
Reducer<Text,IntWritable, Text, IntWritable> {
// Reduce function public void reduce(Text key, Iterator<IntWritable> value,
OutputCollector<Text, IntWritable> output,Reporter rep) throws
IOException
```

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

## Output:

```
Found 2 items
-rw-r--r-- 1 hadoop supergroup 0 2023-06-24 18:18 /optemp/_SUCCESS
-rw-r--r-- 1 hadoop supergroup 0 2023-06-24 18:18 /optemp/part-r-000000
hadoop@bsccscce-WP-Elite-Tower-000-G3-Desktop-PC: $ hadoop fs -cat /optemp/part-r-000000
1901 46
hadoop@bsccscce-WP-Elite-Tower-000-G3-Desktop-PC: $ hadoop jar /home/hadoop/Desktop/AvgTemp.jar AvgDriver /iptemp/1902 /optemp
2023-06-24 11:03:30.175 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2023-06-24 11:03:30.214 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2023-06-24 11:03:30.214 INFO impl.MetricsSystemImpl: JobTracker metrics system started
Exception in thread "main" org.apache.hadoop.mapred.FileAlreadyExistsException: Output directory hdfs://localhost:9000/optemp already exists
    at org.apache.hadoop.mapreduce.lib.output.FileOutputFormat.checkOutputSpecs(FileOutputFormat.java:164)
    at org.apache.hadoop.mapreduce.JobSubmitter.checkSpecs(JobSubmitter.java:277)
    at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:143)
    at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1571)
    at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1568)
    at java.base/java.security.AccessController.doPrivileged(Native Method)
    at java.base/javax.security.auth.Subject.doAs(Subject.java:423)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1878)
    at org.apache.hadoop.mapreduce.Job.submit(Job.java:1568)
    at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1589)
    at AvgDriver.main(AvgDriver.java:22)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.base/java.lang.reflect.Method.invoke(Method.java:566)
    at org.apache.hadoop.util.RunJar.run(RunJar.java:323)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:236)
hadoop@bsccscce-WP-Elite-Tower-000-G3-Desktop-PC: $ hadoop jar /home/hadoop/Desktop/AvgTemp.jar AvgDriver /iptemp/1902 /optemp1
2023-06-24 11:03:34.060 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2023-06-24 11:03:34.100 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2023-06-24 11:03:34.100 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2023-06-24 11:03:34.162 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2023-06-24 11:03:34.198 INFO input.FileInputFormat: Total input files to process : 1
2023-06-24 11:03:34.225 INFO mapreduce.JobSubmitter: number of splits:1
2023-06-24 11:03:34.287 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1935691243_0001
2023-06-24 11:03:34.287 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-06-24 11:03:34.349 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2023-06-24 11:03:34.349 INFO mapreduce.Job: Running job: job_local1935691243_0001
2023-06-24 11:03:34.350 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2023-06-24 11:03:34.354 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-24 11:03:34.354 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2023-06-24 11:03:34.354 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2023-06-24 11:03:34.391 INFO mapred.LocalJobRunner: Waiting for map tasks
2023-06-24 11:03:34.391 INFO mapred.LocalJobRunner: Starting task: attempt local1935691243_0001_m_000000_0
2023-06-24 11:03:34.400 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-24 11:03:34.401 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2023-06-24 11:03:34.407 INFO mapred.Task: Using ResourceCalculatorProcessTree: [1]
2023-06-24 11:03:34.408 INFO mapred.MapTask: Processing split: hdfs://localhost:9000/iptemp/1902:0+888978
2023-06-24 11:03:34.441 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
2023-06-24 11:03:34.441 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
2023-06-24 11:03:34.441 INFO mapred.MapTask: soft limit at 838860800
2023-06-24 11:03:34.441 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
2023-06-24 11:03:34.441 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
2023-06-24 11:03:34.443 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
```

## Program 06: Average Temperature

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

```
AverageMapper.java package temp;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class AverageMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
    public static final int MISSING = 9999;
    public void map(LongWritable key, Text value, Mapper<LongWritable,
    Text, Text, IntWritable>.Context context) throws IOException,
    InterruptedException
    {
        int temperature;
```

```

String line = value.toString(); String year =
line.substring(15, 19); if (line.charAt(87) ==
'&#39;+&#39;) {
    temperature = Integer.parseInt(line.substring(88, 92));
} else {
    temperature = Integer.parseInt(line.substring(87, 92)); }
String quality = line.substring(92, 93); if (temperature != 9999 &&
quality.matches("[01459]"))
context.write(new Text(year), new IntWritable(temperature));
}
}

```

```

AverageReducer.java package temp; import
java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class AverageReducer extends Reducer<Text, IntWritable, Text,
IntWritable> {

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

```



## Output:

```
Bytes Written=0
2023-06-24 11:03:34,740 INFO mapred.LocalJobRunner: Finishing task: attempt local1935691243_0001_r_000000_0
2023-06-24 11:03:34,740 INFO mapred.LocalJobRunner: reduce task executor complete.
2023-06-24 11:03:35,352 INFO mapreduce.Job: Job job local1935691243_0001 running in uber mode : false
2023-06-24 11:03:35,354 INFO mapreduce.Job: map 100% reduce 100%
2023-06-24 11:03:35,355 INFO mapreduce.Job: Job job local1935691243_0001 completed successfully
2023-06-24 11:03:35,369 INFO mapreduce.Job: Counters: 36
  File System Counters
    FILE: Number of bytes read=153042
    FILE: Number of bytes written=1504567
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=1777956
    HDFS: Number of bytes written=8
    HDFS: Number of read operations=15
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=4
    HDFS: Number of bytes read erasure-coded=0
  Map-Reduce Framework
    Map input records=6565
    Map output records=6565
    Map output bytes=59085
    Map output materialized bytes=72221
    Input split bytes=98
    Combine input records=0
    Combine output records=0
    Reduce input groups=1
    Reduce shuffle bytes=72221
    Reduce input records=6565
    Reduce output records=1
    Spilled Records=13130
    Shuffled Maps =1
    Failed Shuffles=0
    Merged Map outputs=1
    GC time elapsed (ms)=7
    Total committed heap usage (bytes)=1159725056
  Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
  File Input Format Counters
    Bytes Read=888978
  File Output Format Counters
    Bytes Written=0
hadoop@bmscece-HP-Elite-Tower-600-G9-Desktop-PC: $ hadoop fs -ls /optempl
Found 2 items
-rw-r--r--  1 hadoop supergroup          0 2023-06-24 11:03 /optempl/ SUCCESS
-rw-r--r--  1 hadoop supergroup          0 2023-06-24 11:03 /optempl/part-r-00000
hadoop@bmscece-HP-Elite-Tower-600-G9-Desktop-PC: $ hadoop fs -cat /optempl/part-r-00000
1902  21
```

## Program 07: Mean Max Temperature in Hadoop

```
MeanMaxDriver.java package meanmax;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

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

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

## MeanMaxMapper.java

```
import org.apache.hadoop.mapreduce.Mapper;

public class MeanMaxMapper extends Mapper<LongWritable, Text, Text,
IntWritable>
{
    public static final int MISSING = 9999;
    public void map(LongWritable key, Text value,
Mapper>LongWritable, Text, Text, IntWritable>.Context context) throws
IOException, InterruptedException
    {
        int temperature;
        String line = value.toString();
        String month = line.substring(19, 21);
        if (line.charAt(87) == '&#39;+&#39;')
        {
            temperature = Integer.parseInt(line.substring(88, 92));
        }
        Else
        {
            temperature = Integer.parseInt(line.substring(87, 92));
        }
        String quality = line.substring(92, 93);

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

    }

}
```

MeanMaxReducer.java package

```
meanmax;
```

```
import java.io.IOException;
```

```
import org.apache.hadoop.io.IntWritable;
```

```
import org.apache.hadoop.io.Text;
```

```
import org.apache.hadoop.mapreduce.Reducer;
```

```
public class MeanMaxReducer extends <Text, IntWritable, Text, IntWritable> {  
    public void reduce(Text key, Iterable<IntWritable> values, Reducer<Text, IntWritable,  
    Text, IntWritable>.Context context) throws IOException,  
    InterruptedException { int  
        max_temp = 0;  
        int total_temp = 0; int  
        count = 0; int days = 0;  
  
        for (IntWritable value : values) {  
            int temp = value.get(); if (temp >  
            max_temp)  
                max_temp = temp; count++;  
            if (count == 3) { total_temp +=  
                max_temp; max_temp = 0;  
                count = 0; days++;  
            }  
        } context.write(key, new IntWritable(total_temp / days)); }  
}
```

## Output:

```
hadoop@bmscscse-HP-Elite-Tower-600-G9-Desktop-PC:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
localhost: namenode is running as process 7391. Stop it first and ensure /tmp/hadoop-hadoop-namenode.pid file is empty before retry.
Starting datanodes
localhost: datanode is running as process 7570. Stop it first and ensure /tmp/hadoop-hadoop-datanode.pid file is empty before retry.
Starting secondary namenodes [bmscscse-HP-Elite-Tower-600-G9-Desktop-PC]
bmscscse-HP-Elite-Tower-600-G9-Desktop-PC: secondarynamenode is running as process 7865. Stop it first and ensure /tmp/hadoop-hadoop-secondarynamenode.pid file is empty before retry.
Starting resource manager
resource manager is running as process 8150. Stop it first and ensure /tmp/hadoop-hadoop-resourcemanager.pid file is empty before retry.
Starting node managers
localhost: nodemanager is running as process 8337. Stop it first and ensure /tmp/hadoop-hadoop-nodemanager.pid file is empty before retry.
hadoop@bmscscse-HP-Elite-Tower-600-G9-Desktop-PC:~$ hadoop fs -mkdir /iptemp
hadoop@bmscscse-HP-Elite-Tower-600-G9-Desktop-PC:~$ hadoop fs -put /home/hadoop/Desktop/1901 /iptemp
hadoop@bmscscse-HP-Elite-Tower-600-G9-Desktop-PC:~$ hadoop jar /home/hadoop/Desktop/AvroTemp.jar AvroDriver /iptemp/1901/optemp
2023-06-24 10:18:36,257 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2023-06-24 10:18:36,297 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2023-06-24 10:18:36,297 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2023-06-24 10:18:36,357 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2023-06-24 10:18:36,396 INFO input.FileInputFormat: Total input files to process : 1
2023-06-24 10:18:36,423 INFO mapreduce.JobSubmitter: number of splits:1
2023-06-24 10:18:36,484 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1783357305_0001
2023-06-24 10:18:36,484 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-06-24 10:18:36,543 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2023-06-24 10:18:36,544 INFO mapreduce.Job: Running job: job_local1783357305_0001
2023-06-24 10:18:36,544 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2023-06-24 10:18:36,548 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-24 10:18:36,548 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2023-06-24 10:18:36,548 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2023-06-24 10:18:36,598 INFO mapred.LocalJobRunner: Waiting for map tasks
2023-06-24 10:18:36,598 INFO mapred.LocalJobRunner: Starting task: attempt_local1783357305_0001_m_000000_0
2023-06-24 10:18:36,611 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-24 10:18:36,611 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2023-06-24 10:18:36,617 INFO mapred.Task: Using ResourceCalculatorProcessTree : [ ]
2023-06-24 10:18:36,618 INFO mapred.MapTask: Processing split: hdfs://localhost:9000/iptemp/1901:0+888190
2023-06-24 10:18:36,650 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
2023-06-24 10:18:36,650 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
2023-06-24 10:18:36,650 INFO mapred.MapTask: soft limit at 83886080
2023-06-24 10:18:36,650 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
2023-06-24 10:18:36,650 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
2023-06-24 10:18:36,652 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
2023-06-24 10:18:36,727 INFO mapred.LocalJobRunner:
2023-06-24 10:18:36,728 INFO mapred.MapTask: Starting flush of map output
2023-06-24 10:18:36,728 INFO mapred.MapTask: Spilling map output
2023-06-24 10:18:36,728 INFO mapred.MapTask: bufstart = 0; bufend = 59076; bufvoid = 104857600
2023-06-24 10:18:36,728 INFO mapred.MapTask: kvstart = 26214396(104857584); kvend = 26188144(104752576); length = 26253/6553600
2023-06-24 10:18:36,737 INFO mapred.MapTask: Finished spill 0
2023-06-24 10:18:36,741 INFO mapred.Task: Task:attempt_local1783357305_0001_m_000000_0 is done. And is in the process of committing
2023-06-24 10:18:36,743 INFO mapred.Task: Task:attempt_local1783357305_0001_m_000000_0' done.
2023-06-24 10:18:36,746 INFO mapred.Task: Final Counters for attempt_local1783357305_0001_m_000000_0: Counters: 23
```

## **Program 08: 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.libMultipleInputs;

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, Textvalue, 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  
#39;Department Emp Strength input#39; with  
#39;Department Name input#39;");  
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

```
} 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 + &quot;\t&quot; + 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);
}
}

}

```

## Output:

```

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=0
File Output Format Counters
Bytes Written=85
hduser@bnsce-Precision-T1700:~/khushil/Join/MapReduceJoin$ hdfs dfs -cat /khushil_join/output2/part-00000
A11      50      Finance
B12      100     HR
C13      250     Manufacturing
Dept_ID Total_Employee      Dept_Name
hduser@bnsce-Precision-T1700:~/khushil/Join/MapReduceJoin$

```

## Program 09: Word Count in Spark

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

```
scala> data.collect; res0: Array[String] = Array(hello world, this is BDA spark lab)
```

```
scala> val splitdata = data.flatMap(line => line.split(" "));  
splitdata: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at flatMap at  
<console>:25
```

```
scala> splitdata.collect;  
res1: Array[String] = Array(hello, world,, this, is, BDA, spark, lab)
```

```
scala> val mapdata = splitdata.map(word => (word,1));  
mapdata: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at map at  
<console>:25
```

```
scala> mapdata.collect; res2: Array[(String, Int)] = Array((hello,1), (world,,1), (this,1),  
(is,1), (BDA,1), (spark,1), (lab,1))
```

```
scala> val reducedata = mapdata.reduceByKey(_+_); reducedata:  
org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:25
```

```
scala> reducedata.collect; res3: Array[(String, Int)] = Array((this,1), (is,1), (hello,1),  
(world,,1), (lab,1), (spark,1), (BDA,1))
```

**Program 10: Using RDD and FlatMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark.**

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

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

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

```
scala> val sorted=ListMap(counts.collect.sortWith(_._2 > _. _2):_*)// sort in descending order
based
sorted: scala.collection.immutable.ListMap[String,Int] =
ListMap(hello -> 6, world -> 5, this -> 2, is -> 2, lab -> 2, BDA ->
2, word -> 1)
```

```
scala> println(sorted)
ListMap(hello -> 6, world -> 5, this -> 2, is -> 2, lab -> 2, BDA ->
2, word -> 1)
```

```
scala> for((k,v)<-sorted){
  | if(v>4)
  | {
  | print(k+",")
  | print(v)
  | println()
  | }
  | }
hello,6
```