

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



## LAB REPORT

on

## BDA LAB REPORT

*Submitted by*

**Preetham Kolli**

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

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**



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

(Autonomous Institution under VTU)

**BENGALURU-560019**

**May-2023 to July-2023**

**B. M. S. College of Engineering,**  
**Bull Temple Road, Bangalore 560019**  
(Affiliated To Visvesvaraya Technological University, Belgaum)  
**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the Lab work entitled “**BIG DATA ANALYTICS**” carried out by **Preetham Kolli (1BM20CS114)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics - (20CS6PEBDA)** work prescribed for the said degree.

Vikranth BM  
Assistant Professor  
Department of CSE  
BMSCE, Bengaluru

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

# Index Sheet

Sl.No.	Experiment Title
1	Cassandra Lab Program1: - Employee Database
2	Cassandra Lab Program1: - Library Database
3	MongoDB- CRUD Demonstration
4	Hadoop installation
5	Hadoop Commands
6	Hadoop Program: Average Temperature
7	Hadoop Program: Word Count
8	Hadoop program: Join operation
9	Scal Program
10	Scala Program: Word Count

## Course Outcome

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

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

**1. Create a keyspace by name Employee**

**2. Create a column family by**

**name Employee-Info with**

**attributes Emp\_Id Primary Key,**

**Emp\_Name,**

**Designation, Date\_of\_Joining, Salary, Dept\_Name**

**3. Insert the values into the table in batch**

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

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

**6. Alter the schema of the table Employee\_Info to add a column Projects which stores a set of**

**Projects done by the corresponding Employee.**

**7. Update the altered table to add project names.**

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

```
cqlsh:employee> CREATE KEYSPACE employee WITH REPLICATION={ 'class': 'SimpleStrategy', 'replication_factor' : 1};
```

```
cqlsh:employee> USE employee;
```

```
cqlsh:employee> create table employee_info(emp_id int PRIMARY KEY, emp_name text, ... designation text, date_of_joining timestamp, salary double PRIMARY KEY, dept_name text);
```

```
cqlsh:employee> CREATE TABLE employee_info(emp_id int, emp_name text, designation text, date_of_joining timestamp, salary double, dept_name text, PRIMARY KEY(emp_id, salary));
```

```
cqlsh:employee> BEGIN BATCH INSERT INTO
```

...

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

```
VALUES(100,'John','MANAGER','2021-09-11',30000,'TESTING');
```

```
... INSERT INTO
```

```
...
```

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

```
VALUES(111,'Tom','ASSOCIATE','2021-06-22',25000,'DEVELOPING');
```

```
... INSERT INTO
```

```
...
```

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

```
VALUES(121,'Elsa','MANAGER','2021-03-30',35000,'HR');
```

```
... INSERT INTO
```

```
...
```

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

```
VALUES(115,'Chris','ASSISTANT','2021-12-30',20000,'DEVELOPING');
```

```
... INSERT INTO
```

```
...
```

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

```
VALUES(105,'Sarah','ASSOCIATE','2021-06-25',25000,'TESTING');
```

```
... APPLY BATCH;
```

```
cqlsh:employee> SELECT * FROM employee_info
```

```
... ;
```

```
cqlsh:employee> UPDATE employee_info SET emp_name = 'Jessica', dept_name =  
'DEVELOPING' WHERE emp_id = 121;
```

```
cqlsh:employee> UPDATE employee_info SET emp_name = 'Jessica', dept_name =  
'DEVELOPING' WHERE emp_id = 121 AND salary = 35000;
```

```
cqlsh:employee> SELECT * FROM employee_info ;  
MANAGER | John
```

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

Disabled Query paging.

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

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

```
cqlsh:employee> UPDATE employee_info SET projects = 'Chat App' WHERE emp_id = 111;
```

```
cqlsh:employee> UPDATE employee_info SET projects = 'Chat App' WHERE emp_id = 111  
and salary = 25000;
```

```
cqlsh:employee> UPDATE employee_info SET projects = 'Discord Bot' WHERE emp_id  
= 115 and salary = 20000;
```

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

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

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

```
cqlsh:employee> SELECT * FROM employee_infor  
... ;
```

```
cqlsh:employee> SELECT * FROM employee_info ;
```

```
cqlsh:employee> INSERT INTO
```

...

employee\_info(emp\_id,emp\_name,designation,date\_of\_joining,salary,dept\_name) ...

... ;

cqlsh:employee> INSERT INTO

...

employee\_info(emp\_id,emp\_name,designation,date\_of\_joining,salary,dept\_name) ...

VALUES(110,'SAM','ASSOCIATE','2021-01-11',28000,'TESTING')

USING TTL 15;

cqlsh:employee> SELECT TTL(emp\_name) from employee\_info

WHERE emp\_id = 110; ttl(emp\_name)

3

cqlsh:employee> SELECT \* FROM employee\_info;



## Output:

```
Activities Terminal Apr 29 10:43 bmsce@bmsce-Precision-T1700 ~
cqlsh:employee> insert into EmployeeInfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('jal',121,'manager','2022-03-13',40000,'cs');
cqlsh:employee> APPLY BATCH
... Insert into EmployeeInfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('jal',121,'manager','2022-03-13',40000,'cs');
cqlsh:employee> BEGIN BATCH insert into EmployeeInfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('jal',121,'manager','2022-03-13',40000,'cs');
... Insert into EmployeeInfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('nani',131,'STAFF','2022-03-13',40000,'MARKETING');
... Insert into EmployeeInfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('mont',141,'STAFF','2022-05-23',80000,'MARKETING');
... APPLY BATCH;
cqlsh:employee> SELECT * from employeeinfo
... ;

emp_id | date_of_joining | dept_name | designation | emp_name | salary
-----|-----|-----|-----|-----|-----
121 | 2022-03-12 18:30:00.000000+0000 | cs | manager | jal | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | mont | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | nani | 40000
(3 rows)
cqlsh:employee> update employeeinfo set emp_name='neha' where emp_id=121;
cqlsh:employee> SELECT * from employeeinfo ;

emp_id | date_of_joining | dept_name | designation | emp_name | salary
-----|-----|-----|-----|-----|-----
121 | 2022-03-12 18:30:00.000000+0000 | cs | manager | neha | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | mont | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | nani | 40000
(3 rows)
cqlsh:employee> update employeeinfo set emp_name='neha',dept_name='staffing' where emp_id=121;
cqlsh:employee> SELECT * from employeeinfo ;

emp_id | date_of_joining | dept_name | designation | emp_name | salary
-----|-----|-----|-----|-----|-----
121 | 2022-03-12 18:30:00.000000+0000 | staffing | manager | neha | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | mont | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | nani | 40000
(3 rows)
cqlsh:employee> SELECT *
... SELECT * from employeeinfo ;
cqlsh:employee> SELECT * from employeeinfo ;
cqlsh:employee> SELECT * from employeeinfo ;

emp_id | date_of_joining | dept_name | designation | emp_name | salary
-----|-----|-----|-----|-----|-----
121 | 2022-03-12 18:30:00.000000+0000 | staffing | manager | neha | 40000
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | mont | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | nani | 40000
(3 rows)
cqlsh:employee> SELECT * from employeeinfo order by salary;
cqlsh:employee> create index on employeeinfo(salary);
cqlsh:employee> SELECT * from employeeinfo order by salary;
cqlsh:employee> SELECT * from employeeinfo order by salary;
```

```
Activities Terminal Apr 29 10:44 bmsce@bmsce-Precision-T1700 ~
141 | 2022-05-22 18:30:00.000000+0000 | MARKETING | STAFF | mont | null | 80000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | nani | null | 40000
(3 rows)
cqlsh:employee> select TTL(*)
... from employeeinfo
... ;
cqlsh:employee> insert into EmployeeInfo (emp_name,emp_ID,designation,date_of_joining,salary,dept_name) values('seena',141,'STAFF','2021-03-13',50000,'MARKETING') using ttl 15;
cqlsh:employee> select ttl(*) from employeeinfo;
cqlsh:employee> select ttl(emp_name,dept_name) from employeeinfo;
cqlsh:employee> select ttl(emp_name) from employeeinfo;

ttl(emp_name)
-----
null
(2 rows)
cqlsh:employee> select ttl(emp_name,emp_ID) from employeeinfo;
cqlsh:employee> select ttl(emp_name) from employeeinfo;

ttl(emp_name)
-----
null
(2 rows)
cqlsh:employee> cd ..
...
...
...
Incomplete statement at end of file
bmsce@bmsce-Precision-T1700:~$ cqlsh
Connected to test cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.4 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> use employee;
... ;
cqlsh:employee> select * from employeeinfo;

emp_id | date_of_joining | dept_name | designation | emp_name | projects | salary
-----|-----|-----|-----|-----|-----|-----
121 | 2022-03-12 18:30:00.000000+0000 | staffing | manager | neha | ['ai ml', 'cyber security', 'data science'] | 40000
131 | 2022-03-12 18:30:00.000000+0000 | MARKETING | STAFF | nani | null | 40000
(2 rows)
cqlsh:employee> create index on employeeinfo(emp_name);
cqlsh:employee> select * from employeeinfo where emp_name='neha';
cqlsh:employee> select * from employeeinfo where emp_name='neha';
cqlsh:employee> select * from employeeinfo where emp_name='neha';

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

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

**1.Create a keyspace by name Library**

**2. Create a column family by name Library-Info with**

**attributes Stud\_Id Primary Key, Counter\_value of type**

**Counter, Stud\_Name, Book-Name, Book-Id, Date\_of\_issue**

**3. Insert the values into the table in batch**

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

**the counter 5. Write a query to show that a student with id 112 has**

**taken a book “BDA” 2 times. 6. Export the created column to a csv**

**file**

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

**family** cqlsh:library> CREATE KEYSPACE library WITH replication = {'class':

'SimpleStrategy','replication\_factor':1}; cqlsh:library> USE library ;

cqlsh:library> CREATE TABLE Library\_info(stud\_id int, stud\_name text, book\_name text,  
book\_id text, date\_of\_issue timestamp, counter\_value counter, PRIMARY  
KEY(stud\_id,stud\_name, book\_name, book\_id, date\_of\_issue));

cqlsh:library> BEGIN COUNTER BATCH

... UPDATE library\_info set counter\_value +=1 where stud\_id = 111 and  
stud\_name = 'Manoj' and book\_name = 'Operations Research' and book\_id = '56TXT'  
and date\_of\_issue = '2021-09-12';

... UPDATE library\_info set counter\_value +=1 where stud\_id = 112 and  
stud\_name = 'Kamal' and book\_name = 'Engineering Mathematics-3' and book\_id =  
'5ERW4' and date\_of\_issue = '2021-04-10';

... UPDATE library\_info set counter\_value +=1 where stud\_id = 113 and  
stud\_name = 'Mahesh' and book\_name = 'Robinson Crusoe' and book\_id = '34EDC' and  
date\_of\_issue = '2021-02-01';

... UPDATE library\_info set counter\_value +=1 where stud\_id = 114 and  
stud\_name = 'Raj' and book\_name = 'Engineering Drawing' and book\_id = '123ER'  
and date\_of\_issue = '2021-04-03';

... APPLY BATCH;

cqlsh:library> SELECT \* FROM library\_info ;

cqlsh:library> UPDATE library\_info set counter\_value += 1 where stud\_id = 112 and stud\_name = 'Kamal' and book\_name = 'Engineering Mathematics-3' and book\_id = '5ERW4' and date\_of\_issue = '2021-04-09';

cqlsh:library> SELECT \* FROM library\_info ;

```
cqlsh> use library;
cqlsh:library> create table library_info (
    ... .. stud_id int,
    ... .. stud_name text,
    ... .. book_id int,
    ... .. book_name text,
    ... .. date_of_issue timestamp,
```

```
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 112 and stud_name = 'Ram' and book_id = 200 and book_name = 'DSA' and date_of_issue = '2022-04-06';
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 113 and stud_name = 'sohan' and book_id = 300 and book_name = 'JAVA' and date_of_issue = '2022-04-07';
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 111 and stud_name = 'Raj' and book_id = 100 and book_name = 'ADA' and date_of_issue = '2022-04-05';
cqlsh:library> update library_info set counter_value = counter_value+1 where stud_id = 114 and stud_name = 'rohan' and book_id = 400 and book_name = 'UNIX' and date_of_issue = '2022-04-07';
cqlsh:library> select * from library_info ;
```

stud_id	book_id	stud_name	book_name	date_of_issue	counter_value
114	400	rohan	UNIX	2022-04-06 18:30:00.000000+0000	1
111	100	Raj	ADA	2022-04-04 18:30:00.000000+0000	1
112	200	Ram	DSA	2022-04-05 18:30:00.000000+0000	1
113	300	sohan	JAVA	2022-04-06 18:30:00.000000+0000	1

(4 rows)

```

113 | 300 | sohan | JAVA | 2022-04-06 18:30:00.000000+0000 | 1
(4 rows)
cqlsh:library> copy library_info(stud_id,stud_name,book_id,book_name,date_of_issue,counter_value ) to '/home/bmscece/Documents/library/lib.csv' with header=true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_id, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate: 122 rows/s; Avg. rate: 122 rows/s
4 rows exported to 1 files in 0.069 seconds.
cqlsh:library> copy library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value ) to '/home/bmscece/Documents/library/lib.csv' with header=true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate: 129 rows/s; Avg. rate: 129 rows/s
4 rows exported to 1 files in 0.066 seconds.
cqlsh:library> copy library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value ) from '/home/bmscece/Documents/library/lib.csv' with header = true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate: 7 rows/s; Avg. rate: 11 rows/s
4 rows imported from 1 files in 0.375 seconds (0 skipped).
cqlsh:library> select * from library_info ;

stud_id | book_id | stud_name | book_name | date_of_issue | counter_value
-----|-----|-----|-----|-----|-----
114 | 400 | rohan | UNIX | 2022-04-06 18:30:00.000000+0000 | 2
111 | 100 | Raj | ADA | 2022-04-04 18:30:00.000000+0000 | 2
112 | 200 | Ran | DSA | 2022-04-05 18:30:00.000000+0000 | 2
113 | 300 | sohan | JAVA | 2022-04-06 18:30:00.000000+0000 | 2
(4 rows)
cqlsh:library> truncate library_info;
cqlsh:library> select * from library_info ;

stud_id | book_id | stud_name | book_name | date_of_issue | counter_value
-----|-----|-----|-----|-----|-----
(0 rows)
cqlsh:library> copy library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value ) from '/home/bmscece/Documents/library/lib.csv' with header = true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate: 8 rows/s; Avg. rate: 11 rows/s
4 rows imported from 1 files in 0.376 seconds (0 skipped).
cqlsh:library> select * from library_info ;

stud_id | book_id | stud_name | book_name | date_of_issue | counter_value
-----|-----|-----|-----|-----|-----
114 | 400 | rohan | UNIX | 2022-04-06 18:30:00.000000+0000 | 1
111 | 100 | Raj | ADA | 2022-04-04 18:30:00.000000+0000 | 1
112 | 200 | Ran | DSA | 2022-04-05 18:30:00.000000+0000 | 1
113 | 300 | sohan | JAVA | 2022-04-06 18:30:00.000000+0000 | 1
(4 rows)
cqlsh:library> 

```

### 3.MongoDB- CRUD Demonstration

bmsce@bmsce-Precision-T1700:~\$ mongo

MongoDB shell version v3.6.8

connecting to: mongodb://127.0.0.1:27017

Implicit session: session { "id" :

UUID("d66acdb3-8482-417d-8b75-d65dae4b53ee") } MongoDB

server version: 3.6.8

> use Student

switched to db Student

> db.createCollection("student");

{ "ok" : 1 }

>

db.Student.insert({\_id:1,StudName:"Megha",Grade:"vii",Hobbies:"InternetSurfing"}); WriteResult({ "nInserted" : 1 })

>

db.Student.update({\_id:3,StudName:"Ayan",Grade:"vii"},{\$set:{Hobbies:"skating"}},{upsert:true}); WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "\_id" : 3 })

> db.Student.find({StudName:"Ayan"})

{ "\_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }

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

{ "StudName" : "Megha", "Grade" : "vii" }

{ "Grade" : "vii", "StudName" : "Ayan" }

> db.Student.find({Grade:{\$eq:'vii'}}).pretty();

{

"\_id" : 1,

"StudName" : "Megha",

```

    "Grade" : "vii",
    "Hobbies" : "InternetSurfing"
  }
  { "_id" : 3, "Grade" : "vii", "StudName" : "Ayan",
    "Hobbies" : "skating" } >
  db.Student.find({Grade:{Seq:'vii'}});
  { "_id" : 1, "StudName" : "Megha", "Grade" : "vii", "Hobbies" :
    "InternetSurfing" } { "_id" : 3, "Grade" : "vii", "StudName" :
    "Ayan", "Hobbies" : "skating" } >
  db.Student.find({Grade:{Seq:'vii'}}).pretty();
  {
    "_id" : 1,
    "StudName" : "Megha",
    "Grade" : "vii",
    "Hobbies" : "InternetSurfing"
  }
  { "_id" : 3, "Grade" : "vii", "StudName" : "Ayan",
    "Hobbies" : "skating" } >
  db.Student.find({Hobbies:{$in:['Chess','Skating']}}).pr
etty(); >
  db.Student.find({Hobbies:{$in:['Skating']}}).pretty();
  > db.Student.find({Hobbies:{$in:['skating']}}).pretty();
  { "_id" : 3, "Grade" : "vii", "StudName" : "Ayan", "Hobbies" : "skating" }
  > db.Student.find({StudName:/^M/}).pretty();
  {
    "_id" : 1,
    "StudName" : "Megha",

```

```

        "Grade" : "vii",
        "Hobbies" : "InternetSurfing"
    }
> db.Student.find({StudName:/e/}).pretty();
{
    "_id" : 1,
    "StudName" : "Megha",
    "Grade" : "vii",
    "Hobbies" : "InternetSurfing"
}
> db.Student.c
ount(); 2
> db.Student.find().sort({StudName:-1}).pretty();
{
    "_id" : 1,
    "StudName" : "Megha",
    "Grade" : "vii",
    "Hobbies" : "InternetSurfing"
}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan",
"Hobbies" : "skating" } >
db.Student.save({StudName:"Vamsi",Greade
:"vi"}) WriteResult({ "nInserted" : 1 })
>
db.Students.update({_id:4},{ $set:{Location:"N
etwork"}}) WriteResult({ "nMatched" : 0,

```

```

    "nUpserted" : 0, "nModified" : 0 })
>
db.Students.update({_id:4},{Sunset:{Location:
"Network" }}) WriteResult({ "nMatched" : 0,
    "nUpserted" : 0, "nModified" : 0 })
>
db.Student.find({_id:1},{StudName:1,
Grade:1,_id:0}); { "StudName" :
    "Megha", "Grade" : "vii" }
> db.Student.find({Grade:{$ne:'VII'}}).pretty();
{
    "_id" : 1,
    "StudName" : "Megha",
    "Grade" : "vii",
    "Hobbies" : "InternetSurfing"
}
{ "_id" : 3, "Grade" : "vii", "StudName" : "Ayan",
    "Hobbies" : "skating" } {
    "_id" : ObjectId("6253f413e88b8c9e787b194e"),
    "StudName" : "Vamsi",
    "Grade" : "vi"
}
> db.Student.find({StudName:/s$/}).pretty();
>
db.Students.update({_id:3},{Sset:{Location:null
    }}) WriteResult({ "nMatched" : 0, "nUpserted" :

```



```

0, "nModified" : 0 }) > db.Students.c
ount() 0
> db.Students.count({Grade:
"VII"}) 0
> db.Student.find({Grade:"VII").limit(3).pretty();
> db.Student.update({_id:3},{ $set:{Location:null}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.count({Grade:"VII"})
0
> db.Students.count({Grade:
"vii"}) 0
> db.Student.c
ount() 3
> db.Student.count({Grade:
"vii"}) 2
>
db.Student.find({Grade:"vii").
limit(3).pretty(); {
    "_id" : 1,
    "StudName" : "Megha",
    "Grade" : "vii",
    "Hobbies" : "InternetSurfing"
}
{
    "_id" : 3,
    "Grade" : "vii",
    "StudName" : "Ayan",

```

```

        "Hobbies" : "skating",
        "Location" : null
    }
}
>
db.Student.find().sort({StudName:1}).pretty(); {
  "_id" : 3,
  "Grade" :
  "vii",
  "StudName" : "Ayan",
  "Hobbies" : "skating",
  "Location" : null
}
{
  "_id" : 1,
  "StudName" : "Megha",
  "Grade" : "vii",
  "Hobbies" : "InternetSurfing"
}
{
  "_id" : ObjectId("6253f413e88b8c9e787b194e"),
  "StudName" : "Vamsi",
  "Grade" : "vi"
}
> db.Student.find().skip(2).pretty()
{
  "_id" : ObjectId("6253f413e88b8c9e787b194e"),

```

```

    "StudName" : "Vamsi",
    "Grade" : "vi"
}
> db.food.insert( { _id:1, fruits:['grapes','mango','apple']; } )
2022-04-11T15:05:51.894+0530 E QUERY [thread1] SyntaxError: missing ] after element
list @(shell):1:57 > db.food.insert({_id:1,fruits:['grapes','mango','ap
ple']}) WriteResult({ "nInserted" : 1 })
> db.food.insert({_id:2,fruits:['grapes','mango','che
rry']}) WriteResult({ "nInserted" : 1 })
> db.food.insert({_id:3,fruits:['banana','ma
ngo']}) WriteResult({ "nInserted" : 1 })
> db.food.find({fruits:['grapes','mango','apple']}).pretty();
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
> db.food.find({'fruits.1':'grapes'})
> db.food.find({"fruits":{$size:2}})
{ "_id" : 3, "fruits" : [ "banana", "mango" ] }
> db.food.find({_id:1},{"fruits":{$slice:2}})
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }
> db.food.find({fruits:{$all:["mango","grapes"]}})
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }
> db.food.update({_id:3},{ $set:{"fruits.1":"apple"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.food.update({_id:2},{ $push:{price:{grapes:80,mango:200,cherry:1
00}}}) WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>db.Customers.insert({_custID:1,AcctBal:'100000',AcctType:"saving"});

```

```
WriteResult({ "nInserted" : 1 })
```

```
> db.Customers.aggregate({$group:{_id:"$custID",TotAccBal:{$sum:"$AccBal"}}});
```

```
{ "_id" : null, "TotAccBal" : 0 }
```

```
db.Customers.aggregate({$match:{AcctType:"saving"}},{ $group:{_id:"$custID",TotAccBal:{$sum:"$AccBal"}}}); { "_id" : null, "TotAccBal" : 0 }
```

```
db.Customers.aggregate({$match:{AcctType:"saving"}},{ $group:{_id:"$custID",TotAccBal:{$sum:"$AccBal"}},{ $ match:{TotAccBal:{$gt:1200}}
```

#### 4. Screenshot of Hadoop installed

```
Microsoft Windows [Version 10.0.22000.739]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
f
C:\WINDOWS\system32>jps
7072 DataNode
13492 Jps
15844 ResourceManager
16196 NameNode
1388 NodeManager

C:\WINDOWS\system32>hdfs dfs -ls -R /
drwxr-xr-x - khush supergroup 0 2022-06-27 14:09 /input
drwxr-xr-x - khush supergroup 0 2022-06-21 09:03 /input/inputtest
-rw-r--r-- 1 khush supergroup 21 2022-06-21 09:03 /input/inputtest/output.txt
-rw-r--r-- 1 khush supergroup 21 2022-06-21 08:19 /input/sample.txt
-rw-r--r-- 1 khush supergroup 21 2022-06-27 14:09 /input/sample2.txt
drwxr-xr-x - khush supergroup 0 2022-06-21 13:30 /test
-rw-r--r-- 1 khush supergroup 19 2022-06-21 13:30 /test/sample.txt

C:\WINDOWS\system32>hadoop version
Hadoop 3.3.3
Source code repository https://github.com/apache/hadoop.git -r d37506cbda38c338d9fe481addda5a05fb516f71
Compiled by stevel on 2022-05-09T16:36Z
Compiled with protoc 3.7.1
From source with checksum eb96dd4a797b6989ae0cdb9db6efc6
This command was run using /C:/hadoop-3.3.3/share/hadoop/common/hadoop-common-3.3.3.jar

C:\WINDOWS\system32>
```

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

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -cat /mydir/file1.txt
21/04/19 23:38:07 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
I am using Hadoop
line1
line2
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -copyFromLocal ~/file1.txt /mydir
21/04/19 23:19:36 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /mydir
21/04/19 23:20:13 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r-- 1 hduser supergroup 30 2021-04-19 23:19 /mydir/file1.txt
hduser@lab-VirtualBox:/usr/local/sbin$
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -copyToLocal /mydir ~/hadoopcopy
21/04/19 23:29:39 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /
21/04/19 23:48:41 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
drwxr-xr-x - hduser supergroup 0 2021-04-19 23:45 /mydir
drwxr-xr-x - hduser supergroup 0 2021-04-19 23:41 /newdir
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -cp /mydir/sample.txt /newdir
21/04/19 23:48:56 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /newdir
21/04/19 23:49:22 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
drwxr-xr-x - hduser supergroup 0 2021-04-19 23:21 /newdir/mydir
-rw-r--r-- 1 hduser supergroup 13 2021-04-19 23:48 /newdir/sample.txt
hduser@lab-VirtualBox:/usr/local/sbin$
```



```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -mkdir /mydir
21/04/19 22:58:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /
21/04/19 23:41:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
```

```
drwxr-xr-x - hduser supergroup          0 2021-04-19 23:19 /mydir
drwxr-xr-x - hduser supergroup          0 2021-04-19 23:21 /mydr
drwxr-xr-x - hduser supergroup          0 2021-04-19 23:39 /newdir
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -mv /mydr /newdir
21/04/19 23:41:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /
21/04/19 23:41:44 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
```

```
drwxr-xr-x - hduser supergroup          0 2021-04-19 23:19 /mydir
drwxr-xr-x - hduser supergroup          0 2021-04-19 23:41 /newdir
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /newdir
21/04/19 23:42:05 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
```

```
drwxr-xr-x - hduser supergroup          0 2021-04-19 23:21 /newdir/mydr
hduser@lab-VirtualBox:/usr/local/sbin$
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -put ~/file1.txt /mydr
21/04/19 23:21:41 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /mydr
hadoop: command not found
```

```
hduser@lab-VirtualBox:/usr/local/sbin$ hadoop fs -ls /mydr
21/04/19 23:22:20 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
```

```
-rw-r--r--  1 hduser supergroup          30 2021-04-19 23:21 /mydr/file1.txt
```

## 6. Create a Map Reduce program to

a) find average temperature for each year from the NCDC data set. b) find the mean max temperature for every month

### AverageDriver

```
package temp;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import
org.apache.hadoop.mapreduce.lib.output.FileOutput
utFormat; public class AverageDriver {

    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Please Enter the input and output parameters");
            System.exit(-1);
        }

        Job job = new Job();
        job.setJarByClass(AverageDriver.class);
        job.setJobName("Max temperature");
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
```

```

job.setMapperClass(AverageMapper.class);

job.setReducerClass(AverageReducer.class);
job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

### **AverageMapper**

```

package temp;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

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

public void map(LongWritable key, Text value,
Mapper<LongWritable, Text, Text, IntWritable>.Context
context) throws IOException, InterruptedException { int
temperature;

String line = value.toString();

String year = line.substring(15, 19);

if (line.charAt(87) == '+') {

temperature = Integer.parseInt(line.substring(88, 92));

```



```

    } else {
        temperature = Integer.parseInt(line.substring(87, 92));
    }

    String quality = line.substring(92, 93);
    if (temperature != 9999 && quality.matches("[01459]"))
        context.write(new Text(year), new IntWritable(temperature));
    }
}

```

### **AverageReducer**

```

package temp;

import java.io.IOException;

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

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

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

        int max_temp = 0;

        int count = 0;

        for (IntWritable value : values) {
            max_temp += value.get();

            count++;
        }

        context.write(key, new IntWritable(max_temp / count));
    }
}

```

```
}
```

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

#### **b) MeanMaxDriver.class**

```
package meanmax;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import
org.apache.hadoop.mapreduce.lib.input.FileInputF
ormat; import
org.apache.hadoop.mapreduce.lib.output.FileOutp
utFormat; public class MeanMaxDriver {
    public static void main(String[] args)
        throws Exception { if (args.length !=
        2) {
            System.err.println("Please Enter the input and
            output parameters"); System.exit(-1);
        }
        Job job = new Job();
        job.setJarByClass(MeanMaxDriver.class);
        job.setJobName("Max temperature");
        FileInputFormat.addInputPath(job, new
```

```

Path(args[0]));
FileOutputFormat.setOutputPath(job,
new Path(args[1]));
job.setMapperClass(MeanMaxMapper.class);
job.setReducerClass(MeanMaxReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
MeanMaxMapper.class

```

```

package meanmax;

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

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

public void map(LongWritable key, Text value,
Mapper<LongWritable, Text, Text, IntWritable>.Context
context) throws IOException, InterruptedException { int
temperature;

String line = value.toString();

```

```

String month = line.substring(19, 21);
if (line.charAt(87) == '+') {
temperature = Integer.parseInt(line.substring(88, 92));
} else {
temperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (temperature != 9999 && quality.matches("[01459]"))
context.write(new Text(month), new IntWritable(temperature));
}
}

```

### **MeanMaxReducer.class**

```

package meanmax;

import java.io.IOException;

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

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

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

        int max_temp = 0;

        int total_temp = 0;

        int count = 0;

        int days = 0;

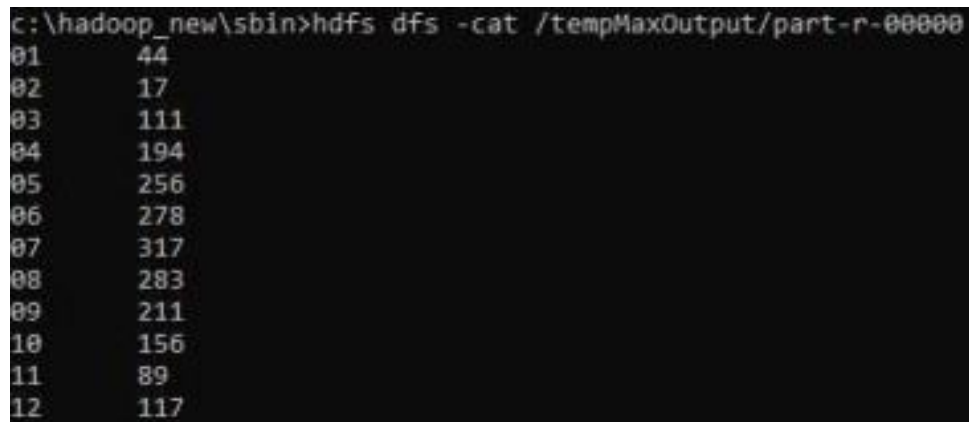
        for (IntWritable value : values) {

            int temp = value.get();

```

```
if (temp > max_temp)
    max_temp = temp;
    count++;
    if (count == 3) {
        total_temp += max_temp;
        max_temp = 0;
        count = 0;
        days++;
    }
}

context.write(key, new IntWritable(total_temp / days));
}
}
```



A terminal window showing the execution of an HDFS command. The command is `c:\hadoop_new\sbin>hdfs dfs -cat /tempMaxOutput/part-r-00000`. The output consists of 12 lines, each with a two-digit number followed by a space and a three-digit number. The numbers are: 01 44, 02 17, 03 111, 04 194, 05 256, 06 278, 07 317, 08 283, 09 211, 10 156, 11 89, and 12 117.

Index	Value
01	44
02	17
03	111
04	194
05	256
06	278
07	317
08	283
09	211
10	156
11	89
12	117

**7. For a given Text file, Create a Map Reduce program to sort the content in an alphabetic order listing only top 10 maximum occurrences of words.**

```
//Driver Code

package wordCount;

import java.io.IOException;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.FileInputFormat;

import org.apache.hadoop.mapred.FileOutputFormat;


import org.apache.hadoop.mapred.JobClient;

import org.apache.hadoop.mapred.JobConf;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;


public class WCDriver extends Configured implements Tool {

    public int run(String args[]) throws IOException
    {

        if (args.length < 2)
        {

            System.out.println("Please give valid inputs");

            return -1;

        }

    }

}
```

```

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

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

```

```

//Mapper Code
package wordCount;
import java.io.IOException;

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

```

```

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.MapReduceBase;

import org.apache.hadoop.mapred.Mapper;

import org.apache.hadoop.mapred.OutputCollector;


import org.apache.hadoop.mapred.Reporter;

public class WCMapper extends MapReduceBase implements Mapper<LongWritable,Text,
    Text, IntWritable> { // Map function

    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable>
output, Reporter rep) throws IOException
    {

        String line = value.toString();

        // Splitting the line on spaces
        for (String word : line.split(" "))
        {

            if (word.length() > 0)
            {

                output.collect(new Text(word), new IntWritable(1));

            }

        }

    }

}

```

```

//Reducer Code
package wordCount;

```

```

import java.io.IOException;

```



```
import java.util.Iterator;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.MapReduceBase;

import org.apache.hadoop.mapred.OutputCollector;

import org.apache.hadoop.mapred.Reducer;

import org.apache.hadoop.mapred.Reporter;


public class WCReducer extends MapReduceBase implements Reducer<Text,IntWritable,
    Text, IntWritable> { // Reduce function

    public void reduce(Text key, Iterator<IntWritable> value,
OutputCollector<Text, IntWritable> output,Reporter rep) throws IOException
    {

        int count = 0;

        // Counting the frequency of each words
        while (value.hasNext())
        {

            IntWritable i = value.next();

            count += i.get();

        }

        output.collect(key, new IntWritable(count));

    }

}
```

```
Activities Terminal Apr 26 15:26
hduser@lab-VirtualBox: /home/lab/hadoop-2.6.0/share/ha...

Bytes Read=44
File Output Format Counters
Bytes Written=35
hduser@lab-VirtualBox:/home/lab/hadoop-2.6.0/share/hadoop/mapreduce$ hadoop dfs
-ls /firstExampleOut
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

21/04/26 15:20:15 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r-- 1 hduser supergroup 0 2021-04-26 15:19 /firstExampleOut/_
SUCCESS
-rw-r--r-- 1 hduser supergroup 35 2021-04-26 15:19 /firstExampleOut/p
art-r-00000
hduser@lab-VirtualBox:/home/lab/hadoop-2.6.0/share/hadoop/mapreduce$ hadoop dfs
-cat /firstExampleOut/part-r-00000
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

21/04/26 15:22:01 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
bear 2
car 3
deer 1
deer 1
river 2
hduser@lab-VirtualBox:/home/lab/hadoop-2.6.0/share/hadoop/mapreduce$ hadoop dfs
-cat /firstExampleOut/part-r-00000
```

## 8. Create a Map Reduce program to demonstrating join operation

```
// JoinDriver.java

import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.mapred.lib.MultipleInputs;
import org.apache.hadoop.util.*;

public class JoinDriver extends Configured implements Tool {

    public static class KeyPartitioner implements Partitioner<TextPair, Text> {

        @Override

        public void configure(JobConf job) {}

        @Override

        public int getPartition(TextPair key, Text value, int numPartitions) {

            return (key.getFirst().hashCode() & Integer.MAX_VALUE) %

            numPartitions;

        }

    }

    @Override

    public int run(String[] args) throws Exception {

        if (args.length != 3) {

            System.out.println("Usage: <Department Emp Strength input>

            <Department Name input>

            <output>"); return -1;

        }

    }

}
```

```

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

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

Path AInputPath = new Path(args[0]);

Path BInputPath = new Path(args[1]);

Path outputPath = new Path(args[2]);

MultipleInputs.addInputPath(conf, AInputPath,
    TextInputFormat.class, Posts.class);

MultipleInputs.addInputPath(conf, BInputPath,
    TextInputFormat.class, User.class);

FileOutputFormat.setOutputPath(conf, outputPath);

conf.setPartitionerClass(KeyPartitioner.class);

conf.setOutputValueGroupingComparator(TextPair.FirstComparator.class);

conf.setMapOutputKeyClass(TextPair.class);

conf.setReducerClass(JoinReducer.class); conf.setOutputKeyClass(Text.class);

JobClient.runJob(conf);

return 0;
}

public static void main(String[] args) throws Exception {
    int exitCode = ToolRunner.run(new JoinDriver(), args);
    System.exit(exitCode);
}
}

```

// JoinReducer.java

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

```

import java.util.Iterator;
import org.apache.hadoop.io.Text;

import org.apache.hadoop.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);
}
}}

```

```

hduser@bmsce-Precision-T1700:/home/bmsce$ hdfs dfs -cat /join/output/*
A11      Finance      50
B12      HR              100
C13      Manufacturing    250
Dept_ID  Dept_Name        Total_Employee

```

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

```
val data=sc.textFile("sparkdata.txt")

data.collect;

val splitdata = data.flatMap(line => line.split(" "));

splitdata.collect;

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

mapdata.collect;

val reducedata = mapdata.reduceByKey(_+_);

reducedata.collect;
```



```
binase@binase-Vision-F178C ~/Desktop
spark sessions available as 'spark',
welcome to

Spark version 2.4.0

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_212)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val data=sc.textFile("sample.txt")
data: org.apache.spark.rdd.RDD[String] = sample.txt MapPartitionsRDD[1] at textFile at <console>:14

scala> data.collect;
res0: Array[String] = Array(hi how are you, how is your job, how is your family, how is your brother, how is your sister)

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(hi, how, are, you, how, is, your, job, how, is, your, family, how, is, your, brother, how, is, your, sister)

scala> val mapdata = splitdata.map(word => (word,1));
mapdata: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at map at <console>:28

scala> mapdata.collect;
res2: Array[(String, Int)] = Array((hi,1), (how,1), (are,1), (you,1), (how,1), (is,1), (your,1), (job,1), (how,1), (is,1), (your,1), (family,1), (how,1), (is,1), (your,1), (brother,1), (how,1), (is,1), (your,1), (sister,1))

scala> val reducedata = mapdata.reduceByKey(_+_);
reducedata: org.apache.spark.rdd.RDD[(String, Int)] = ShuffleRDD[4] at reduceByKey at <console>:29

scala> reducedata.collect;
res3: Array[(String, Int)] = Array((are,1), (brother,1), (is,4), (sister,1), (family,1), (how,5), (job,1), (you,1), (hi,1), (your,4))

scala>
```

## 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

```
val textFile = sc.textFile("/home/bhoom/Desktop/wc.txt")
val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)
import scala.collection.immutable.ListMap
val sorted=ListMap(counts.collect.sortWith(_._2 > _._2):_*)// sort in descending order based
on values
println(sorted)
for((k,v)<-sorted)
{
  if(v>4)
  {
    print(k+",")
    print(v)
    println()
  }
}
```

```
scala> val textFile = sc.textFile("sample.txt")
textFile: org.apache.spark.rdd.RDD[String] = sample.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] = Map(how -> 5, is -> 4, your -> 4, are -> 1, brother -> 1, sister -> 1, family -> 1, job -> 1, you -> 1, hl -> 1)

scala> println(sorted)
Map(how -> 5, is -> 4, your -> 4, are -> 1, brother -> 1, sister -> 1, family -> 1, job -> 1, you -> 1, hl -> 1)

scala> for((k,v)<-sorted)
| {
| |
Display all 689 possibilities? (y or n)
| |
| | {
| | | {
| | | | print(k+",")
| | | | print(v)
| | | | println()
| | | }
| | }
| }
```