

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



**LAB REPORT**

**on**

**COURSE TITLE**

*Submitted by*

**DEREK STANLEY KANNATHE (1BM19CS045)**

*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-2022 to July-2022**

**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 **DEREK STANLEY KANNATHE (1BM19CS045)**, 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.

**Dr. Pallavi G.B**  
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	Page No.
<b>1</b>	DB operations using Cassandra - Employee	<b>4</b>
<b>2</b>	DB operations using Cassandra – Library	<b>6</b>
<b>3</b>	MongoDB- CRUD Demonstration	<b>9</b>
<b>4</b>	Screenshot of Hadoop installed	<b>13</b>
<b>5</b>	Execution of HDFS Commands for interaction with Hadoop Environment.	<b>14</b>
<b>6</b>	Create a Map Reduce program for weather data: a) find average temperature for each year from NCDC data set. b) find the mean max temperature for every month	<b>16</b>
<b>7</b>	Create a Map Reduce program to sort the content in an alphabetic order listing only top 10 maximum occurrences of words.	<b>19</b>
<b>8</b>	Create a Map Reduce program to demonstrating join operation	<b>20</b>
<b>9</b>	Program to print word count on Scala shell and print “Hello world” on Scala IDE	<b>21</b>
<b>10</b>	Using RDD and Flat Map 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	<b>22</b>

## 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. DB operations using Cassandra – Employee:

## LAB 1 - Cassandra Commands

```sql

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

```
cqlsh> describe keyspaces;
```

```
students system_auth system_schema system_views
```

```
system system_distributed system_traces system_virtual_schema
```

```
cqlsh> use students;
```

```
cqlsh:students> create table student_info( rollNo int primary key, name text, joinDate timestamp, lastExamPerc double );
```

```
cqlsh:students> describe tables
```

```
student_info
```

```
cqlsh:students> describe table student
```

```
student_info students.
```

```
cqlsh:students> describe table student_info
```

```
CREATE TABLE students.student_info (
```

```
    rollno int PRIMARY KEY,
```

```
    joindate timestamp,
```

```
    lastexamperc double,
```

```
    name text
```

```
) WITH additional_write_policy = '99p'
```

```
    AND bloom_filter_fp_chance = 0.01
```

```
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
```

```
    AND cdc = false
```

```
    AND comment = ''
```

```
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
```

```
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
```

```
    AND crc_check_chance = 1.0
```

```
    AND default_time_to_live = 0
```

```
    AND extensions = {}
```

```
    AND gc_grace_seconds = 864000
```

```
AND gc_grace_seconds = 864000
```

```
AND max_index_interval = 2048
```

```
AND memtable_flush_period_in_ms = 0
```

```
AND min_index_interval = 128
```

```
AND read_repair = 'BLOCKING'
```

```
AND speculative_retry = '99p';
```

```
cqlsh:students> begin batch insert into student_info(rollno, joindate, lastexamperc, name) values (1, '2021-05-23', 90.0, 'Adam') insert into student_info(rollno, joindate, lastexamperc, name) values (2, '2021-05-22', 97.7, 'Eve') apply batch;
```

```
cqlsh:students> select * from student_info;
```

| rollno | joindate                        | lastexamperc | name |
|--------|---------------------------------|--------------|------|
| 1      | 2021-05-22 18:30:00.000000+0000 | 90           | Adam |
| 2      | 2021-05-21 18:30:00.000000+0000 | 97.7         | Eve  |

(2 rows)

```
cqlsh:students> update student_info set name = 'Micheal' where rollno = 1;
```

```
cqlsh:students> select * from student_info where rollno in (1,2);
```

| rollno | joindate                        | lastexamperc | name    |
|--------|---------------------------------|--------------|---------|
| 1      | 2021-05-22 18:30:00.000000+0000 | 90           | Micheal |
| 2      | 2021-05-21 18:30:00.000000+0000 | 97.7         | Eve     |

(2 rows)

```
cqlsh:students> create index on student_info(lastexamperc);
```

```
cqlsh:students> select rollno, name from student_info limit 2;
```

| rollno | name    |
|--------|---------|
| 1      | Micheal |
| 2      | Eve     |

(2 rows)

```
cqlsh:students> create index on student_info(name);
```

```
cqlsh:students> update student_info set name='Eve2', lastexamperc=100.0 where rollno=2;
```

```
cqlsh:students> select * from student_info;
```

(2 rows)

```
cqlsh:students> create index on student_info(lastexamperc);
```

```
cqlsh:students> select rollno, name from student_info limit 2;
```

| rollno | name |
|--------|------|
|--------|------|

|   |         |
|---|---------|
| 1 | Micheal |
|---|---------|

|   |     |
|---|-----|
| 2 | Eve |
|---|-----|

(2 rows)

```
cqlsh:students> create index on student_info(name);
```

```
cqlsh:students> update student_info set name='Eve2', lastexamperc=100.0 where rollno=2;
```

```
cqlsh:students> select * from student_info;
```

| rollno | joindate | lastexamperc | name |
|--------|----------|--------------|------|
|--------|----------|--------------|------|

|   |                                 |    |         |
|---|---------------------------------|----|---------|
| 1 | 2021-05-22 18:30:00.000000+0000 | 90 | Micheal |
|---|---------------------------------|----|---------|

|   |                                 |     |      |
|---|---------------------------------|-----|------|
| 2 | 2021-05-21 18:30:00.000000+0000 | 100 | Eve2 |
|---|---------------------------------|-----|------|

|   |                                 |     |      |
|---|---------------------------------|-----|------|
| 2 | 2021-05-21 18:30:00.000000+0000 | 100 | Eve2 |
|---|---------------------------------|-----|------|

(2 rows)

```
cqlsh:students> delete lastexamperc from student_info where rollno=2;
```

```
cqlsh:students> select * from student_info;
```

| rollno | joindate | lastexamperc | name |
|--------|----------|--------------|------|
|--------|----------|--------------|------|

|   |                                 |    |         |
|---|---------------------------------|----|---------|
| 1 | 2021-05-22 18:30:00.000000+0000 | 90 | Micheal |
|---|---------------------------------|----|---------|

|   |                                 |    |         |
|---|---------------------------------|----|---------|
| 2 | 2021-05-21 18:30:00.000000+0000 | 90 | Micheal |
|---|---------------------------------|----|---------|

|   |                                 |      |      |
|---|---------------------------------|------|------|
| 2 | 2021-05-21 18:30:00.000000+0000 | null | Eve2 |
|---|---------------------------------|------|------|

(2 rows)

```
cqlsh:students> delete from student_info where rollno=2;
```

```
cqlsh:students> select * from student_info;
```

| rollno | joindate | lastexamperc | name |
|--------|----------|--------------|------|
|--------|----------|--------------|------|

|   |                                 |    |         |
|---|---------------------------------|----|---------|
| 1 | 2021-05-22 18:30:00.000000+0000 | 90 | Micheal |
|---|---------------------------------|----|---------|

|   |                                 |    |         |
|---|---------------------------------|----|---------|
| 1 | 2021-05-22 18:30:00.000000+0000 | 90 | Micheal |
|---|---------------------------------|----|---------|

(1 rows)

...

## 2. DB operations using Cassandra – Library:

```
```sql
cqlsh> create keyspace employee_info with
replication={'class':'SimpleStrategy','replication_factor':1};
cqlsh> use employee_info;
cqlsh:employee_info> create table employee_details(emp_id int, emp_name text, designation text,
doj timestamp, salary double, dept_name text, primary key(emp_id,salary));
cqlsh:employee_info> describe table employee_details;
CREATE TABLE employee_info.employee_details (
    emp_id int,
    salary double,
    dept_name text,
    designation text,
    doj timestamp,
    emp_name text,
    PRIMARY KEY (emp_id, salary)
) WITH CLUSTERING ORDER BY (salary ASC)
    AND additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND cdc = false
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy',
'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '16', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND extensions = {}
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';
cqlsh:employee_info> begin batch insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values
(100,'tanya','manager','2020-09-11',30000,'testing') insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values
(111,'sriram','associate','2020-06-11',25000,'development') insert into
employee_details(emp_id,emp_name,designation,doj,salary,dept_name) values
(121,'shiva','manager','2020-01-03',35000,'hr') apply batch;
cqlsh:employee_info> select * from employee_details;
emp_id | salary | dept_name | designation | doj | emp_name
-----+-----+-----+-----+-----+-----
111 | 25000 | development | associate | 2020-06-10 18:30:00.000000+0000 | sriram
121 | 35000 | hr | manager | 2020-01-02 18:30:00.000000+0000 | shiva
100 | 30000 | testing | manager | 2020-09-10 18:30:00.000000+0000 | tanya
```

(3 rows)

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

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

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

(3 rows)

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

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

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

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

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

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

(3 rows)

```
cqlsh:employee_info> insert into
```

```
employee_details(emp_id,emp_name,designation,doj,salary,dept_name)
```

```
values(113,'sam','manager','2020-09-09',30000,'testing') using ttl 30;
```

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

```
ttl(emp_name)
```

```
-----
```

22

(1 rows)

```
cqlsh:employee_info> paging off;
```

Disabled Query paging.

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

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



### 3. MongoDB- CRUD Demonstration:

## BDA LAB-3

### MongoDB

1. Create a new collection

```
```
```

```
use Student
```

```
```
```

2. Insert a value

```
```json
```

```
db.Student.insert({
```

```
  "Name" : "XYZ",
```

```
  "RollNo" : 1,
```

```
  "Age" : 21,
```

```
  "ContactNo" : "1234567890",
```

```
  "EmailId" : "user1@lab.com"
```

```
})
```

```
```
```

3. Insert multiple values at once

```
```json
```

```
var MyStudents = [
```

```
{
```

```
  "Name" : "ABC",
```

```
  "RollNo" : 3,
```

```
  "Age" : 22,
```

```
  "ContactNo" : "2234567890",
```

```
  "EmailId" : "user2@lab.com"
```

```
},
```

```
{
```

```
  "Name" : "DEF",
```

```
  "RollNo" : 5,
```

```
  "Age" : 21,
```

```
  "ContactNo" : "3234567890",
```

```
  "EmailId" : "user3@lab.com"
```

```
},
```

```
{
```

```
  "Name" : "GHI",
```

```
  "RollNo" : 7,
```

```
  "Age" : 20,
```

```
  "ContactNo" : "4234567890",
```

```
  "EmailId" : "user4@lab.com"
```

```
},
```

```
{
```

```
  "Name" : "JKL",
```

```
  "RollNo" : 10,
```

```
  "Age" : 18,
```

```
        "ContactNo" : "5234567890",
        "EmailId" : "user5@lab.com"
    },
]
```

```
db.Student.insert(MyStudents);
...
```

4. Print all current values

```
```json
db.getCollection('Student').find({}).forEach(printjson)
...
```json
{
    "_id" : ObjectId("606ad5a6e581cc0b904470a5"),
    "Name" : "XYZ",
    "RollNo" : 1,
    "Age" : 21,
    "ContactNo" : "1234567890",
    "EmailId" : "user1@lab.com"
}
{
    "_id" : ObjectId("606ad60fe581cc0b904470a6"),
    "Name" : "ABC",
    "RollNo" : 3,
    "Age" : 22,
    "ContactNo" : "2234567890",
    "EmailId" : "user2@lab.com"
}
{
    "_id" : ObjectId("606ad60fe581cc0b904470a7"),
    "Name" : "DEF",
    "RollNo" : 5,
    "Age" : 21,
    "ContactNo" : "3234567890",
    "EmailId" : "user3@lab.com"
}
{
    "_id" : ObjectId("606ad60fe581cc0b904470a8"),
    "Name" : "GHI",
    "RollNo" : 7,
    "Age" : 20,
    "ContactNo" : "4234567890",
    "EmailId" : "user4@lab.com"
}
{
    "_id" : ObjectId("606ad60fe581cc0b904470a9"),
    "Name" : "JKL",
```

```
    "RollNo:" : 10,  
    "Age" : 18,  
    "ContactNo" : "5234567890",  
    "EmailId" : "user5@lab.com"  
  }  
  ...
```

#### 5. Update RollNo of a student

```
```json  
db.Student.update(  
  {"RollNo:" : 10},  
  {$set: { "EmailId" : "modified@lab.com"}});  
...  
  
```json  
db.getCollection('Student').find({"RollNo":10}).forEach(printjson)  
...  
  
```json  
{  
  "_id" : ObjectId("606ad60fe581cc0b904470a9"),  
  "Name" : "JKL",  
  "RollNo:" : 10,  
  "Age" : 18,  
  "ContactNo" : "5234567890",  
  "EmailId" : "modified@lab.com"  
}  
...
```

#### 6. Update Name of a student

```
```json  
db.Student.update(  
  {"Name" : "XYZ"},  
  {$set: { "Name" : "EcksWhyZee"}});  
...  
  
```json  
db.getCollection('Student').find({"Name" : "EcksWhyZee"}).forEach(printjson)  
...  
  
```json  
{  
  "_id" : ObjectId("606ad5a6e581cc0b904470a5"),  
  "Name" : "EcksWhyZee",  
  "RollNo:" : 1,  
  "Age" : 21,  
  "ContactNo" : "1234567890",  
  "EmailId" : "user1@lab.com"  
}  
...
```

#### 7. Export to json

```
...  
mongoexport --db testdb --collection Student --out C:\Users\shaan\Desktop\Exported\Student.json  
...
```

```
```json
```

```
{ "_id": {"$oid": "606ad5a6e581cc0b904470a5"}, "Name": "EcksWhyZee", "RollNo": "1.0", "Age": 21.0, "ContactNo": "1234567890", "EmailId": "user1@lab.com" }
```

```
{ "_id": {"$oid": "606ad60fe581cc0b904470a6"}, "Name": "ABC", "RollNo": "3.0", "Age": 22.0, "ContactNo": "2234567890", "EmailId": "user2@lab.com" }
```

```
{ "_id": {"$oid": "606ad60fe581cc0b904470a7"}, "Name": "DEF", "RollNo": "5.0", "Age": 21.0, "ContactNo": "3234567890", "EmailId": "user3@lab.com" }
```

```
{ "_id": {"$oid": "606ad60fe581cc0b904470a8"}, "Name": "GHI", "RollNo": "7.0", "Age": 20.0, "ContactNo": "4234567890", "EmailId": "user4@lab.com" }
```

```
{ "_id": {"$oid": "606ad60fe581cc0b904470a9"}, "Name": "JKL", "RollNo": "10.0", "Age": 18.0, "ContactNo": "5234567890", "EmailId": "modified@lab.com" }
```

```
```
```

#### 8. Drop Student

```
```json
```

```
db.getCollection('Student').drop()
```

```
```
```

#### 9. Import from exported file

```
```
```

```
mongoimport --db testdb --collection Student C:\Users\shaan\Desktop\Exported\Student.json
```

```
```
```

```
---
```

#### 4. Screenshot of Hadoop installed:

```
C:\Users\derek>hadoop version
Hadoop 3.3.0
Source code repository https://gitbox.apache.org/repos/asf/hadoop.git -r aa96f1871bfd858f9bac59cf2a81ec470da649af
Compiled by brahma on 2020-07-06T18:44Z
Compiled with protoc 3.7.1
From source with checksum 5dc29b802d6ccd77b262ef9d04d19c4
This command was run using /C:/hadoop-3.3.0/share/hadoop/common/hadoop-common-3.3.0.jar
```

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

### Hadoop Commands

To start with:

```
hduser@bmsce-Precision-T1700:~$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [localhost]
hduser@localhost's password:
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hduser-namenode-bmsce-
Precision-T1700.out
hduser@localhost's password:
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hduser-datanode-bmsce-
Precision-T1700.out
Starting secondary namenodes [0.0.0.0]
hduser@0.0.0.0's password:
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-hduser-
secondarynamenode-bmsce-Precision-T1700.out
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-hduser-resourcemanager-bmsce-
Precision-T1700.out
hduser@localhost's password:
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hduser-nodemanager-
bmsce-Precision-T1700.out
```

```
hduser@bmsce-Precision-T1700:~$ jps
7097 DataNode
7802 NodeManager
12540 Jps
7469 ResourceManager
6925 NameNode
7310 SecondaryNameNode
```

Commands:

```
1:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -mkdir /hadoop
2:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -ls /
Found 1 item
drwxr-xr-x - hduser supergroup    0 2022-06-06 11:37 /hadoop
3:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -put /home/hduser/Desktop/hadoop.txt
/hadoop/hadoop.txt
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /hadoop/hadoop.txt
Hello, I'm Hadoop
4:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -copyFromLocal /home/hduser/Desktop/hadoop.txt
/hadoop/hadoop2.txt
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /hadoop/hadoop.txt
Hello, I'm Hadoop
5:
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -get /hadoop/hadoop1.txt
/home/hduser/Desktop/hd.txt
hduser@bmsce-Precision-T1700:~$ ls Desktop/hd.txt
Desktop/hd.txt
hduser@bmsce-Precision-T1700:~$ hdfs dfs -getmerge /hadoop/hadoop.txt /hadoop/hadoop2.txt
/home/hduser/Desktop/hd_merge.txt
hduser@bmsce-Precision-T1700:~$ ls Desktop/hd_merge.txt
Desktop/hd_merge.txt
hduser@bmsce-Precision-T1700:~$ hdfs dfs -getfacl /hadoop
# file: /hadoop
# owner: hduser
# group: supergroup
user::rwx
group::r-x
other::r-x
6:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -copyToLocal /hadoop/hadoop.txt
/home/hduser/Desktop/hd2.txt
hduser@bmsce-Precision-T1700:~$ ls Desktop/hd2.txt
Desktop/hd2.txt
7:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /hadoop/hadoop.txt
Hello, I'm Hadoop
8:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -mkdir /hadoop/AA
hduser@bmsce-Precision-T1700:~$ hdfs dfs -mv /hadoop/hadoop.txt /hadoop/AA/hadoop.txt
hduser@bmsce-Precision-T1700:~$ hdfs dfs -ls /hadoop/AA
Found 1 items
-rw-r--r--  1 hduser supergroup    18 2022-06-06 11:41 /hadoop/AA/hadoop.txt
9:
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cp /hadoop/AA/hadoop.txt /hadoop/hadoop2.txt
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /hadoop/hadoop2.txt
Hello, I'm Hadoop
To stop Hadoop:
hduser@bmsce-Precision-T1700:~$ stop-all.sh
This script is Deprecated. Instead use stop-dfs.sh and stop-yarn.sh
Stopping namenodes on [localhost]
hduser@localhost's password:
localhost: stopping namenode
hduser@localhost's password:
localhost: stopping datanode
Stopping secondary namenodes [0.0.0.0]
hduser@0.0.0.0's password:
0.0.0.0: stopping secondarynamenode
stopping yarn daemons
stopping resourcemanager
hduser@localhost's password:
localhost: stopping nodemanager
no proxyserver to stop
```

## 6. Map Reduce program for weather data:

### Average

MAPPER

```
#!/usr/bin/python
```

```
import sys
```

```
for line in sys.stdin:
```

```
    line = line.strip()
```

```
    year = line[15:19]
```

```
    if line[87] == '+':
```

```
        temperature = int(line[88:92])
```

```
    else:
```

```
        temperature = int(line[87:92])
```

```
    quality = line[92:93]
```

```
    if temperature != 9999 and quality in "[01459]":
```

```
        print(year+"\t"+str(temperature))
```

REDUCER

```
#!/usr/bin/python
```

```
import sys
```

```
cur_year = None
```

```
average_temp = 0
```

```
count = 0
```

```
for line in sys.stdin:
```

```
    line = line.strip()
```

```
    year, temperature = line.split("\t",1)
```

```
    if cur_year == None:
```

```
        cur_year = year
```

```
    elif cur_year != year:
```

```
        print(cur_year+"\t"+str(average_temp // count))
```

```
        average_temp = 0
```

```
        count = 0
```

```
    average_temp += int(temperature)
```

```
    count += 1
```

```
if cur_year == year:
```

```
    print(cur_year+"\t"+str(average_temp // count))
```

#OUTPUT

1901 46



## Mean max

MAPPER

```
#!/usr/bin/python
import sys
```

```
for line in sys.stdin:
```

```
    line = line.strip()
    month = line[19:21]
    if line[87] == '+':
        temperature = int(line[88:92])
    else:
        temperature = int(line[87:92])
    quality = line[92]
    if temperature != 9999 and quality in "[01459]":
        print(month+"\t"+str(temperature))
```

REDUCER

```
#!/usr/bin/python
import sys
```

```
cur_month = None
```

```
max_temp = 0
```

```
temp_sum = 0
```

```
count = 0
```

```
days = 0
```

```
for line in sys.stdin:
```

```
    line = line.strip()
    month, temperature = line.split("\t", 1)
    if cur_month == None:
        cur_month = month
    elif cur_month != month:
        print(cur_month+"\t"+str(temp_sum//days))
        cur_month = month
        max_temp = 0
        temp_sum = 0
        count = 0
        days = 0
    if int(temperature) > max_temp:
        max_temp = int(temperature)
    count += 1
    if count == 3:
        temp_sum += max_temp
        max_temp = 0
        count = 0
        days += 1
if cur_month == month:
    print(cur_month+"\t"+str(temp_sum//days))
```

```
c:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \tempMaxOutput\part-r-00000
```

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

## 7. Map Reduce program - Top N:

MAPPER

```
#!/usr/bin/python
```

```
import sys
```

```
for line in sys.stdin:
```

```
    line = line.strip()
```

```
    words = line.split()
```

```
    for word in words:
```

```
        print(word+"\t"+str(1))
```

REDUCER

```
#!/usr/bin/python
```

```
import sys
```

```
current_word = None
```

```
current_count = 0
```

```
word = None
```

```
word_map = []
```

```
N = 20
```

```
for line in sys.stdin:
```

```
    line = line.strip()
```

```
    word, count = line.split("\t", 1)
```

```
    try:
```

```
        count = int(count)
```

```
    except ValueError:
```

```
        continue
```

```
    if current_word == word:
```

```
        current_count += 1
```

```
    else:
```

```
        if current_word:
```

```
            word_map.append([(current_count), current_word])
```

```
            current_count = count
```

```
            current_word = word
```

```
if current_word == word:
```

```
    word_map.append([(current_count), current_word])
```

```
word_map.sort(reverse=True)
```

```
for v, k in word_map:
```

```
    print("%s\t%d" % (k, v))
```

OUTPUT

```
hadoop@ubuntuVM:~/Downloads$ hadoop fs -cat /user/hadoop/output/part-r-00000
```

```
car 7
```

```
deer 6
```

```
bear 3
```

## 8. Map Reduce program to demonstrating join operation:

MAPPER

```
#!/usr/bin/python
```

```
import sys
```

```
for line in sys.stdin:
```

```
    dept_ID = "-1" # default sorted as first
```

```
    dept_Name = "-1" # default sorted as first
```

```
    no_Emp = "-1" # default sorted as first
```

```
    line = line.strip()
```

```
    splits = line.split("\t")
```

```
    if splits[-1].isdigit(): # dept strength data
```

```
        dept_ID = splits[0]
```

```
        no_Emp = str(splits[1])
```

```
    else:
```

```
        dept_ID = splits[0]
```

```
        dept_Name = str(splits[1])
```

```
    print('%s^%s^%s' % (dept_ID, dept_Name, no_Emp))
```

REDUCER

```
#!/usr/bin/python
```

```
import sys
```

```
new_list = {}
```

```
for line in sys.stdin:
```

```
    line = line.strip()
```

```
    dept_ID, dept_Name, no_Emp = line.split("^")
```

```
    if dept_ID not in new_list.keys():
```

```
        new_list[dept_ID] = [dept_Name, int(no_Emp)]
```

```
    else:
```

```
        if dept_Name != -1:
```

```
            new_list[dept_ID][0] = dept_Name
```

```
        if no_Emp != -1:
```

```
            if new_list[dept_ID][1] != -1:
```

```
                new_list[dept_ID][1] += int(no_Emp)
```

```
            else:
```

```
                new_list[dept_ID][1] = int(no_Emp)
```

```
for i in new_list:
```

```
    print(i+"\t"+new_list[i][0]+"\\t"+str(new_list[i][1]))
```

OUTPUT

```
hdfs dfs -cat /prog/part-00000
```

```
C13 Manufacturing 249
```

```
B12 HR 99
```

```
A11 FINANCE 49
```

## 9. Word count on Scala shell:

```
hadoop@ubuntuVM: ... x hadoop@ubuntuVM: ... x hadoop@ubuntuVM: ... x ▼  
  
scala> val txt = sc.textFile("./input.txt")  
txt: org.apache.spark.rdd.RDD[String] = ./input.txt MapPartitionsRDD[7] at textF  
ile at <console>:24  
  
scala> val counts = txt.flatMap(line => line.split(" ")).map(word => (word, 1)).  
reduceByKey(_ + _)  
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[10] at reduceByKey  
at <console>:25  
  
scala> counts.collect()  
res2: Array[(String, Int)] = Array((this,1), (wolf,1), (is,1), (spot.,1), (repea  
ted,1), (cappucino.,1), (anything,1), (with,1), (some,2), (as,1), (come,1), (dog  
,2), (cat,3), (Here,1), (up,1), (not,1), (text,1), (on,1), (could,1), (I,1), (aa  
re,1), (else,1), (random,1), (words,1), (the,1))  
  
scala> |
```

## 10. RDD and Flat Map count how many times each word appears strictly greater than 4 times:

```
val textFile = sc.textFile("D:\\sparkdata2.txt")

val counts = textFile.flatMap(line = line.split( )).map(word = (word, 1)).reduceByKey(_ + _)

import scala.collection.immutable.ListMap

val sortedWords=ListMap(counts.collect.sortWith(_. _2 >= _. _2)_)

println(sortedWords)

for((k,v)<-sortedWords)
{
  if(v>4)
  {
    print(k+",")
    print(v)
    println()
  }
}
```

Spark, 6