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

Autonomous Institute, Affiliated to VTU

Lab Record

**BIG DATA ANALYTICS**

*Submitted in partial fulfillment for the 6th Semester Laboratory*

Bachelor of Technology in

Computer Science and Engineering

*Submitted by:*

**Akshaya Deep Prasad**

1BM19CS013

Department of Computer Science and Engineering

B.M.S. College of Engineering

Bull Temple Road, Basavanagudi, Bangalore 560 019 Mar-July 2022

B.M.S. COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



***CERTIFICATE***

This is to certify that the Big Data Analytics (20CS6PEBDA) laboratory has been carried out by Akshaya Deep Prasad **(**1BM19CS013) during the 6th Semester Mar- July-2022.

Signature of the Faculty In-charge: **Pallavi GB**

Department of Computer Science and Engineering

B.M.S. College of Engineering, Bangalore

# TABLE OF CONTENTS

|  |  |
| --- | --- |
| **SL NO** | **TITLE** |
| 1 | EMPLOYEE DATABASE |
| 2 | LIBRARY DATABASE |
| 3 | MONGODB SAMPLE |
| 4 | HADOOP INSTALLATION |
| 5 | HADOOP SAMPLE |
| 6 | MAPREDUCE TEMPERATURE |
| 7 | MAPREDUCE TOPN |
| 8 | MAPREDUCE JOIN |
| 9 | SCALA INSTALLATION |
| 10 | SCALA WORDCOUNT |

# Employee database (CASSANDRA)

Question -

Perform the following DB operations using Cassandra.

1. Create a keyspace by name Employee
2. Create a column family by name Employee-Info with attributes Emp\_Id Primary Key, Emp\_Name, Designation, Date\_of\_Joining, Salary, Dept\_Name
3. Insert the values into the table in batch
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> create keyspace employee\_info with replication={'class':'SimpleStrategy','replication\_factor':1};

cqlsh> use employee\_info;

cqlsh:employee\_info> create table employee\_details(emp\_id int,emp\_name text,designation text,doj timestamp,salary double,dept\_name text,primary key(emp\_id,salary));

cqlsh:employee\_info> begin batch

... insert into employee\_details(emp\_id,emp\_name,designation,doj,salary,dept\_name) values (100,'tanya','manager','2021-09-11',30000,'testing')

... insert into employee\_details(emp\_id,emp\_name,designation,doj,salary,dept\_name) values (111,'sriram','associate','2021-06-11',25000,'development')

... insert into employee\_details(emp\_id,emp\_name,designation,doj,salary,dept\_name) values (121,'shiva','manager','2021-01-03',35000,'hr')

... apply batch;

cqlsh:employee\_info> select \* from employee\_details;

emp\_id | salary | dept\_name | designation | doj | emp\_name

+ + + + +

111 | 25000 | development | associate | 2021-06-10 18:30:00.000000+0000 | sriram

121 | 35000 | hr | manager | 2021-01-02 18:30:00.000000+0000 | shiva

100 | 30000 | testing | manager | 2021-09-10 18:30:00.000000+0000 | tanya

(3 rows)

cqlsh:employee\_info> update employee\_details set emp\_name='adp' 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 | 2021-06-10 18:30:00.000000+0000 | sriram

121 | 35000 | hr | manager | 2021-01-02 18:30:00.000000+0000 | adp

100 | 30000 | testing | manager | 2021-09-10 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 | 2021-06-10 18:30:00.000000+0000 | sriram | chat app

121 | 35000 | hr | manager | 2021-01-02 18:30:00.000000+0000 | adp | campusx 100 | 30000 | testing | manager | 2021-09-10 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','2021-09-09',30000,'testing') using ttl 30;

cqlsh:employee\_info> select ttl(emp\_name) from employee\_details where emp\_id=113 and salary=30000;

ttl(emp\_name)

29

(1 rows)

cqlsh:employee\_info> select \* from employee\_details;

emp\_id | salary | dept\_name | designation | doj | emp\_name | project

+ + + + + +

111 | 25000 | development | associate | 2021-06-10 18:30:00.000000+0000 | sriram | chat app

113 | 30000 | testing | manager | 2021-09-08 18:30:00.000000+0000 | sam | null 121 | 35000 | hr | manager | 2021-01-02 18:30:00.000000+0000 | adp | campusx 100 | 30000 | testing | manager | 2021-09-10 18:30:00.000000+0000 | tanya | canteen

app

(4 rows)

cqlsh:employee\_info> select \* from employee\_details;

emp\_id | salary | dept\_name | designation | doj | emp\_name | project

+ + + + + +

111 | 25000 | development | associate | 2021-06-10 18:30:00.000000+0000 | sriram | chat app

121 | 35000 | hr | manager | 2021-01-02 18:30:00.000000+0000 | adp | campusx 100 | 30000 | testing | manager | 2021-09-10 18:30:00.000000+0000 | tanya | canteen

app

(3 rows)

cqlsh:employee\_info> paging off; Disabled Query paging.

cqlsh:employee\_info> select \* from employee\_details where emp\_id in (111,121,100) order by salary;

emp\_id | salary | dept\_name | designation | doj | emp\_name | project

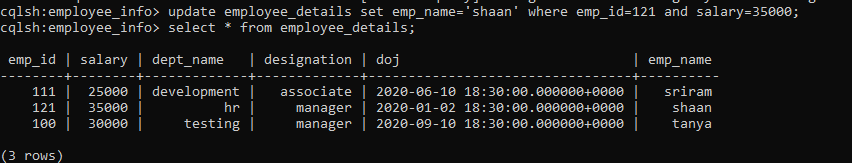
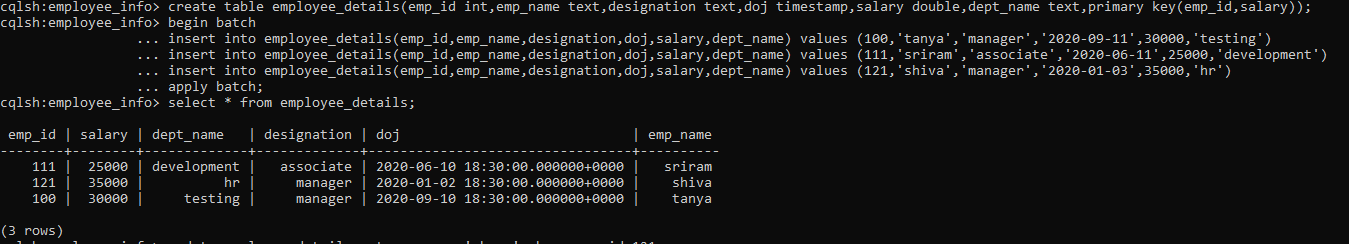
+ + + + + +

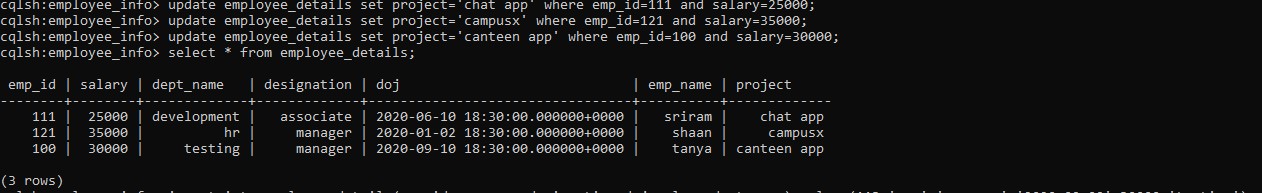
111 | 25000 | development | associate | 2021-06-10 18:30:00.000000+0000 | sriram | chat app

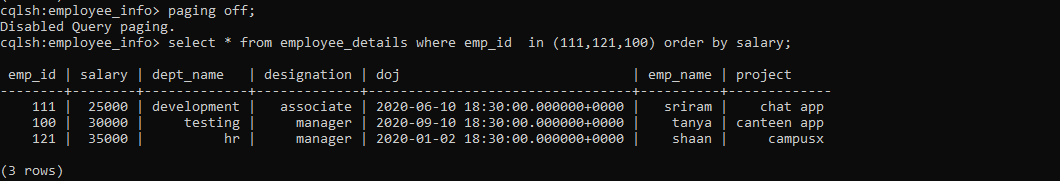
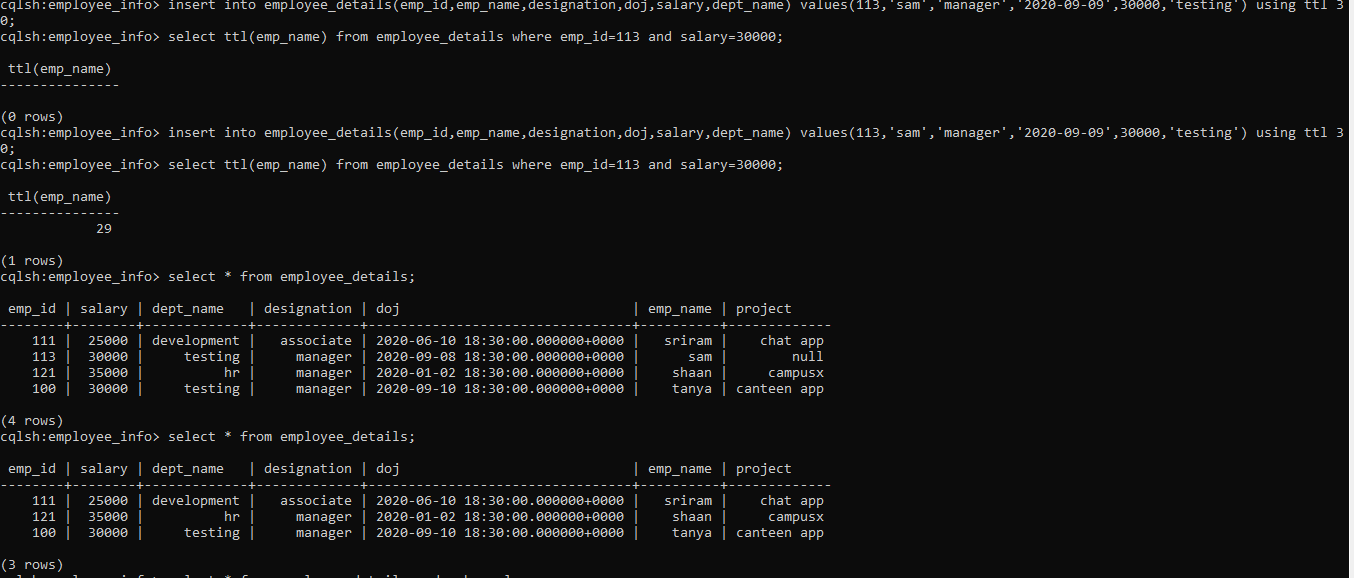
100 | 30000 | testing | manager | 2021-09-10 18:30:00.000000+0000 | tanya | canteen app

121 | 35000 | hr | manager | 2021-01-02 18:30:00.000000+0000 | adp | campusx

(3 rows) SCREENSHOTS -







# LIBRARY DATABASE (CASSANDRA)

Question -

Perform the following DB operations using Cassandra. 1.Create a keyspace by name Library

1. 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

1. Insert the values into the table in batch
2. Display the details of the table created and increase the value of the counter
3. Write a query to show that a student with id 112 has taken a book “BDA” 2 times.
4. Export the created column to a csv file
5. Import a given csv dataset from local file system into Cassandra column family

cqlsh> create keyspace library\_info with replication =

{'class':'SimpleStrategy','replication\_factor':1}; cqlsh> use library\_info;

cqlsh:library\_info> create table library\_details(stud\_id int,counter\_value counter,stud\_name text,book\_name text,date\_of\_issue timestamp,book\_id int,primary key(stud\_id,stud\_name,book\_name,date\_of\_issue,book\_id));

cqlsh:library\_info> update library\_details set counter\_value=counter\_value+1 where stud\_id=111 and stud\_name='sam' and book\_name='ML' and date\_of\_issue='2021-11-09' and book\_id=200;

cqlsh:library\_info> update library\_details set counter\_value=counter\_value+1 where stud\_id=112 and stud\_name='adp' and book\_name='BDA' and date\_of\_issue='2021-01-01' and book\_id=300;

cqlsh:library\_info> update library\_details set counter\_value=counter\_value+1 where stud\_id=113 and stud\_name='ayman' and book\_name='OOMD' and date\_of\_issue='2021-06-01' and book\_id=400;

cqlsh:library\_info> select \* from library\_details;

stud\_id | stud\_name | book\_name | date\_of\_issue | book\_id | counter\_value

+ + + + +

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 111 | | sam | | ML | 2021-11-08 18:30:00.000000+0000 | 200 | | 1 |  | |
| 113 | | ayman | | OOMD | 2021-05-31 18:30:00.000000+0000 | 400 | |  |  | 1 |
| 112 | | adp | | BDA | 2020-12-31 18:30:00.000000+0000 | 300 | |  | 1 |  |

(3 rows)

cqlsh:library\_info> update library\_details set counter\_value=counter\_value+1 where stud\_id=112 and stud\_name='adp' and book\_name='BDA' and date\_of\_issue='2021-01-01' and book\_id=300;

cqlsh:library\_info> select \* from library\_details where stud\_id=112;

stud\_id | stud\_name | book\_name | date\_of\_issue | book\_id | counter\_value

+ + + + + 112 | adp | BDA | 2020-12-31 18:30:00.000000+0000 | 300 | 2

(1 rows)

cqlsh:library\_info> copy library\_details(stud\_id,stud\_name,book\_name,book\_id,date\_of\_issue,counter\_value) to 'E:\sample.csv';

Using 3 child processes

Starting copy of library\_info.library\_details with columns [stud\_id, stud\_name, book\_name, book\_id, date\_of\_issue, counter\_value].

Processed: 3 rows; Rate: 1 rows/s; Avg. rate: 1 rows/s 3 rows exported to 1 files in 3.684 seconds.

cqlsh:library\_info> truncate library\_details;

cqlsh:library\_info> copy library\_details(stud\_id,stud\_name,book\_name,book\_id,date\_of\_issue,counter\_value) from 'E:\sample.csv';

Using 3 child processes

Starting copy of library\_info.library\_details with columns [stud\_id, stud\_name, book\_name, book\_id, date\_of\_issue, counter\_value].

Processed: 3 rows; Rate: 1 rows/s; Avg. rate: 1 rows/s 3 rows imported from 1 files in 2.602 seconds (0 skipped).

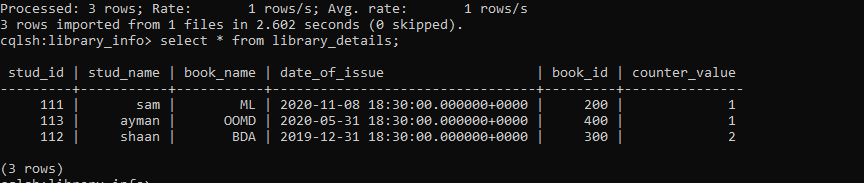
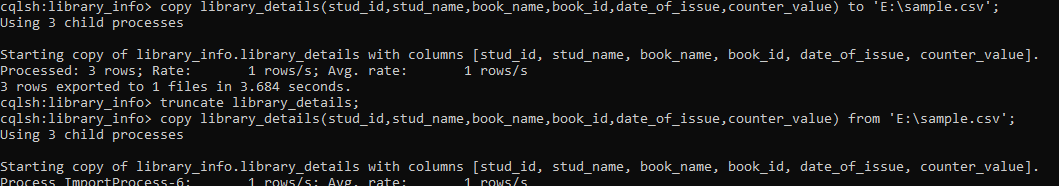
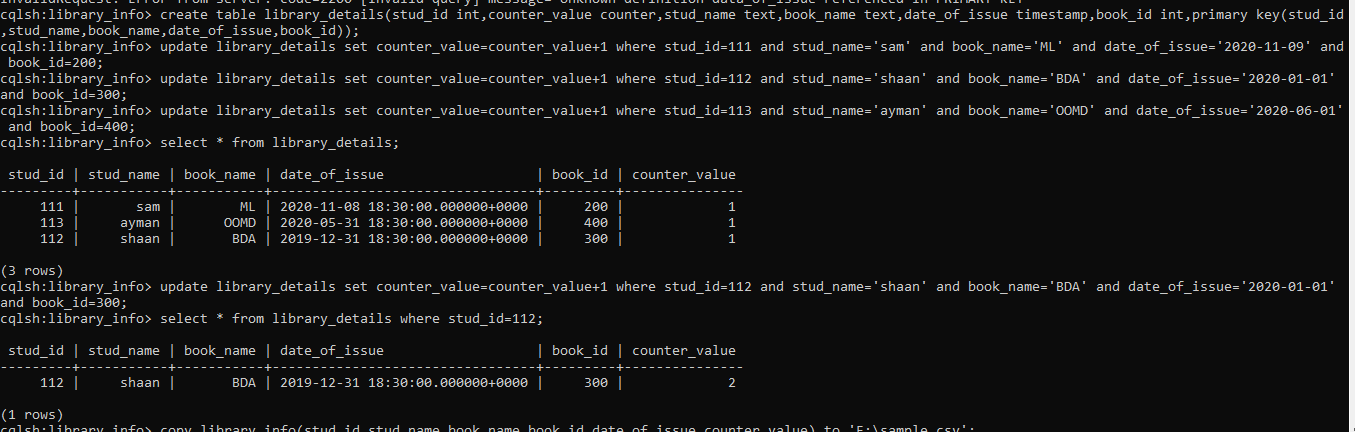
cqlsh:library\_info> select \* from library\_details;

stud\_id | stud\_name | book\_name | date\_of\_issue | book\_id | counter\_value

+ + + + +

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 111 | | sam | | ML | 2021-11-08 18:30:00.000000+0000 | 200 | | 1 |  |
| 113 | | ayman | | OOMD | 2021-05-31 18:30:00.000000+0000 | 400 | |  | 1 |
| 112 |  (3 rows) | adp | | BDA | 2020-12-31 18:30:00.000000+0000 | 300 | |  | 2 |

SCREENSHOTS -



# MONGODB SAMPLE

Question -

Perform the following DB operations using MongoDB.

1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.
2. Insert appropriate values
3. Write a query to update Email-Id of a student with rollno 10.
4. Replace the student name from “ABC” to “FEM” of rollno 11.
5. Export the created table into local file system
6. Drop the table
7. Import a given csv dataset from the local file system into mongodb collection.

use studentdb

switched to db studentdb

db.createCollection("student\_details")

{ "ok" : 1 }

db.student\_details.insert({'name':'abc','rollno':1,'age':19,'contactno':9090909090,'email':'abc@lab. com'})

WriteResult({ "nInserted" : 1 })

db.student\_details.insert({'name':'mno','rollno':2,'age':20,'contactno':9999900000,'email':'mno@l ab.com'})

WriteResult({ "nInserted" : 1 })

db.student\_details.insert({'name':'xyz','rollno':3,'age':21,'contactno':9999911111,'email':'xyz@lab

.com'})

WriteResult({ "nInserted" : 1 })

db.student\_details.find({})

{ "\_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "[abc@lab.com"](mailto:abc@lab.com) }

{ "\_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : ["mno@lab.com"](mailto:mno@lab.com) }

{ "\_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "xyz", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "[xyz@lab.com"](mailto:xyz@lab.com) }

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

db.student\_details.find({'rollno':3})

{ "\_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "xyz", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "[update@lab.com"](mailto:update@lab.com) }

db.student\_details.update({'name':'xyz'},{$set:{'name':'pqr'}}) WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 }) db.student\_details.find({'name':'pqr'})

{ "\_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "pqr", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "[update@lab.com"](mailto:update@lab.com) }

mongoexport --db studentdb --collection student\_details --out E:\Desktop\sample.json 2022-05-22T10:43:30.687+0530 connected to: mongodb://localhost/

2022-05-22T10:43:31.026+0530 exported 3 records

db.getCollection('student\_details').drop() true

mongoimport --db studentdb --collection student\_details --type=json --file= E:\Desktop\sample.json

2022-05-22T10:46:49.898+0530 connected to: mongodb://localhost/

2022-05-22T10:46:50.044+0530 3 document(s) imported successfully. 0 document(s) failed to import.

db.student\_details.find({})

{ "\_id" : ObjectId("60a88f8fffecf7c8abe76777"), "name" : "pqr", "rollno" : 3, "age" : 21, "contactno" : 9999911111, "email" : "[update@lab.com"](mailto:update@lab.com) }

{ "\_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "[abc@lab.com"](mailto:abc@lab.com) }

{ "\_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : ["mno@lab.com"](mailto:mno@lab.com) }

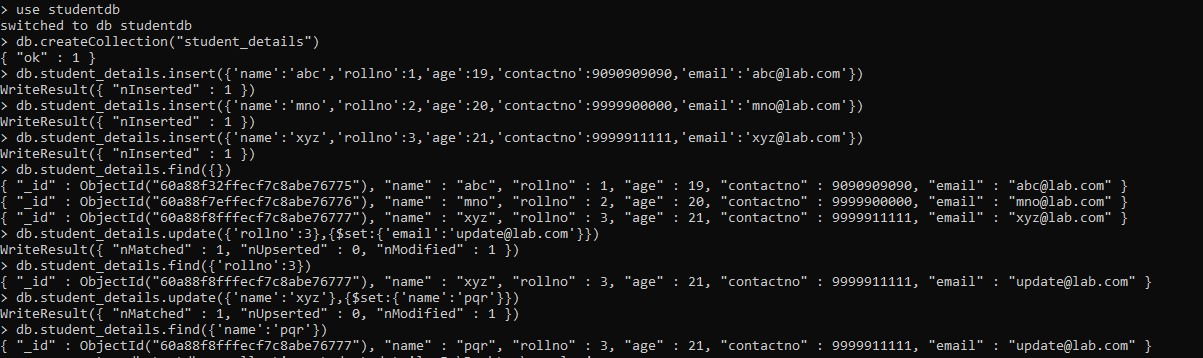
db.student\_details.remove({age:{$gt:20}}) WriteResult({ "nRemoved" : 1 })

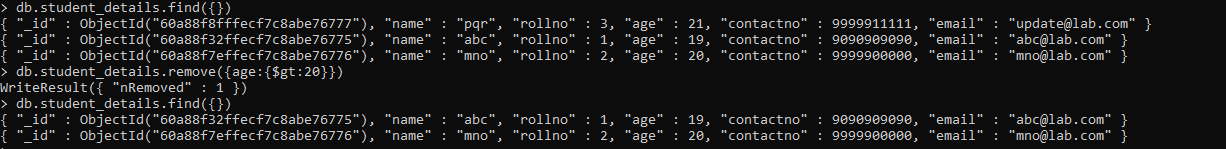
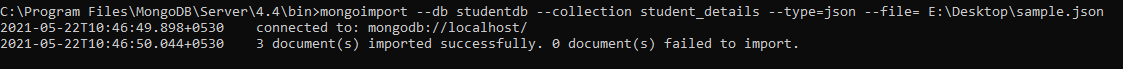
db.student\_details.find({})

{ "\_id" : ObjectId("60a88f32ffecf7c8abe76775"), "name" : "abc", "rollno" : 1, "age" : 19, "contactno" : 9090909090, "email" : "[abc@lab.com"](mailto:abc@lab.com) }

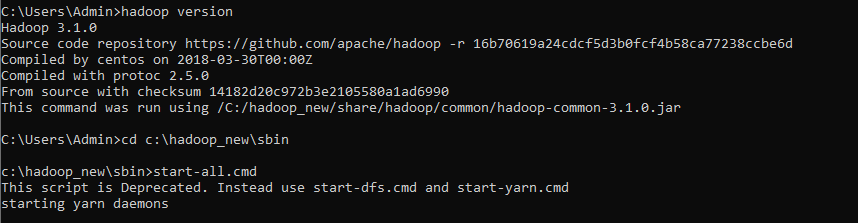
{ "\_id" : ObjectId("60a88f7effecf7c8abe76776"), "name" : "mno", "rollno" : 2, "age" : 20, "contactno" : 9999900000, "email" : "[mno@lab.com"](mailto:mno@lab.com) }

SCREENSHOTS -





# SCREENSHOT OF HADOOP INSTALLATION



# HADOOP SAMPLE

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

c:\hadoop\_new\sbin>hdfs dfs -mkdir /temp

c:\hadoop\_new\sbin>hdfs dfs -copyFromLocal E:\Desktop\sample.txt \temp c:\hadoop\_new\sbin>hdfs dfs -ls \temp

Found 1 items

-rw-r--r-- 1 Admin supergroup 11 2022-06-11 21:12 /temp/sample.txt

c:\hadoop\_new\sbin>hdfs dfs -cat \temp\sample.txt hello world

c:\hadoop\_new\sbin>hdfs dfs -get \temp\sample.txt E:\Desktop\temp c:\hadoop\_new\sbin>hdfs dfs -put E:\Desktop\temp \temp

c:\hadoop\_new\sbin>hdfs dfs -ls \temp Found 2 items

-rw-r--r-- 1 Admin supergroup 11 2022-06-11 21:12 /temp/sample.txt

drwxr-xr-x - Admin supergroup 0 2022-06-11 21:15 /temp/temp c:\hadoop\_new\sbin>hdfs dfs -mv \lab1 \temp

c:\hadoop\_new\sbin>hdfs dfs -ls \temp Found 3 items

drwxr-xr-x - Admin supergroup 0 2022-04-19 15:07 /temp/lab1

-rw-r--r-- 1 Admin supergroup 11 2022-06-11 21:12 /temp/sample.txt

drwxr-xr-x - Admin supergroup 0 2022-06-11 21:15 /temp/temp

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

c:\hadoop\_new\sbin>hdfs dfs -ls \temp Found 2 items

drwxr-xr-x - Admin supergroup 0 2022-04-19 15:07 /temp/lab1

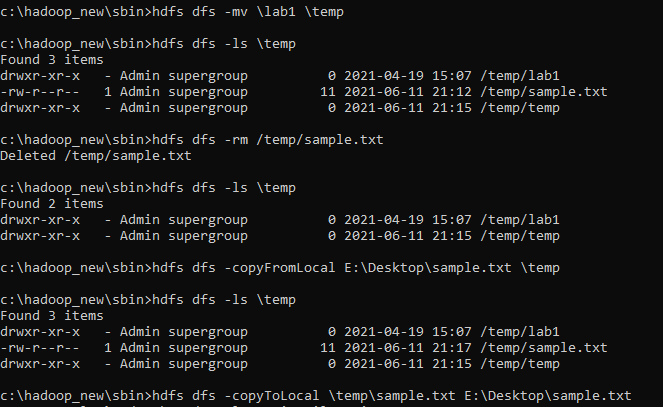
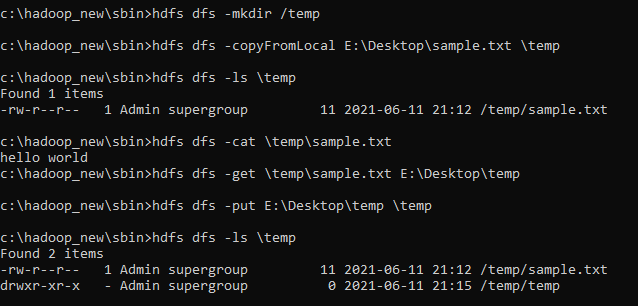
drwxr-xr-x - Admin supergroup 0 2022-06-11 21:15 /temp/temp c:\hadoop\_new\sbin>hdfs dfs -copyFromLocal E:\Desktop\sample.txt \temp

c:\hadoop\_new\sbin>hdfs dfs -ls \temp Found 3 items

drwxr-xr-x - Admin supergroup 0 2022-04-19 15:07 /temp/lab1

-rw-r--r-- 1 Admin supergroup 11 2022-06-11 21:17 /temp/sample.txt

drwxr-xr-x - Admin supergroup 0 2022-06-11 21:15 /temp/temp c:\hadoop\_new\sbin>hdfs dfs -copyToLocal \temp\sample.txt E:\Desktop\sample.txt SCREENSHOTS -



# MAPREDUCE TEMPERATURE

For the given file, Create a Map Reduce program to

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

// AverageDriver.java package temperature;

import org.apache.hadoop.io.\*; import org.apache.hadoop.fs.\*;

import org.apache.hadoop.mapreduce.\*;

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

public class AverageDriver

{

public static void main (String[] args) throws Exception

{

if (args.length != 2)

{

System.err.println("Please Enter the input and output parameters"); System.exit(-1);

}

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

job.setMapperClass(AverageMapper.class);

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

}

}

//AverageMapper.java package temperature;

import org.apache.hadoop.io.\*;

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

public class AverageMapper extends Mapper <LongWritable, Text, Text, IntWritable>

{

public static final int MISSING = 9999;

public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException

{

String line = value.toString(); String year = line.substring(15,19); int temperature;

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

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

else

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

String quality = line.substring(92, 93);

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

}

}

//AverageReducer.java package temperature;

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

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

{

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

{

int max\_temp = 0; int count = 0;

for (IntWritable value : values)

{

max\_temp += value.get(); count+=1;

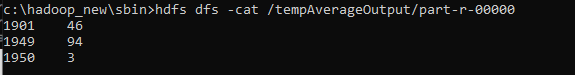
}

context.write(key, new IntWritable(max\_temp/count));

}

}

SCREENSHOT -



1. Find the mean max temperature for every month.

//TempDriver.java package temperatureMax;

import org.apache.hadoop.io.\*; import org.apache.hadoop.fs.\*;

import org.apache.hadoop.mapreduce.\*;

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

public class TempDriver

{

public static void main (String[] args) throws Exception

{

if (args.length != 2)

{

System.err.println("Please Enter the input and output parameters"); System.exit(-1);

}

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

job.setMapperClass(TempMapper.class);

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

}

}

//TempMapper.java package temperatureMax;

import org.apache.hadoop.io.\*;

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

public class TempMapper extends Mapper <LongWritable, Text, Text, IntWritable>

{

public static final int MISSING = 9999;

public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException

{

String line = value.toString();

String month = line.substring(19,21); int temperature;

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

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

else

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

String quality = line.substring(92, 93);

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

}

}

//TempReducer.java package temperatureMax;

import org.apache.hadoop.io.\*;

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

public class TempMapper extends Mapper <LongWritable, Text, Text, IntWritable>

{

public static final int MISSING = 9999;

public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException

{

String line = value.toString();

String month = line.substring(19,21); int temperature;

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

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

else

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

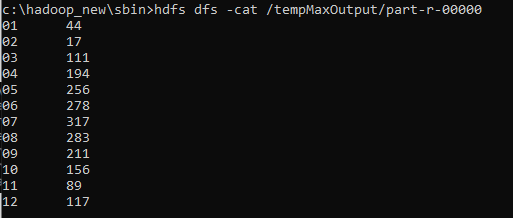
String quality = line.substring(92, 93);

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

}

}

SCREENSHOT -



# MAPREDUCE TOPN

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

// TopN.java package sortWords;

import org.apache.hadoop.conf.Configuration; import org.apache.hadoop.fs.Path;

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

import org.apache.hadoop.mapreduce.Job; import org.apache.hadoop.mapreduce.Mapper; import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; import org.apache.hadoop.util.GenericOptionsParser;

import utils.MiscUtils;

import java.io.IOException; import java.util.\*;

public class TopN {

public static void main(String[] args) throws Exception { Configuration conf = new Configuration();

String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs(); if (otherArgs.length != 2) {

System.err.println("Usage: TopN <in> <out>");

System.exit(2);

}

Job job = Job.getInstance(conf); job.setJobName("Top N"); job.setJarByClass(TopN.class); job.setMapperClass(TopNMapper.class);

//job.setCombinerClass(TopNReducer.class); job.setReducerClass(TopNReducer.class); job.setOutputKeyClass(Text.class); job.setOutputValueClass(IntWritable.class); FileInputFormat.addInputPath(job, new Path(otherArgs[0])); FileOutputFormat.setOutputPath(job, new Path(otherArgs[1])); System.exit(job.waitForCompletion(true) ? 0 : 1);

}

/\*\*

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

\*/

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

private final static IntWritable one = new IntWritable(1); private Text word = new Text();

private String tokens = "[\_|$#<>\\^=\\[\\]\\\*/\\\\,;,.\\-:()?!\"']";

@Override

public void map(Object key, Text value, Context context) throws IOException, InterruptedException {

String cleanLine = value.toString().toLowerCase().replaceAll(tokens, " "); StringTokenizer itr = new StringTokenizer(cleanLine);

while (itr.hasMoreTokens()) {

word.set(itr.nextToken().trim()); context.write(word, one);

}

}

}

/\*\*

* The reducer retrieves every word and puts it into a Map: if the word already exists in the
* map, increments its value, otherwise sets it to 1.

\*/

public static class TopNReducer extends Reducer<Text, IntWritable, Text, IntWritable> { private Map<Text, IntWritable> countMap = new HashMap<>();

@Override

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

// computes the number of occurrences of a single word int sum = 0;

for (IntWritable val : values) { sum += val.get();

}

// puts the number of occurrences of this word into the map.

// We need to create another Text object because the Text instance

// we receive is the same for all the words countMap.put(new Text(key), new IntWritable(sum));

}

@Override

protected void cleanup(Context context) throws IOException, InterruptedException { Map<Text, IntWritable> sortedMap = MiscUtils.sortByValues(countMap);

int counter = 0;

for (Text key : sortedMap.keySet()) { if (counter++ == 3) {

break;

}

context.write(key, sortedMap.get(key));

}

}

}

/\*\*

* The combiner retrieves every word and puts it into a Map: if the word already exists in the
* map, increments its value, otherwise sets it to 1.

\*/

public static class TopNCombiner extends Reducer<Text, IntWritable, Text, IntWritable> {

@Override

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

// computes the number of occurrences of a single word int sum = 0;

for (IntWritable val : values) { sum += val.get();

}

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

}

}

}

// MiscUtils.java package utils;

import java.util.\*; public class MiscUtils {

/\*\*

* sorts the map by values. Taken from:
* <http://javarevisited.blogspot.it/2012/12/how-to-sort-hashmap-java-by-key-and-value.html>

\*/

public static <K extends Comparable, V extends Comparable> Map<K, V> sortByValues(Map<K, V> map) {

List<Map.Entry<K, V>> entries = new LinkedList<Map.Entry<K, V>>(map.entrySet()); Collections.sort(entries, new Comparator<Map.Entry<K, V>>() {

@Override

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

}

});

//LinkedHashMap will keep the keys in the order they are inserted

//which is currently sorted on natural ordering

Map<K, V> sortedMap = new LinkedHashMap<K, V>(); for (Map.Entry<K, V> entry : entries) {

sortedMap.put(entry.getKey(), entry.getValue());

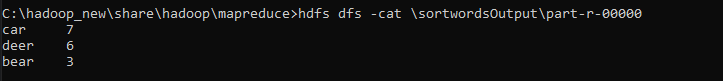
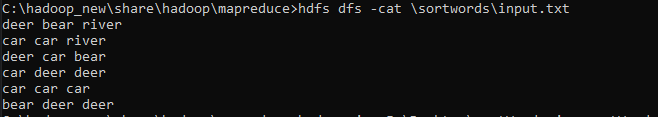
}

return sortedMap;

}

}

SCREENSHOTS -



# MAPREDUCE JOIN

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

// JoinDriver.java

import org.apache.hadoop.conf.Configured; import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.Text; import org.apache.hadoop.mapred.\*;

import org.apache.hadoop.mapred.lib.MultipleInputs; import org.apache.hadoop.util.\*;

public class JoinDriver extends Configured implements Tool {

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

public void configure(JobConf job) {}

numPartitions;

@Override

public int getPartition(TextPair key, Text value, int numPartitions) { return (key.getFirst().hashCode() & Integer.MAX\_VALUE) %

}

}

@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;

}

input'");

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

conf.setJobName("Join 'Department Emp Strength input' with 'Department Name

Path AInputPath = new Path(args[0]); Path BInputPath = new Path(args[1]); Path outputPath = new Path(args[2]);

Posts.class); User.class);

MultipleInputs.addInputPath(conf, AInputPath, TextInputFormat.class, MultipleInputs.addInputPath(conf, BInputPath, TextInputFormat.class,

FileOutputFormat.setOutputPath(conf, outputPath);

conf.setPartitionerClass(KeyPartitioner.class); conf.setOutputValueGroupingComparator(TextPair.FirstComparator.class);

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

JobClient.runJob(conf);

return 0;

}

public static void main(String[] args) throws Exception {

int exitCode = ToolRunner.run(new JoinDriver(), args); System.exit(exitCode);

}

}

// JoinReducer.java

import java.io.IOException; import java.util.Iterator;

import org.apache.hadoop.io.Text; import org.apache.hadoop.mapred.\*;

public class JoinReducer extends MapReduceBase implements Reducer<TextPair, Text, Text, Text> {

@Override

public void reduce (TextPair key, Iterator<Text> values, OutputCollector<Text, Text> output, Reporter reporter)

throws IOException

{

Text nodeId = new Text(values.next()); while (values.hasNext()) {

Text node = values.next();

Text outValue = new Text(nodeId.toString() + "\t\t" + node.toString()); output.collect(key.getFirst(), outValue);

}

}

}

// User.java

import java.io.IOException; import java.util.Iterator;

import org.apache.hadoop.conf.Configuration; import org.apache.hadoop.fs.FSDataInputStream; import org.apache.hadoop.fs.FSDataOutputStream; import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

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

import org.apache.hadoop.mapred.\*; import org.apache.hadoop.io.IntWritable;

public class User extends MapReduceBase implements Mapper<LongWritable, Text, TextPair, Text> {

@Override

public void map(LongWritable key, Text value, OutputCollector<TextPair, Text> output, Reporter reporter)

throws IOException

{

String valueString = value.toString();

String[] SingleNodeData = valueString.split("\t");

output.collect(new TextPair(SingleNodeData[0], "1"), new Text(SingleNodeData[1]));

}

}

//Posts.java

import java.io.IOException;

import org.apache.hadoop.io.\*; import org.apache.hadoop.mapred.\*;

public class Posts extends MapReduceBase implements Mapper<LongWritable, Text, TextPair, Text> {

@Override

public void map(LongWritable key, Text value, OutputCollector<TextPair, Text> output, Reporter reporter)

throws IOException

{

String valueString = value.toString();

String[] SingleNodeData = valueString.split("\t"); output.collect(new TextPair(SingleNodeData[3], "0"), new

Text(SingleNodeData[9]));

}

}

// TextPair.java import java.io.\*;

import org.apache.hadoop.io.\*;

public class TextPair implements WritableComparable<TextPair> {

private Text first; private Text second;

public TextPair() {

set(new Text(), new Text());

}

public TextPair(String first, String second) { set(new Text(first), new Text(second));

}

public TextPair(Text first, Text second) { set(first, second);

}

public void set(Text first, Text second) { this.first = first;

this.second = second;

}

public Text getFirst() { return first;

}

public Text getSecond() { return second;

}

@Override

public void write(DataOutput out) throws IOException { first.write(out);

second.write(out);

}

@Override

public void readFields(DataInput in) throws IOException { first.readFields(in);

second.readFields(in);

}

@Override

public int hashCode() {

return first.hashCode() \* 163 + second.hashCode();

}

@Override

public boolean equals(Object o) { if (o instanceof TextPair) { TextPair tp = (TextPair) o;

return first.equals(tp.first) && second.equals(tp.second);

}

return false;

}

@Override

public String toString() { return first + "\t" + second;

}

@Override

public int compareTo(TextPair tp) { int cmp = first.compareTo(tp.first); if (cmp != 0) {

return cmp;

}

return second.compareTo(tp.second);

}

// ^^ TextPair

// vv TextPairComparator

public static class Comparator extends WritableComparator {

private static final Text.Comparator TEXT\_COMPARATOR = new Text.Comparator(); public Comparator() {

super(TextPair.class);

}

@Override

public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {

try {

int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1); int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);

int cmp = TEXT\_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2); if (cmp != 0) {

return cmp;

}

return TEXT\_COMPARATOR.compare(b1, s1 + firstL1, l1 - firstL1, b2, s2 + firstL2, l2 - firstL2);

} catch (IOException e) {

throw new IllegalArgumentException(e);

}

}

}

static {

WritableComparator.define(TextPair.class, new Comparator());

}

public static class FirstComparator extends WritableComparator {

private static final Text.Comparator TEXT\_COMPARATOR = new Text.Comparator(); public FirstComparator() {

super(TextPair.class);

}

@Override

public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {

try {

int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1); int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2); return TEXT\_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);

} catch (IOException e) {

throw new IllegalArgumentException(e);

}

}

@Override

public int compare(WritableComparable a, WritableComparable b) { if (a instanceof TextPair && b instanceof TextPair) {

return ((TextPair) a).first.compareTo(((TextPair) b).first);

}

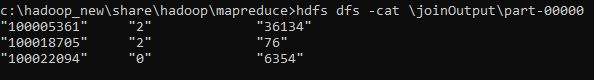
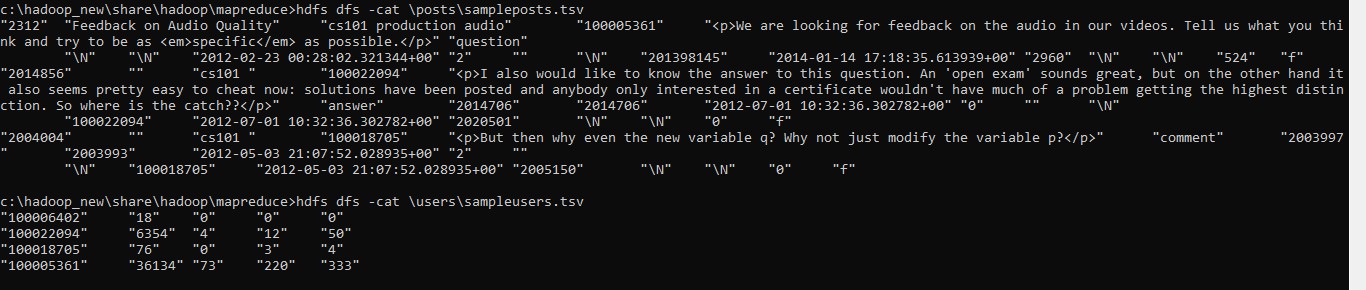
return super.compare(a, b);

}

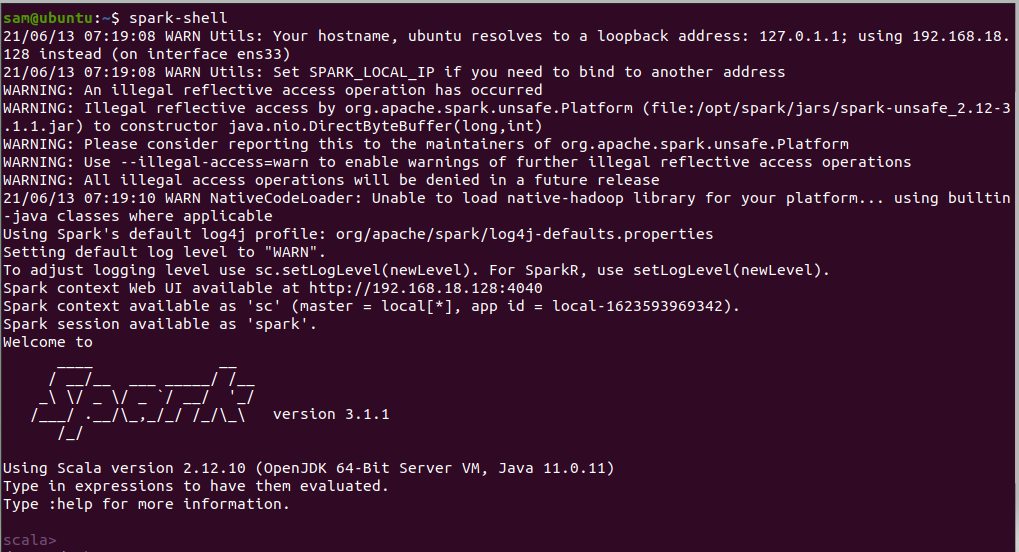
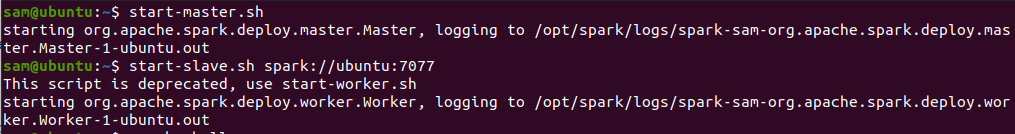
}

}

SCREENSHOTS -



# SCALA INSTALLATION SCREENSHOT



# SCALA WORDCOUNT

// scala shell

scala> val textfile = sc.textFile("/home/sam/Desktop/abc.txt")

textfile: org.apache.spark.rdd.RDD[String] = /home/sam/Desktop/abc.txt MapPartitionsRDD[1] at textFile at <console>:24

scala> val counts = textfile.flatMap(line => line.split(" ")).map(word => (word,1)).reduceByKey(+)

counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at

<console>:25

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

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

scala> println(sorted)

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

scala> for((k,v)<-sorted)

| {

| if(v>4)

| {

| println(k+"-"+v)

| }

| } car-7 deer-5

