**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JnanaSangama”, Belgaum -590014, Karnataka.**

****

**LAB REPORT**

**on**

**Big Data Analytics**

***Submitted by***

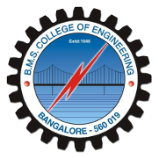
**Mukund Raghavan Sadavarthi (1BM22CS166)**

***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**

**Feb-2024 to July-2024**

**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 “LAB COURSE **Big Data Analytics**” carried out by **Mukund Ragahavan Sadavarthi (1BM22CS166),** 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 2024. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics - (23CS6PCBDA)** work prescribed for the said degree.

Ramya K M              **Dr. Kavitha Sooda**

Assistant Professor Professor and Head

Department of CSE Department of CSE

BMSCE, Bengaluru BMSCE, Bengaluru

`

**Index Sheet**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Experiment Title** | **Page No.** |
| 1 | MongoDB- CRUD Operations Demonstration (Practice and Self Study) | 1 |
| 2 | Perform the following DB operations using Cassandra. | 3 |
| 3 | Perform the following DB operations using Cassandra | 5 |
| 4 | Execution of HDFS Commands for interaction with Hadoop Environment. | 7 |
| 5 | Implement Wordcount program on Hadoop framework | 8 |
| 6 | Create a MapReduce program to find average temperature for each year from data set. find the mean max temperature for every month | 11 |
| 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. | 14 |
| 8 | Write a Scala program to print numbers from 1 to 100 using for loop. | 16 |
| 9 | 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. | 17 |

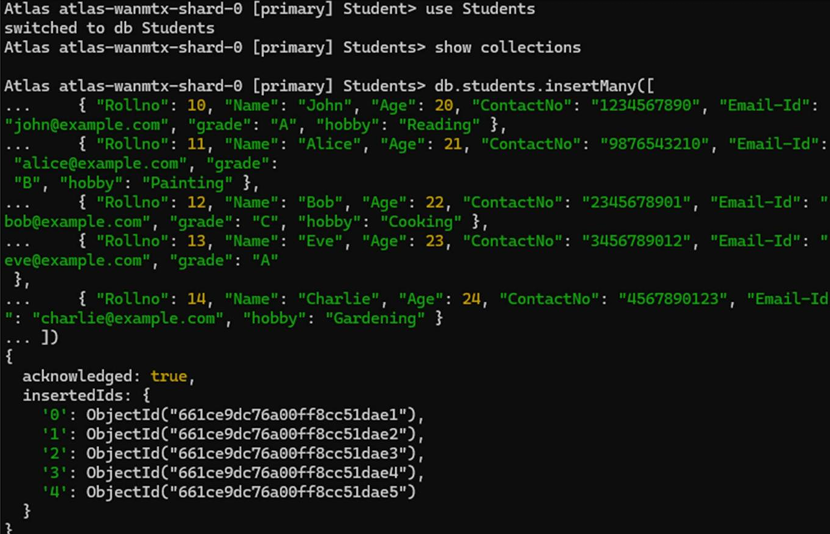
**Course Outcome**

|  |  |
| --- | --- |
| CO1 | Apply the concepts of NoSQL, Hadoop, Spark for a given task |
| CO2 | Analyse data analytic techniques for a given problem . |
| CO3 | Conduct experiments using data analytics mechanisms for a given problem. |

**Experiment – 1**

MongoDB- CRUD Operations Demonstration (Practice and Self Study)

Code Outputs:



A computer screen shot of a program code

AI-generated content may be incorrect.

A computer screen with a black screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Experiment – 2**

Perform the following DB operations using Cassandra.

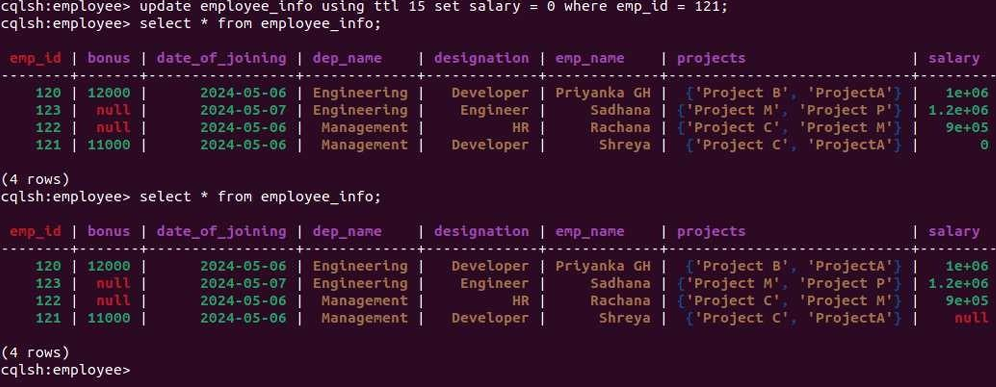
* Create a keyspace by name Employee
* Create a column family by name Employee-Info with attributes Emp\_Id Primary Key, Emp\_Name, Designation, Date\_of\_Joining, Salary, Dept\_Name
* Insert the values into the table in batch
* Update Employee name and Department of Emp-Id 121
* Sort the details of Employee records based on salary
* Alter the schema of the table Employee\_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.
* Update the altered table to add project names.
* Create a TTL of 15 seconds to display the values of Employees.

Codes Output:

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Experiment – 3**

Perform the following DB operations using Cassandra:

● Create a keyspace by name Library

● 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

● Insert the values into the table in batch

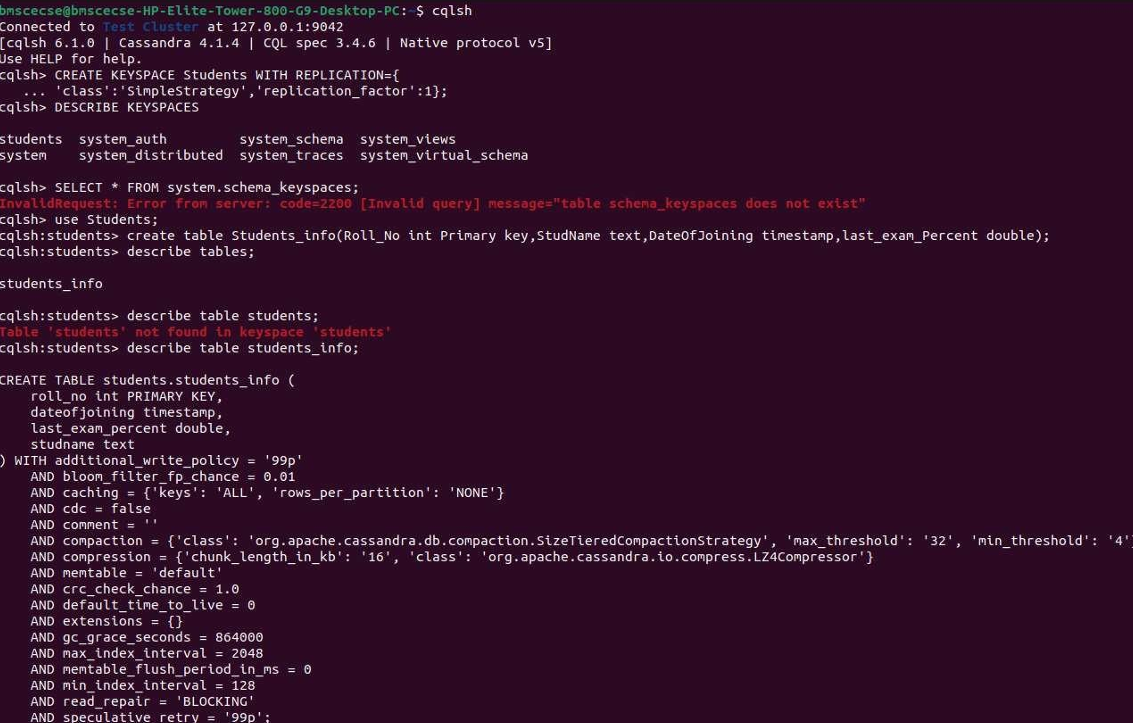
● Display the details of the table created and increase the value of the counter

● Write a query to show that a student with id 112 has taken a book “BDA” 2 times.

● Export the created column to a csv file

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

Codes Output:



A screenshot of a computer

AI-generated content may be incorrect.

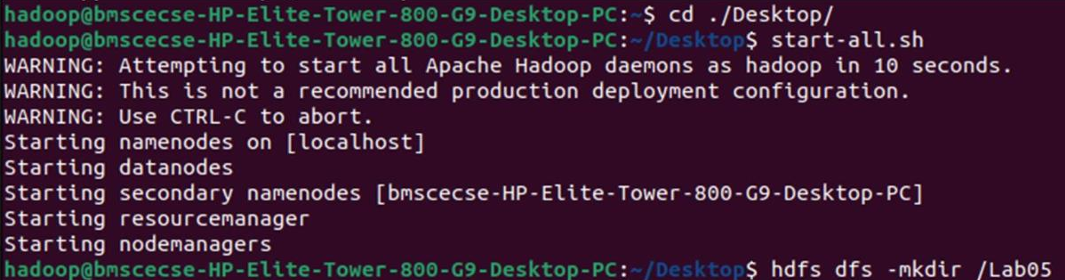
A screenshot of a computer program

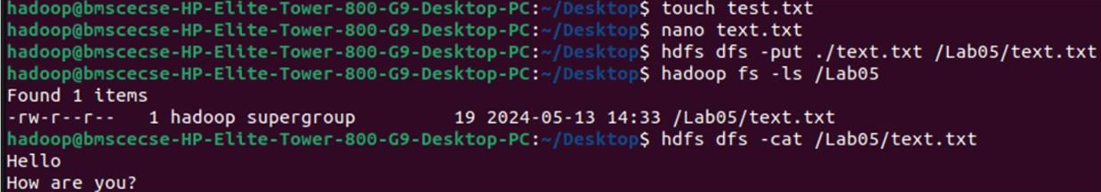
AI-generated content may be incorrect.

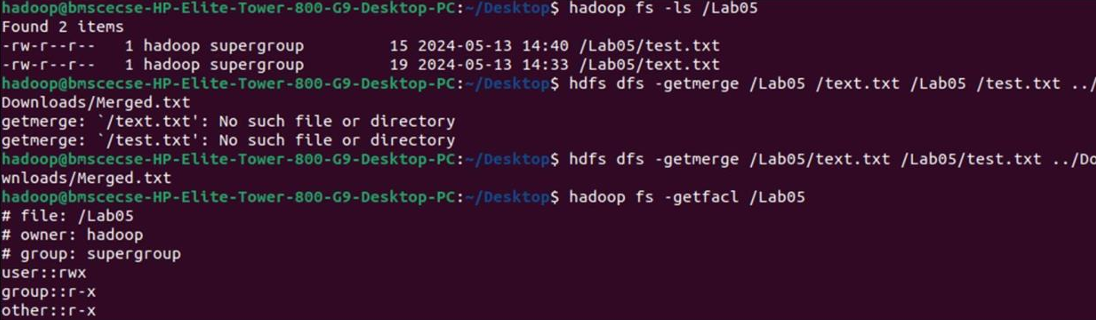
**Experiment-4**

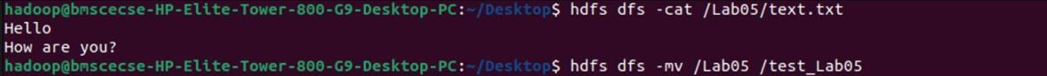
Execution of HDFS Commands for interaction with Hadoop Environment

Codes Output:









A computer screen with numbers and letters

AI-generated content may be incorrect.

**Experiment-5**

Implement Wordcount program on Hadoop framework

Mapper:

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> {

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

throws IOException

{

String line = value.toString();

for (String word : line.split(" "))

{

if (word.length() > 0)

{

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

Reducer:

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));

}}

Driver:

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;

}

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

{

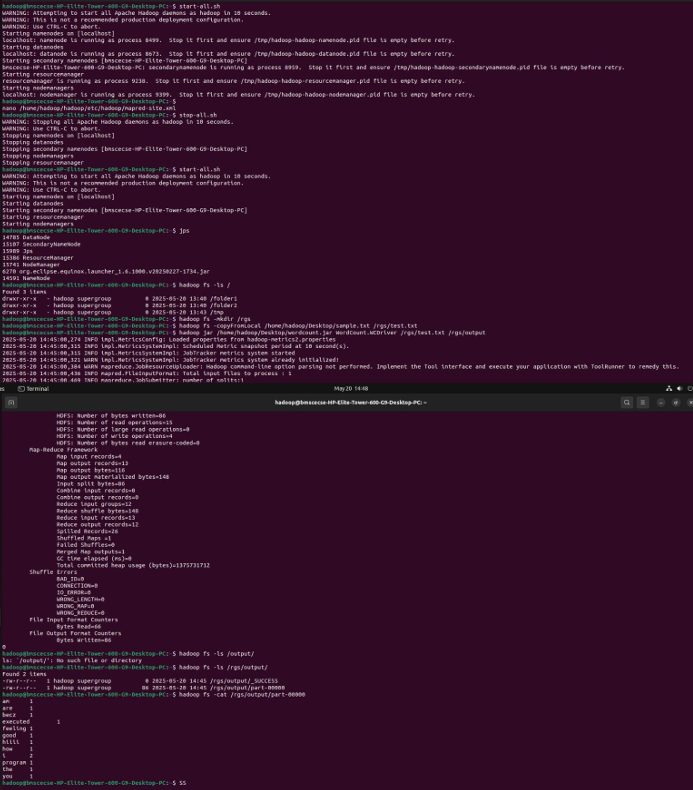
int exitCode = ToolRunner.run(new WCDriver(), args);

System.out.println(exitCode);

}

}

Codes Output:



**Experiment-6**

From the following link extract the weather data: https://github.com/tomwhite/hadoop-book/tree/master/input/ncdc/all

Create a Map Reduce program to:

* Find average temperature for each year from NCDC data set.

Find the mean max temperature for every month.

Mapper:

*#!/usr/bin/env python3*

import sys

for line in sys.stdin:

    line = line.strip()

    parts = line.split()

    date, temp = parts

    temp = float(temp)

    print(f"{date}\t{temp}")

Reducer1:

*#!/usr/bin/env python3*

import sys

count = 0

total\_temp = 0.0

for line in sys.stdin:

    line = line.strip()

    key, value = line.split("\t")

    try:

        total\_temp += float(value)

        count += 1

    except ValueError:

        continue

if count > 0:

    mean\_temp = total\_temp / count

    print(f"Mean Temperature: {mean\_temp:.2f}")

else:

    print("No valid temperature records.")

Reducer2:

#!/usr/bin/env python3

import sys

max\_temp = float('-inf')

for line in sys.stdin:

    line = line.strip()

    if not line:

        continue

    try:

        key, value = line.split("\t")

        temp = float(value)

        if temp > max\_temp:

            max\_temp = temp

    except ValueError:

        continue

if max\_temp != float('-inf'):

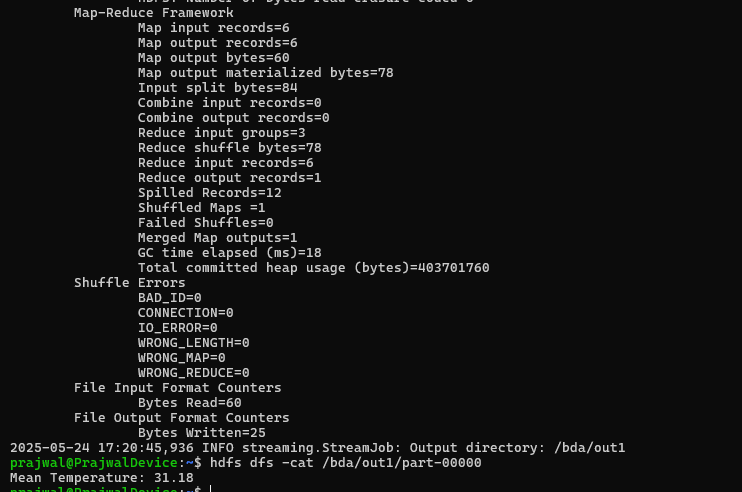
    print(f"Max Temperature: {max\_temp:.2f}")

else:

    print("No valid temperature records.")

A screenshot of a computer

AI-generated content may be incorrect.Codes Output:



**Experiment-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.

Mapper:

*#!/usr/bin/env python3*

import sys

import re

for line in sys.stdin:

    words = re.findall(r'\w+', line.lower())  *# normalize case*

    for word in words:

        print(f"{word}\t1")

Reducer:

#!/usr/bin/env python3

import sys

from collections import defaultdict

N = 10  # change this to desired Top-N

word\_counts = defaultdict(int)

# Aggregate word counts

for line in sys.stdin:

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

    word\_counts[word] += int(count)

# Sort by frequency desc, then word asc

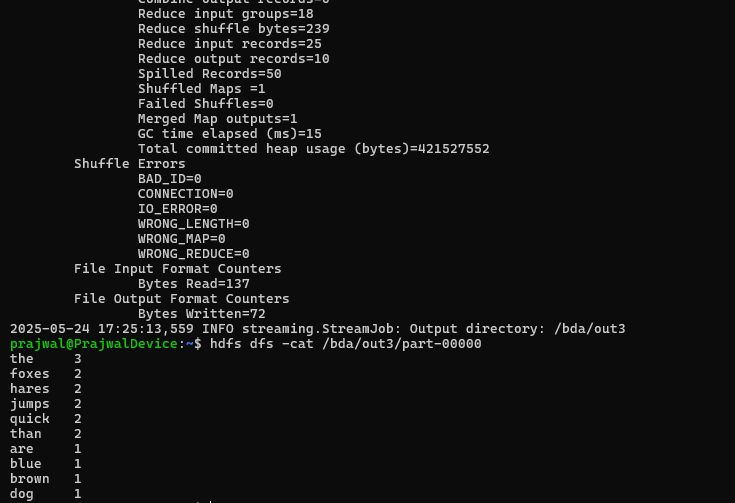
top\_n = sorted(word\_counts.items(), key=lambda x: (-x[1], x[0]))[:N]

# Output Top-N

for word, count in top\_n:

    print(f"{word}\t{count}")

Codes Output:



**Experiment-8**

Write a Scala program to print numbers from 1 to 100 using for loop.

Scala Code:

Scala> for(i <- 0 to 100){

println(i)

}

0

1

2

.

.

Codes Output:

**A screenshot of a computer

AI-generated content may be incorrect.**

**Experiment-9**

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.

A screenshot of a computer program

AI-generated content may be incorrect.Codes Output: