SRM VALLIAMMAI ENGINEERING COLLEGE

(An Autonomous Institution)

SRM Nagar, Kattankulathur-603203

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

ACADEMIC YEAR: 2025-2026

EVEN SEMESTER

LAB MANUAL

(REGULATION - 2023)

AD3663 – BIG DATA ANALYTICS LABORATORY

SIXTH SEMSTER

B. Tech – Artificial Intelligence and Data Science

Prepared By

V. ABARNA, A.P (O.G) / AI&DS

INDEX

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

1. To afford the necessary background in the field of Information Technology to deal with engineering problems to excel as engineering professionals in industries.
2. To improve the qualities like creativity, leadership, teamwork and skill thus contributing towards the growth and development of society.
3. To develop ability among students towards innovation and entrepreneurship that caters to the needs of Industry and society.
4. To inculcate and attitude for life-long learning process through the use of information technology sources.
5. To prepare then to be innovative and ethical leaders, both in their chosen profession and in other activities.

PROGRAMME OUTCOMES (POs)

After going through the four years of study, Information Technology Graduates will exhibit ability to:

PO#	Graduate Attribute	Programme Outcome
1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.
2	Problem analysis	Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3	Design/development of solutions	Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.
4	Conduct investigations of complex problems	Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5	Modern tool usage	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling to complex engineering activities, with an understanding of the limitations.
6	The engineer and society	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7	Environment and sustainability	Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings
10	Communication	Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions
11	Project management and finance	Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments
12	Life-long learning	Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

PROGRAMME SPECIFIC OUTCOMES (PSOs)

After the completion of Bachelor of Technology in Artificial Intelligence and Data Science programme the student will have following Program specific outcomes

1. Design and develop secured database applications with data analytical approaches of data preprocessing, optimization, visualization techniques and maintenance using state of the art methodologies based on ethical values.
2. Design and develop intelligent systems using computational principles, methods and systems for extracting knowledge from data to solve real time problems using advanced technologies and tools.
3. Design, plan and setting up the network that is helpful for contemporary business environments using latest software and hardware.
4. Planning and defining test activities by preparing test cases that can predict and correct errors ensuring a socially transformed product catering all technological needs.

AD3663	BIG DATA ANALYTICS LABORATORY	L T P C
		0 0 3 1.5

OBJECTIVES:

This course will enable students to

- To get familiar with Hadoop distributions, Configuring Hadoop and Performing File Managements tasks.
- To understand HDFS concepts and interfacing with HDFS. Solve problems using searching and backtracking.
- To learn Map Reduce analytics using Hadoop for processing Big Data.
- To Practice programming tools HIVE and HBase in Hadoop eco-system.
- To examine data processing operators and compare with traditional databases.

LIST OF EXPERIMENTS:

1. Downloading and Installing Hadoop: Understanding different Hadoop modes, Startup Scripts, Configuration files.
2. Hadoop Implementation of file management tasks, such as Adding files and directories Retrieving files and Deleting files.
3. Implement of Matrix Multiplications with Hadoop Map Reduce.
4. Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm.
5. Installation of HIVE along with practice examples
6. Installation of HBase, Installing thrift along with Practice examples.
7. Practice importing and exporting data from various databases.

Total: 45 Periods

LIST OF EQUIPMENTS FOR A BATCH OF 30 STUDENTS

SOFTWARE:

Cassandra, Hadoop, Java, Pig, Hive and HBase.

HARDWARE:

Standalone Desktops: 30 Nos.

COURSE OUTCOMES

AD3363.1	Understand the technologies for Handling Big Data and Hadoop Ecosystem.
AD3363.2	Understand various file managements tasks.
AD3363.3	Apply Hadoop, Map Reduce, HIVE, HBase to solve sample and real time problems.
AD3363.4	Perform Map-reduce analytics using Hadoop and to acquire clear understanding of programming tools.
AD3363.5	Understand the use of traditional database

CO- PO-PSO MATRIX

CO	PO												PSO			
	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4
AD3663.1	3	-	3	3	-	-	-	-	-	2	-	-	1	3	-	-
AD3663.2	3	2	-	3	-	-	-	-	2	-	2	-	2	3	-	-
AD3663.3	-	3	3	-	2	-	-	-	-	2	-	2	2	3	-	3
AD3663.4	-	-	3	2	2	-	-	-	-	2	-	-	3	3	-	2
AD3663.5	3	-	-	-	2	-	-	-	-	-	-	2	3	2	-	2
AD3663	**3.0**	**2.5**	**3.0**	**2.6**	**2.0**	**-**	**-**	**-**	**2.0**	**2.0**	**2.0**	**2.0**	**2.2**	**2.8**	**-**	**2.3**

EVALUATION PROCEDURE FOR EACH EXPERIMENT

S. No	Description	Mark
1.	Aim & Procedure	20
2.	Observation	30
3.	Conduction and Execution	30
4.	Output & Result	10
5.	Viva	10
Total		**100**

INTERNAL ASSESSMENT FOR LABORATORY

S. No	Description	Mark
1.	Conduction & Execution of Experiment	25
2.	Record	10
3.	Model Test	15
Total		**50**

Ex.No.1 Downloading and Installing Hadoop: Understanding different Hadoop modes, Startup Scripts, Configuration files.

Aim:

To Downloading and installing Hadoop; Understanding different Hadoop modes. Startup scripts, Configuration files.

Prerequisites to install Hadoop on Windows

- **VIRTUAL BOX (**For Linux): It is used for installing the operating system on it.
- **OPERATING SYSTEM:** You can install Hadoop on Windows or Linux based operating systems Ubuntu and CentOS are very commonly used.
- **JAVA:** You need to install the Java 8 package on your system.
- **HADOOP:** You require Hadoop latest version.

Steps:

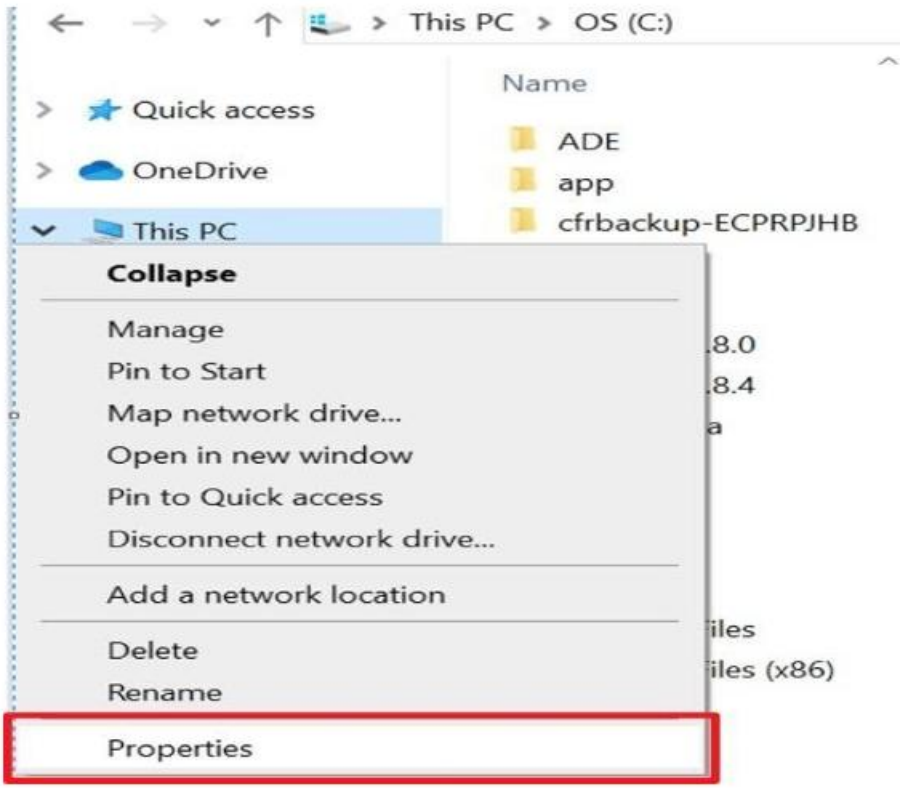
1. **Install Java**
   - Java JDK Link to download https://www.oracle.com/java/technologies/javase-jdk8-downloads.html
   - Extract and install Java in C:\Java
   - Open cmd and type -> javac –version

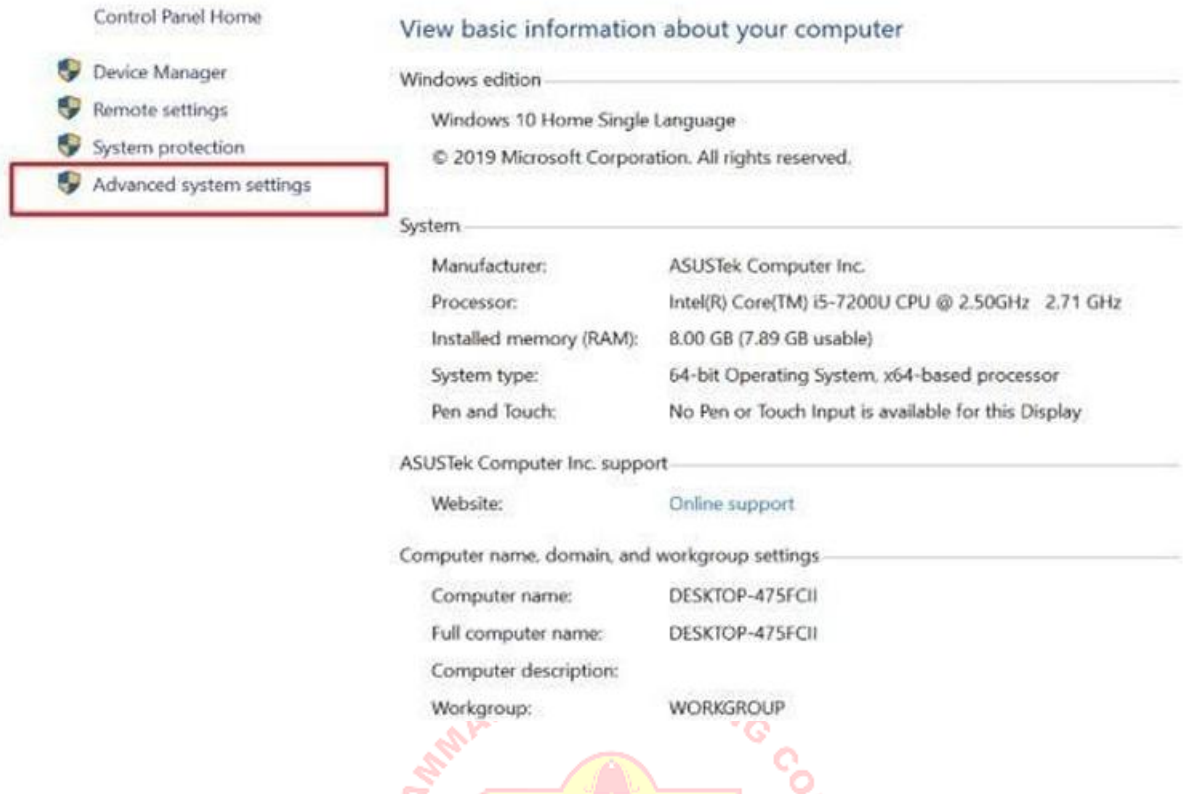
Command Prompt

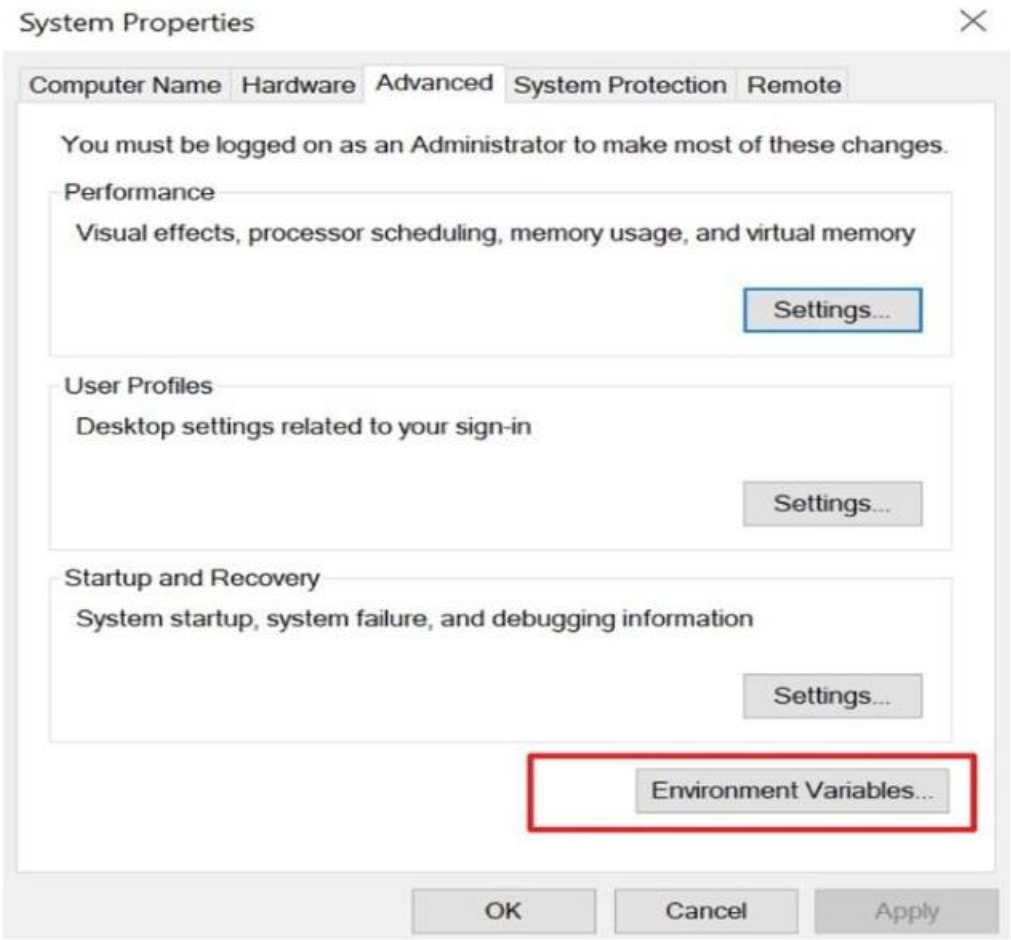
```
Microsoft Windows [Version 10.0.19041.572]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\asus>javac -version
javac 1.8.0_241
```

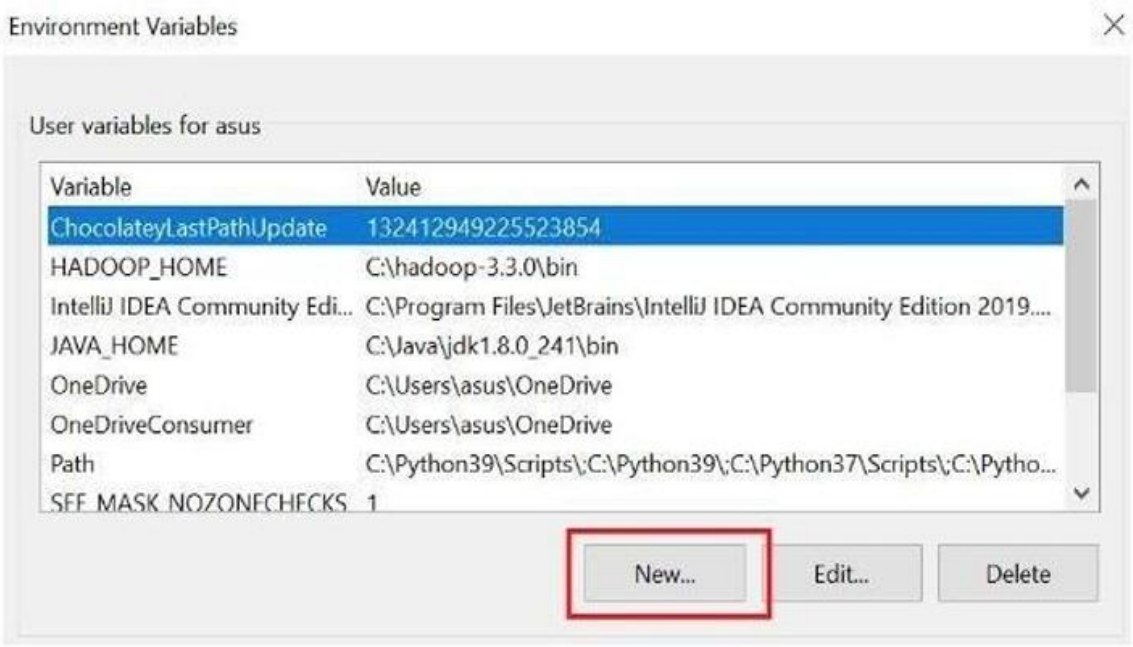
2. **Download Hadoop**
   - Download Hadoop using the link https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-3.3.0/hadoop-3.3.0.tar.gz
   - Extract to C:\Hadoop

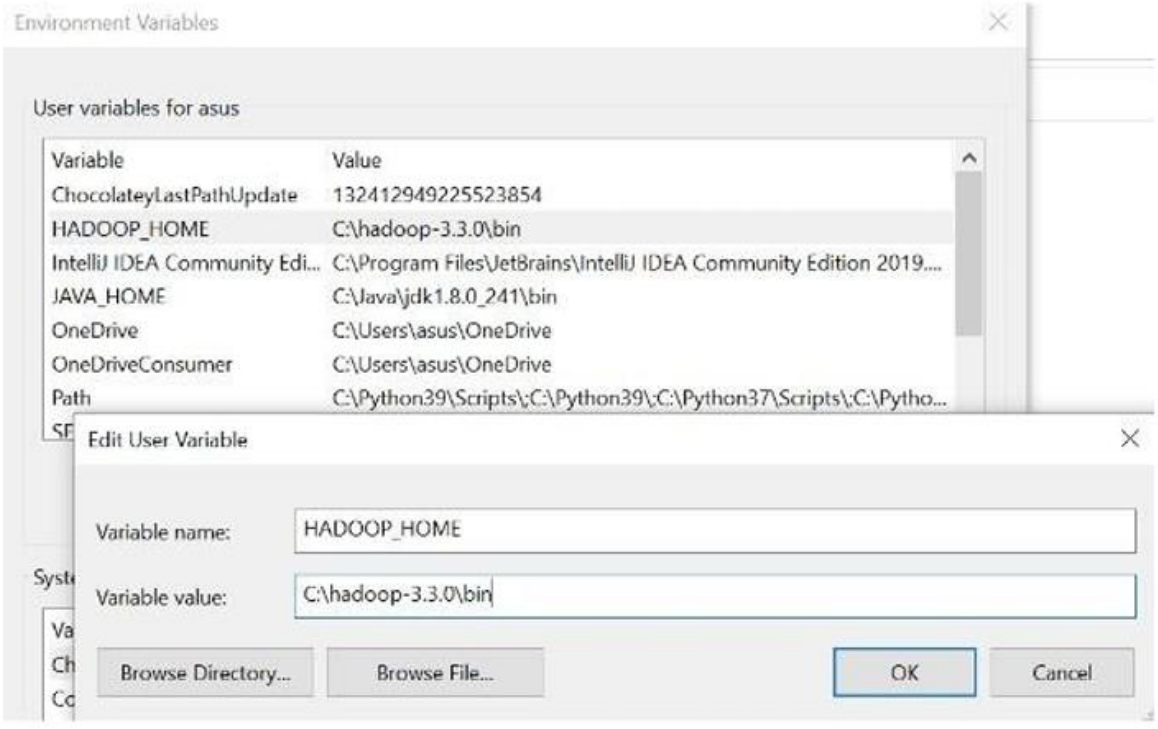
3. Set the path JAVA HOME Environment variable.
4. Set the path HADOOP HOME Environment variable.

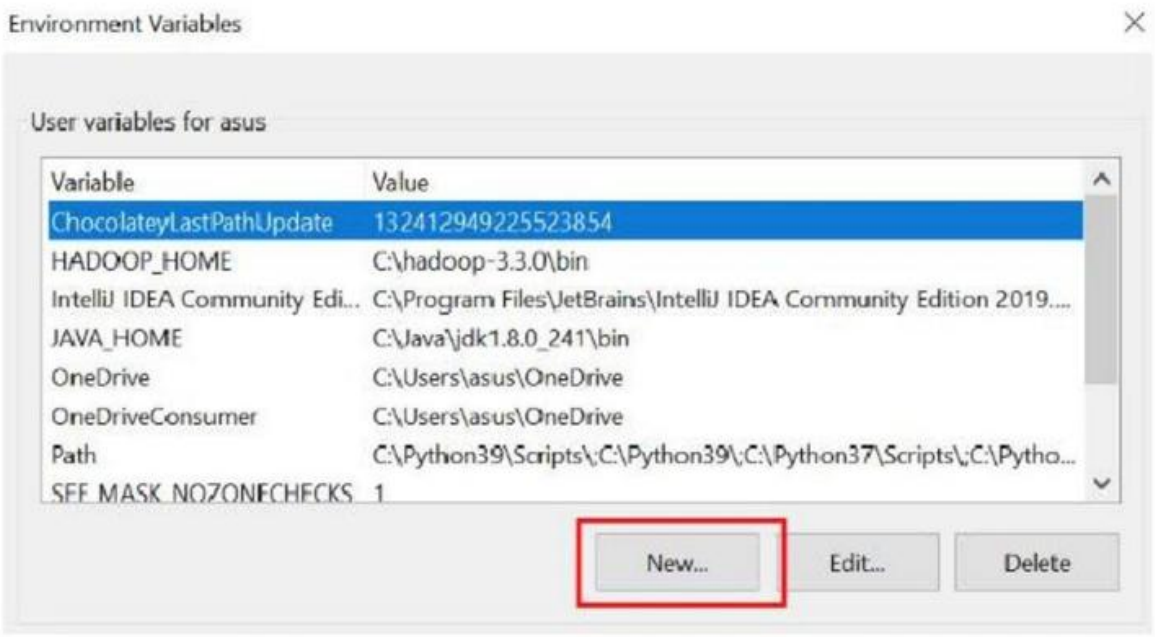


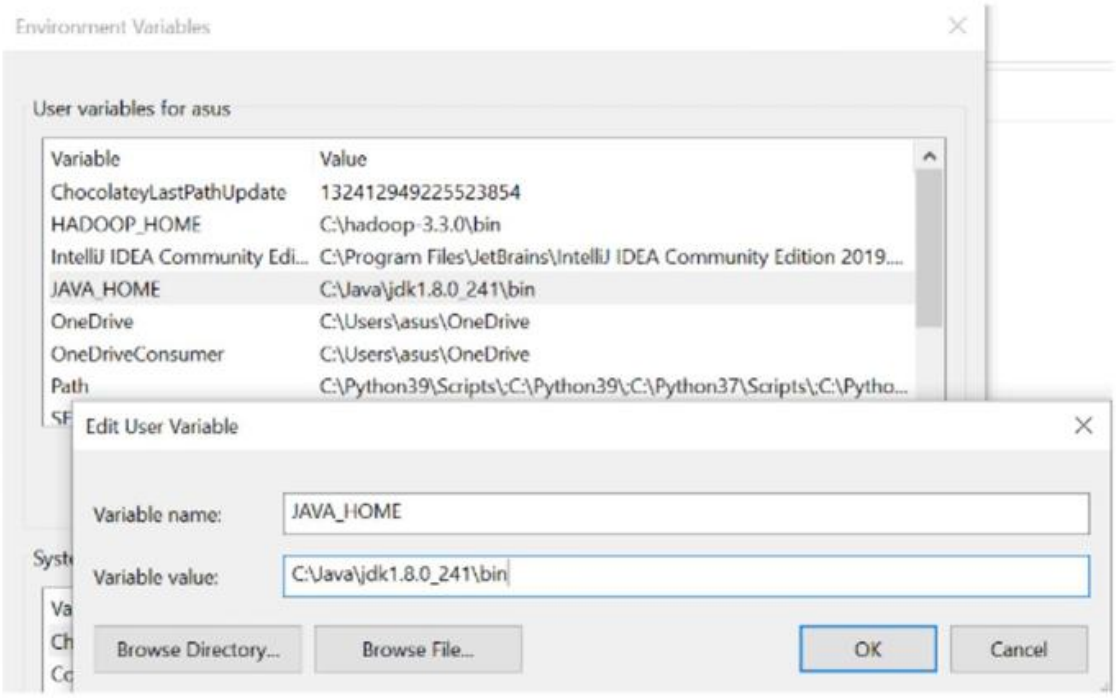

Control Panel Home
Device Manager
Remote settings
System protection
Advanced system settings
View basic information about your computer
Windows edition
Windows 10 Home Single Language
© 2019 Microsoft Corporation. All rights reserved.
System
Manufacturer: ASUSTek Computer Inc.
Processor: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz
Installed memory (RAM): 8.00 GB (7.89 GB usable)
System type: 64-bit Operating System, x64-based processor
Pen and Touch: No Pen or Touch Input is available for this Display
ASUSTek Computer Inc. support
Website: Online support
Computer name, domain, and workgroup settings
Computer name: DESKTOP-475FCII
Full computer name: DESKTOP-475FCII
Computer description:
Workgroup: WORKGROUP

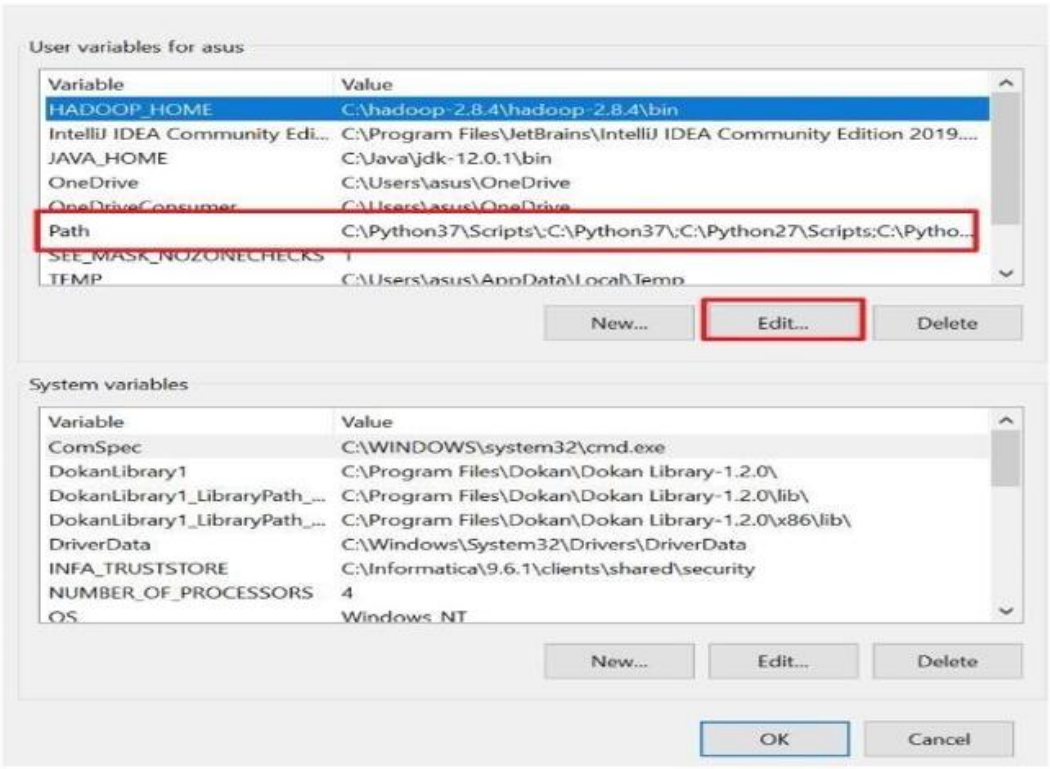

System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
You must be logged on as an Administrator to make most of these changes.
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your sign-in
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply

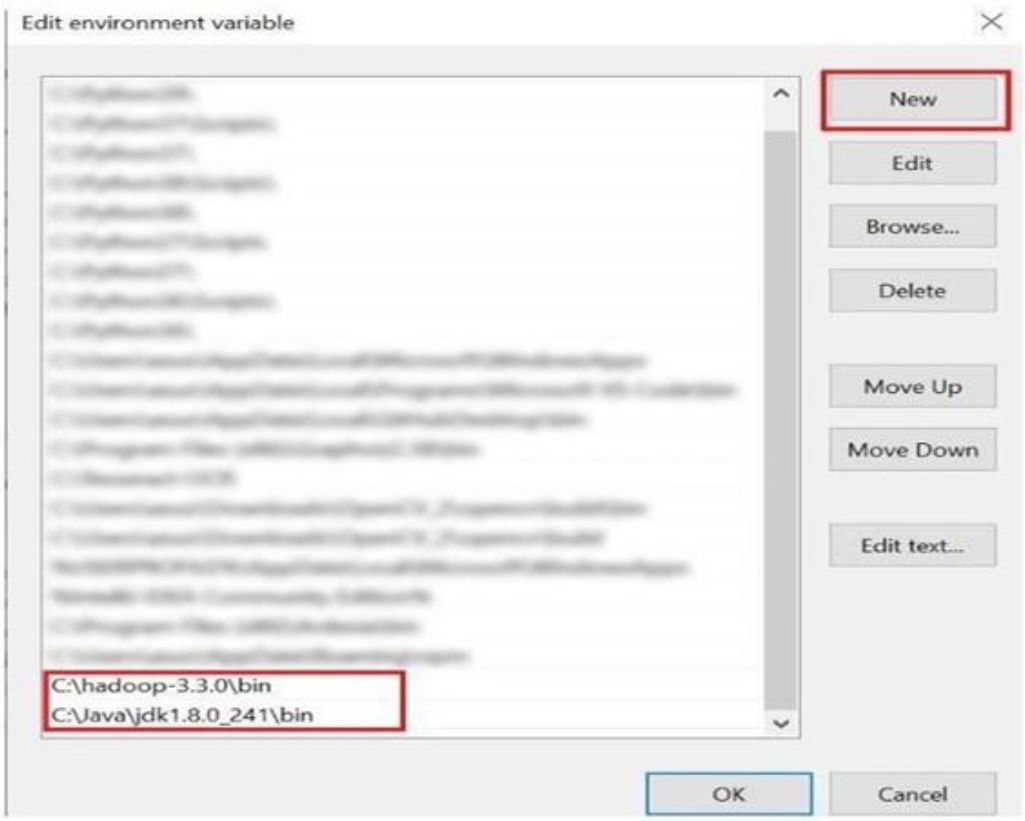

Environment Variables
User variables for asus
Variable
Value
ChocolateyLastPathUpdate
132412949225523854
HADOOP_HOME
C:\hadoop-3.3.0\bin
IntelliJ IDEA Community Edi...
C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2019....
JAVA_HOME
C:\Java\jdk1.8.0_241\bin
OneDrive
C:\Users\asus\OneDrive
OneDriveConsumer
C:\Users\asus\OneDrive
Path
C:\Python39\Scripts\;C:\Python39\;C:\Python37\Scripts\;C:\Pytho...
New...
Edit...
Delete


Environment Variables
User variables for asus
Variable
Value
ChocolateyLastPathUpdate
132412949225523854
HADOOP_HOME
C:\hadoop-3.3.0\bin
IntelliJ IDEA Community Edi...
C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2019....
JAVA_HOME
C:\Java\jdk1.8.0_241\bin
OneDrive
C:\Users\asus\OneDrive
OneDriveConsumer
C:\Users\asus\OneDrive
Path
C:\Python39\Scripts\;C:\Python39\;C:\Python37\Scripts\;C:\Pytho...
Edit User Variable
Variable name:
HADOOP_HOME
Variable value:
C:\hadoop-3.3.0\bin
Browse Directory...
Browse File...
OK
Cancel









5. Configurations

Edit file, **C:/Hadoop-3.3.0/etc/Hadoop/core-site.xml,**

Paste the xml code in folder and save

```
<configuration>
    <property>
      <name> fs. defaultFS</named>
      <value> hdfs://localhost:9000</value>
    </property>
</configuration>
```

Rename "mapred-site.xml.template" to "mapre-site.xml" and edit this file c:/Hadoop-3.3.0/etc/Hadoop/mapred-site.xml, paste xml code and save this file.

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

Create folder "data" under "C:\Hadoop-3.3.0"

Create folder "datanode" under "C:\Hadoop-3.3.0\data"

Create folder "namenode" under "C:\Hadoop-3.3.0\data"

Edit file **C:\Hadoop-3.3.0/etc/Hadoop/hdfs-site.xml**,

Paste xml code and save this file.

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</named>
    <value>/Hadoop-3.3.0/data/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</named>
    <value>/Hadoop-3.3.0/data/datanode</value>
  </property>
```

</configuration>

Edit file C:/Hadoop-3.3.0/etc/Hadoop/yarn-site.xml,

Paste xml code and save this file.

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</names>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
</configuration>
```

Edit the file, **C:/Hadoop-3.3.0/etc/Hadoop/Hadoop-env.cmd**

By closing the command line

"JAVA HOME=%JAVA HOME%" instead of set **"JAVA HOME=C:\JAVA"**

6. **Hadoop Configurations**

   Download

   https://github.com/brainmentorspvtltd/BigDataRDE/blob/master/Hadoop%20Configuration.Zip or (for Hadoop 3)

   https://github.com/s911415/apache-hadoop-3.1.0-winutil1s

   - Copy folder bin and replace existing bin folder in C:\Hadoop-3.3.0\bin
   - Format the NameNode
   - Open cmd and type command "hdfs namenode – format"

7. **Testing**

- Open md and change directory to **C:\Hadoop-3.3.0\sbin**
- Type start-all.cmd

(or you can start like this)

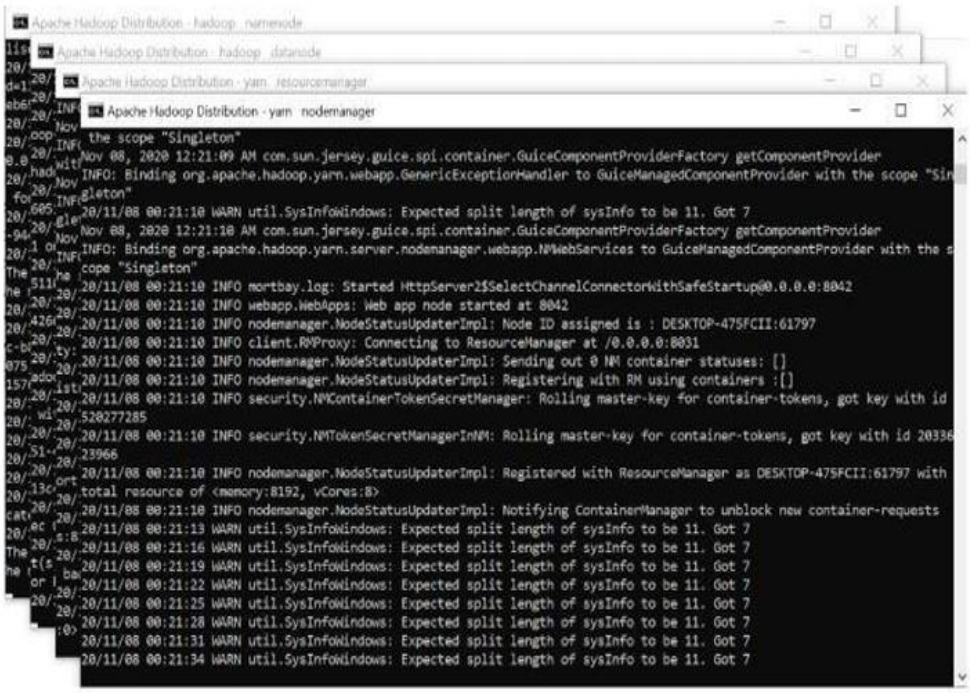
Start namenode and datanode with this command

- type start-dfs.cmd
- start yarn through its command
- type start-yarn.cmd

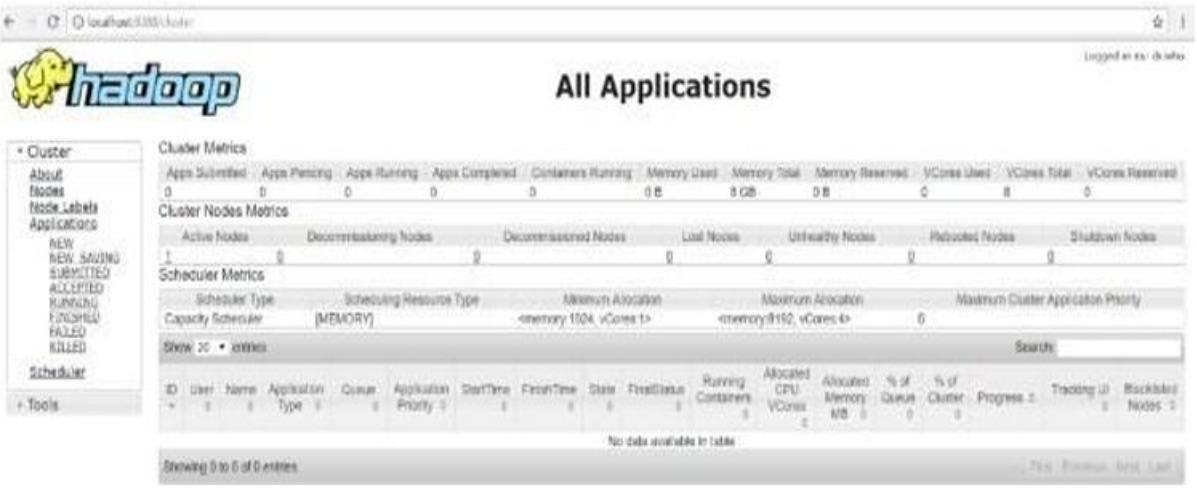
Make sure these apps are running

- Hadoop Namenode
- Hadoop datanode
- YARN Resource Manager

– YARN Node Manager



Open: **http://localhost:8088**



Open: **http://localhost:9870**

Result:

Downloaded and installed Hadoop and also understand different Hadoop modes. Startup scripts, Configuration files are successfully implemented.

Ex.No.2 Hadoop implementation of file management tasks, such as adding files and directories, retrieving files and deleting files.

Aim:

To implement the following file management tasks in Hadoop

1. Adding files and directories
2. Retrieving files
3. Deleting files

Steps:

1. Create a directory in HDFS at given paths.

   Usage:

   hadoop fs-mkdir <paths> Example:

   hadoop fs-mkdir/user/saurzcode/dirl/user/saurzcode/dir2

2. List the contents of a directory.

   Usage:

   hadoop fs – 1s <args>

   Example:

   hadoop fs – 1s/user/saurzcode

3. Upload and download a file in HDFS.

   Upload: Hadoop fs – put:

   Copy single src file, or multiple src files from local file system to the Hadoop data file system

   Usage:

   hadoop fs – put <local src> …. <HDFS dest Path> Exa,ple:

   Hadoop fs -put/home/saurzcode/Samplefile.txt/user/saurzcode/dir3/

   Download:

   hadoop fs-get:

   Copies/Downloads files to the local file system

   Usage:

   hadoop fs -get <hdfs src> <local dst> Example:

   hadoop fs-get/user/saurzcode/dir3/Samplefile.txt/home/

4. See contents of a file Same

   As unix cat command: Usage:

   hadoop fs -cat <path[filename]>

   Example:

   hadoop fs -cat/user/saurzcode/dirt/abc.txt

1. **Copy a file from source to destination**

   This command allows multiple sources as well in which casethe destination must be a directory.

   Usage:

   hadoop fs-cp <source> <dest>

   Example:

   hadoop fs-cp

   /user/saurzcode/dir1/abc.txt

   /user/saurzcode/dir2

2. **Copy a file from / to local file system to HDFS copy from local**

   Usage:

   hadoop fs -copyFromLocal <localsrc>

   URI Example:

   hadoop fs -copyFromLocal /home/saurzcode/abc.txt /user/ saurzcode/abc.txt

   Similar to put command, except that the source is restricted to a local file reference.

   copyToLocal

   Usage:

   hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <localdst>

   Similar to get command, except that the destination is restricted to a local file reference.

3. **Move file from source to destination**

   Note: Moving files across filesystem is not permitted.

   Usage:

   hadoop fs-mv <src> <dest>

   Example:

   hadoop fs-mv/user/saurzcode/dir1/abc.txt/user/saurzcode/dir2

4. **Remove a file or directory in HDFS**

   Remove files specified as argument. Deletes directory only when it is empty

   Usage:

   hadoop fs-rm <arg>

   Example:

   fs -rm/user/saurzcode/dir1/abc.txt

   Recursive version of delete

   Usage:

   hadoop fs-rmr <arg>

Example:

hadoop fs-rmr/user/saurzcode/

5. **Display last few lines of a file.**

   Similar to tail command in Unix.

   Usage:

   hadoop fs-tail <path[filename]

   Example:

   hadoop fs-tail/usr/saurzcode/dir1/abc.txt

6. **Display the aggregate length of a file.**

   Usage:

   hadoop fs-du <path>

   Example:

   hadoop fs-du/user/saurzcode/dirt/abc.txt

Result:

Thus, the Hadoop implementation of file management tasks, such as adding files and directories, retrieving files and deleting files is executed successfully.

Ex.No.3 Implement of Matrix Multiplications with Hadoop Map Reduce.

Aim:

To write a Map Reduce Program that implements Matrix Multiplication.

Algorithm:

We assume that the input matrices are already stored in Hadoop Distributed File System (HDFS) in a suitable format (e.g., CSV, TSV) where each row represents a matrix element. The matrices are compatible for multiplication (the number of columns in the first matrix is equal to the number of rows in the second matrix).

Steps:

Step 1: Mapper

The mapper will take the input matrices and emit key-value pairs for each element in the result matrix. The key will be the (row, column) index of the result element, and the value will be the corresponding element value.

Step2: Reducer

The reducer will take the key-value pairs emitted by the mapper and calculate the partial sum for each element in the result matrix.

Step 3: Main Driver

The main driver class sets up the Hadoop job configuration and specifies the input and output paths for the matrices.

Step 4: Running the Job

To run the MapReduce job, you need to package your classes into a JAR file and then submit it to Hadoop using the hadoop jar command. Make sure to replace input ath and output ath with the actual HDFS paths to your input matrices and desired output directory.

Program:

```
import java.io.IOException;
import java.uti1.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
```

```
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.fs.Path;
public class MatrixMultiplicationMapper extends Mapper <LongWritable, Text, Text, Text>
Protected void map (LongWritable key, Text value, Context context) throws IOException, IntemiptedException(
// Parse the input line to get row, column, and value of each element in the input matrices
String[] elements = value.toString().split(",");
int row = Integer.parseInt(elements[0]);
int col = Integer.parseInt(elements[1]);
int va1 = Integer.parseInt(elements[2]);
// Emit key-value pairs where key is (row, column) index of the result element
// and value is the corresponding element value
context.write(new Text(row + "," + col), new Text(val));
context.write(key, new IntWritable(Sum));

public class MatrixMultiplicationReducer extends Reducer <Text, Text, Text, IntWritable>
 {
protected void reduce(Text key, Iterab1e values, Context context) throws IOException, IntemiptedException ( int result = 0; for (Text value : values)
{
// Accumulate the partial sum for the result element
result += Integer.parseInt(value.tostring());

// Emit the final result for the result element
context.write(key, new IntWritab1e(result));

public class MatrixMultiplicationDriver (

public static void main (String [] args) throws Exception
(Configuration conf= new Configuration ();
Job job = Job.getInstance(conf, "Matrix Multiplication");
job.setJarByClass(MatrixMu1tip1icationDriver.class);
job.setMapperClass(MatrixMu1tip1icationMapper.class);
job.setReducerClass(MatrixMu1tip1icationReducer.class);
```

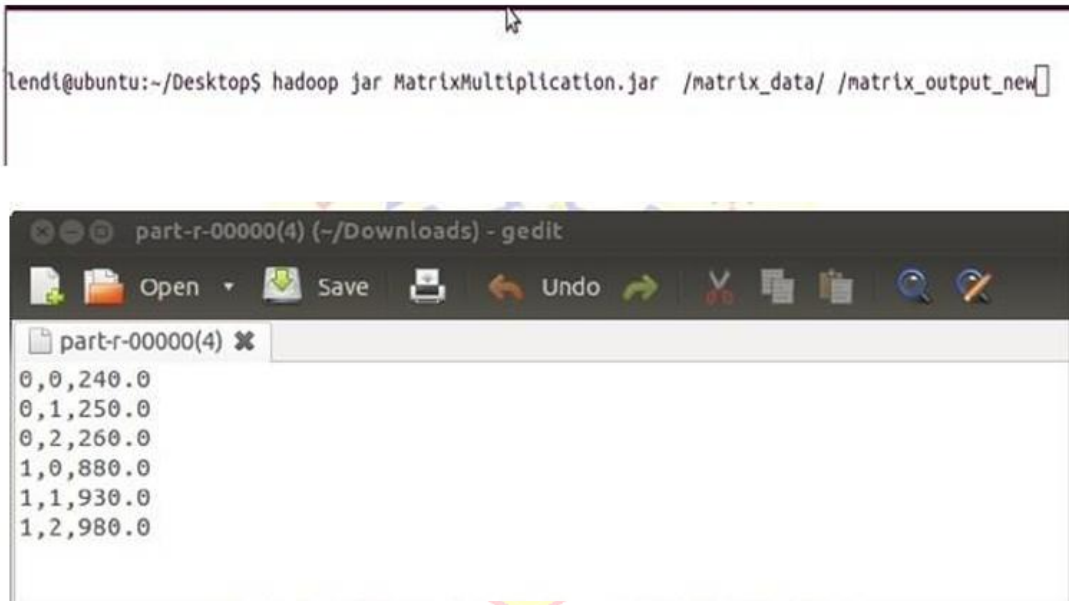
```
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job. waitForComp1etion(true)? 0: 1);
```

Run the MapReduce code:

The command for running a MapReduce code is:

hadoop jar matrixmultip1ication.jar MatrixMultiplicationDriver input ath output ath

Output:



Result:

Thus, the Map Reduce program that implements Matrix Multiplication was executed and verified successfully.

Ex.No.4 Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm.

Aim:

To write a basic word count program to understand Map Reduce Paradigm.

Algorithm:

The entire MapReduce program can be fundamentally divided into three parts:

• Mapper Phase Code

• Reducer Phase Code

• Driver Code

Steps:

STEP 1: MAPPER CODE:

We have created a class Map that extends the class Mapper which is already defined in the MapReduce Framework.

- We define the data types of input and output key/value pair after the class declaration using angle brackets.
- Both the input and output of the Mapper is a key/value pair.

Input:

- The key is nothing but the offset of each line in the text file: LongWritable
- The value is each individual line : Text

Output:

- The key is the tokenized words: Text
- We have the hardcoded value in our case which is 1: IntWritable
- Example — Dear 1, Bear 1, etc.

We have written a java code where we have tokenized each word and assigned them a hardcoded value equal to 1.

STEP 2: REDUCER CODE:

• We have created a class Reduce which extends class Reducer like that of Mapper.

• We define the data types of input and output key/value pair after the class declaration using angle brackets as done for Mapper.

• Both the input and the output of the Reducer is a key value pair.

Input:

- The key nothing but those unique words which have been generated after the sorting and shuffling phase: Text
- The value is a list of integers corresponding to each key: IntWritable
- Example — Bear, [1, 1], etc.

Output:

- The key is all the unique words present in the input text file: Text
- The value is the number of occurrences of each of the unique words: IntWritable
- Example — Bear, 2; Car, 3, etc.
- We have aggregated the values present in each of the list corresponding to each key and produced the final answer.
- In general, a single reducer is created for each of the unique words, but, you can specify the number of reducer in mapred-site.xml.

STEP 3: DRIVER CODE:

- In the driver class, we set the configuration of our MapReduce job to run in Hadoop.
- We specify the name of the job, the data type of input/ output of the mapper and reducer.
- We also specify the names of the mapper and reducer classes.
- The path of the input and output folder is also specified.
- The method setInputFormatClass () is used for specifying that how a Mapper will read the input data or what will be the unit of work. Here, we have chosen TextInputFormat so that single line is read by the mapper at a time from the input text file. The main () method is the entry point for the driver. In this method, we instantiate a new Configuration object for the job.

Program:

```
importjava.io.IOException;
importjava.uti1.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
```

```
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.fs.Path;
public class WordCount
public static class Map extends Mapper <LongWritable, Text, Text, IntWritable>
{
public void map (LongWritab1e key, Text value,Context context)
throws IOException,IntemiptedException(
String line = value.toString();
StringTokenizer tokenizer = new StringTokenizer(line);
while (tokenizer.hasMoreTokens())
( value.set(tokenizer.nextToken());
context.write(value, new IntWritable(1));

public static class Reduce extends Reducer <Text, IntWritable, Text, IntWritable>
(
public void reduce(Text key, Iterab1e<IntWritable> va1ues, Context context)
throws IOException,IntemiptedException (
int sum=0;
for(IntWritab1e x: values)
sum+=x.get();
context.write(key, new IntWritab1e(sum));
public static void main (String [] args) throws Exception
(Configuration conf= new Configuration ();
Job job = new Job(conf,"My Word Count Program");
job.setJarByClass(WordCount.class);
job.setMapperClass(Map.class);
job.setReducerClass(Reduce.class);
job.setOutputKeyClass(Text.class);
 job.setOutputValueClass(IntWritab1e.class);
job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);
Path outputPath = new Path(args[1]);
//Configuring the input/output path from the filesystem into the job
```

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

//deleting the output path automatically from hdfs so that we don't have to delete it explicitly

outputPath.getFi1eSystem(conf). delete(outputPath);

//exiting the job only if the flag value becomes false

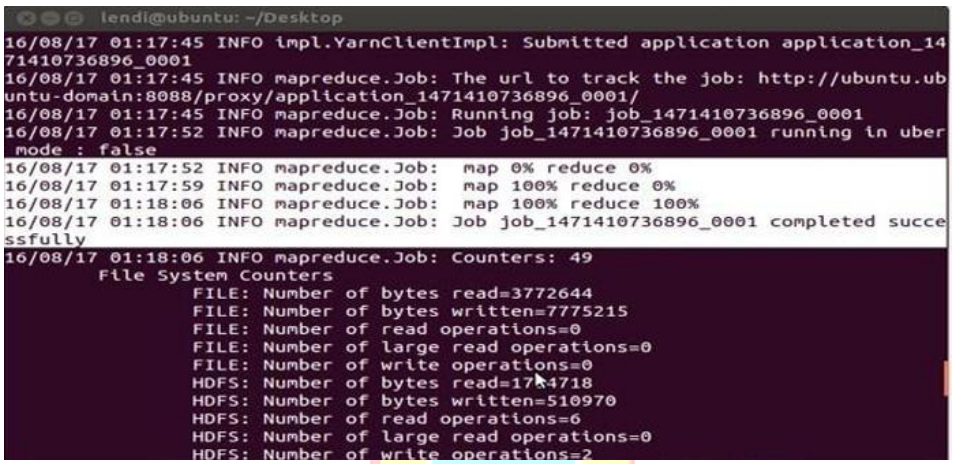
System.exit(job.waitForComp1etion(true) ? 0 : 1);

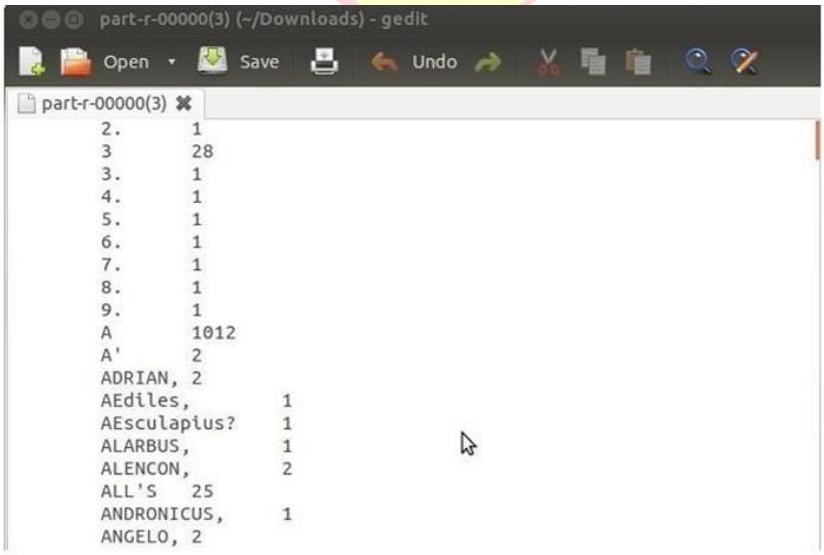
Run the MapReduce code:

The command for running a MapReduce code is:

hadoop jar hadoop-mapreduce-example.jar WordCount /sample/input /sample/output

Output:





Result

Thus, the Map Reduce Program that implements word count was executed and verified successfully.

Ex.No.5 Installation of Hive along with practice examples

Aim:

To install HIVE along with practice examples.

Prerequisites:

- Java Development Kit (JDK) installed and the JAVA HOME environment variable set.
- Hadoop installed and configured on your Windows system.

Step -by-step installation:

1. **Download HIVE:**

   Visit the Apache Hive website and download the latest stable version of Hive.

   Official Apache Hive website: https://hive.apache.org/

2. Extract the **Downloaded** Hive Archive to a Directory on Your Windows Machine, e.g., C:\hive.
3. Configure **Hive:**
   - Open the Hive configuration file (hive-site xml) located in the conf folder of the extracted Hive directory.
   - Set the necessary configurations, such as Hive Metastore connection settings and Hadoop configurations. Make sure to adjust paths accordingly for Windows. Here's an example of some configurations:

```
<configuration>
 <property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:derby:;databaseName=/path/to/metastore  db;create=true
  </value>
  <description>JDBC connect string for a JDBC
  metastore.</description>
```

4. **Environment Variables Setup:**
   - Add the Hive binary directory (C:\hive\bin in this example) to your PATH environment variable.
   - Set the HIVE HOME environment variable to point to the Hive installation directory (C:\hive in this example).
5. Start the Hive Metastore service:

   To start the Hive Metastore service, you can use the schematool script.

```bash
schematool -initSchema -dbType derby
```

6. Start **Hive:**
    - Open a command prompt or terminal and navigate to the Hive installation directory.
    - Execute the hive command to start the Hive shell.

Examples:

1. **Create a Database:**

   To create a new database in HIVE, use the following syntax:

   CREATE DATABASE database name;

   Example:

   CREATE DATABASE mydatabase;

2. **Use a Database**:

   To use a specific database in HIVE, use the following syntax:

   USE database name;

   Example:

   USE mydatabase;

3. **Show Databases:**

   To display a list of available databases in HIVE, use the following syntax:

   SHOW DATABASES;

4. **Create a Table:**

   To create a table in HIVE, use the following syntax:

   CREATE TABLE table name (columnl datatype, column2 datatype)

   Example:

   CREATE TABLE mytable (id INT, name STRING, age INT)

5. **Show Tables:**

   To display a list of tables in the current database, use the following syntax:

   SHOW TABLES;

6. **Describe a Table:**

   To view the schema and details of a specific table, use the following syntax:

   DESCRIBE table name;

   Example:

   DESCRIBE mytable;

7. **Insert Data into a Table:**

To insert data into a table in HIVE, use the following syntax:

INSERT INTO table name (columnl, column2, ...) VALUES (valuel, value2, ...);

Example:

INSERT INTO mytable (id, name, age) VALUES (1, 'John Doe', 25);

8. **Select Data from a Table:**

SELECT columnl, column2, ... FROM table name WHERE condition;

Example:

SELECT * FROM mytable WHERE age > 20;

Result:

Thus, the installation of HIVE was done successfully.

Ex.No.6 Installation of HBase, Installing thrift along with Practice examples

Aim:

To install HBASE using Virtual Machine and perform some operations in HBASE.

Algorithm:

Step 1: Install a Virtual Machine

- Download and install a virtual machine software such as VirtualBox (https://www.virtualbox.org/) or VMware (https://www.vmware.com/).
- Create a new virtual machine and install a Unix-based operating system like Ubuntu or CentOS. You can download the ISO image of your desired Linux distribution from their official websites.

Step 2: Set up the Virtual Machine

- Launch the virtual machine and install the Unix-based operating system following the installation wizard.
- Make sure the virtual machine has network connectivity to download software packages.

Step 3: Install Java

- Open the terminal or command line in the virtual machine.
- Update the package list sudo apt update
- Install OpenJDK (Java Development Kit):

  sudo apt install default-jdk
- Verify the Java installation: java -version

Step 4: Download and Install HBase

- In the virtual machine, navigate to the directory where you want to install HBase.
- Download the HBase binary distribution from the Apache HBase website (https://hbase.apache.org/). Look for the latest stable version.
- Extract the downloaded archive

  tar -xvf .tar.gz
- Replace <hbase archive name> with the actual name of the HBase archive file.
- Move the extracted HBase directory to a desired location:

  sudo mv <hbase_extracted_directory> /opt/hbase
- Replace **<hbase_extracted_directory>** with the actual name of the extracted HBase directory.

Step 5: Configure HBase

- Open the HBase configuration file for editing:

sudo nano /opt/hbase/conf/hbase-site.xml

- Add the following properties to the configuration file:

```
<configuration>
 <property>
  <name>hbase.rootdir</name>
  <va1ue>file:///var/lib/hbase</value>
 </property>
 <property>
  <name>hbase.zookeeper.property.dataDir</name>
  <value>/var/lib/zookeeper</va1ue>
 </property>
</configuration>
```

- Save the file and exit the text editor.

Step 6: Start HBase

- Start the HBase server:

  sudo /opt/hbase/bin/start-hbase.sh

HBASE PRACTICE EXAMPLES:

Step 1: Start HBase

- Make sure HBase is installed and running on your Windows system.

Step 2: Open HBase Shell

- Open a command prompt or terminal window and navigate to the directory where the HBase installation is located. Run the following command to start the HBase shell:

  >>hbase shell

Step 3: Create a Table

- In the HBase shell, you can create a table with column families.
- For example, let's create a table named "my_table" with a column family called "cf':

  >> create 'my_table', 'cP

Step 4: Insert Data

- To insert data into the table, you can use the put command.
- Here's an example of inserting a row with a specific row key and values:

  >> put 'my_table', 'rowl', 'cf:column1', 'valuel'

  >> put 'my_table', 'rowl', 'cf:column2', 'value2'

Step 5: Get Data

- You can retrieve data from the table using the get command.
- For example, to get the values of a specific row:

>> get 'my_table', 'rowl'

- This will display all the column family values for the specified row.

Step 6: Scan Data

- To scan and retrieve multiple rows or the entire table, use the scan command.
- For instance, to scan all rows in the table:

  >> scan 'my_table'

- This will display all rows and their corresponding column family values.

Step 7: Delete Data

- To delete a specific row or a particular cell value, you can use the delete command.
- Here's an example of deleting a specific row:

  >>delete 'my_table', 'rowl'

Step 8: Disable and Drop Table

- If you want to remove the table entirely, you need to disable and drop it.
- Use the following commands:

  >>disable 'my_table' >>drop 'my_table'

RESULT:

Thus, the installation of HBase using Virtual Machine was done successfully.

Installation of Thrift

Aim:

To install Apache thrift on Windows OS.

Algorithm:

Step 1: Download Apache Thrift:

- Visit the Apache Thrift website: https://thrift.apache.org/
- Go to the "Downloads" section and find the latest version of Thrift.
- Download the Windows binary distribution (ZIP file) for the desired version.

Step 2: Extract the ZIP file:

- Locate the downloaded ZIP file and extract its contents to a directory of your choice.
- This directory will be referred to as <THRIFT DIRT> in the following steps.

Step 3: Set up environment variables:

- Open the Start menu and search for "Environment Variables" and select "Edit the system environment variables."
- Click the "Environment Variables" button at the bottom right of the "System Properties" window.
- Under the "System variables" section, find the "Path" variable and click "Edit."
- Add the following entries to the "Variable value" field (replace with the actual directory path):

  <THRIFT_DIR>\bin

  <THRIFT_DIR>\lib

- Click "OK" to save the changes.

Step 4: Verify the installation: •

- Open a new Command Prompt window.
- Run the following command to verify that Thrift is installed and accessible:

  thrift —version

- If everything is set up correctly, you should see the version number of Thrift printed on the screen.

Result:

Thus, the installation of Thrift on windows OS was done Successfully

Ex.No.7 Practice importing and exporting data from various databases

Aim:

To import and export data from various Databases using SQOOP.

Algorithm:

Step 1: Install SQOOP.

- First, you need to install Sqoop on your Hadoop cluster or machine.
- Download the latest version of Sqoop from the Apache Sqoop website (http://sqoop.apache.org/) and follow the installation instructions provided in the documentation

Step 2: Importing data from a database:

- To import data from a database into Hadoop, use the following Sqoop command:

```
Sqoop import —connect
jdbc:<DB TYPE>://<DB HOST>:<DB PORT>/<DB NAME> \
        --username <DB USERNAME> \
        --password <DB PASSWORD> \
        --table <TABLE NAME> \
        --target-dir <HDFS TARGET DIR> \
        --m <NUMBER OF MAP TASKS>
```

- Replace the placeholders
- (<DB TYPE>, <DB HOST>, <DB PORT>, <DB NAME>, <DB USERNAME>, <DB PASSWORD>, <TABLE NAME>, <HDFS TARGET DIR>, and <NUMBER OF MAP TASKS>) with the appropriate values for your database and Hadoop environment.

Step 3: Exporting data to a database:

To export data from Hadoop to a database, use the following Sqoop command:

```
                    e
jdbc:USB TYPEIT 'DB HOST>:<DB PORT>/<DB NAME> \
        --username <DB USERNAME> \
```

```
--password <DB PASSWORD> \
--table <TABLE NAME> \
--export-dir <HDFS EXPORT DIR> \
--input-fields-terminated-by '<DELIMITER>'
```

- Replace the placeholders
- (<DB TYPE>, <DB HOST>, <DB PORT>, <DB NAME>, <DB USERNAME>, <DB_PASSWORD>, <TABLE_NAME>, <HDFS_EXPORT_DIR>, and <DELIMITER>) with the appropriate values for your database and Hadoop environment.

Result:

Thus, the implementation export data from various Databases using SQOOP was done successfully.

Topic Beyond Syllabus

MapReduce program to find the grades of student's

Aim:

To Develop a MapReduce program to find the grades of students.

Program:

```
import java.util.Scanner;
public class JavaExample
{
public static void main(String args[])
{ /* This program assumes that the student has 6 subjects,
 * thats why I have created the array of size 6. You can
* change this as per the requirement. */
int marks[] = new int[6];
int i; float total=0,
 avg; Scanner scanner = new Scanner(System.in);
for(i=0; i<6; i++)
{
System.out.print("Enter Marks of Subject"+(i+1)+":");
marks[i] = scanner.nextInt();
total = total + marks[i];
}
scanner.close();
//Calculating average
here avg = total/6;
System.out.print("The student Grade is: ");
if(avg>=80)
{
System.out.print("A");
}
 else if(avg>=60 && avg<80)
{
System.out.print("B");
}
 else if(avg>=40 && avg<60)
{
```

```
System.out.print("C");
}
else
{
System.out.print("D");
}
}
}
```

Output:

```
Enter Marks of Subject1:40
Enter Marks of Subject2:80
Enter Marks of Subject3:80
Enter Marks of Subject4:40
Enter Marks of Subject5:60
Enter Marks of Subject6:60
The student Grade is: B
```


Result:

Thus, the Map Reduce Program that implements grades of students was executed and verified successfully.

VALLIAMMAI ENGINEERING COLLEGE
SRM

****