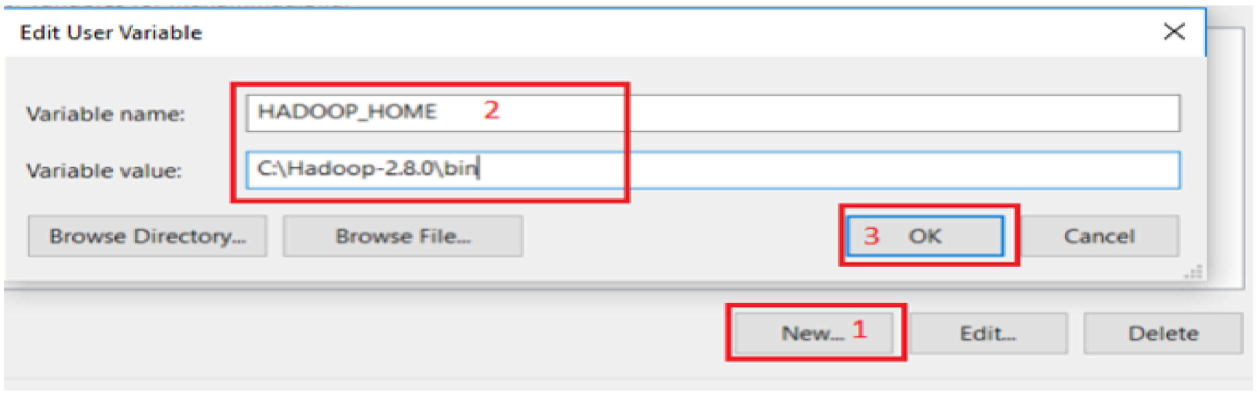
Question 12: To download and install Hadoop framework with necessary Start-up scripts?

Procedure / Algo:

- Step 1: Download these files
    - Hadoop 2.8.0
    - Java JDK 1.8.0.zip
- Step 2: Install the Java JDK using the downloaded zip file
    - use “javac -version” to verify java installation.
- Step 3: Install the Hadoop Software using the downloaded exe file
- Step 4: Set the Path Variables
    - HADOOP_HOME for Hadoop Software
    - JAVA_HOME in for Java
    - Set the Hadoop bin, sbin & JAVA bin path in the environment variable.

Program:

- Set the Hadoop Path

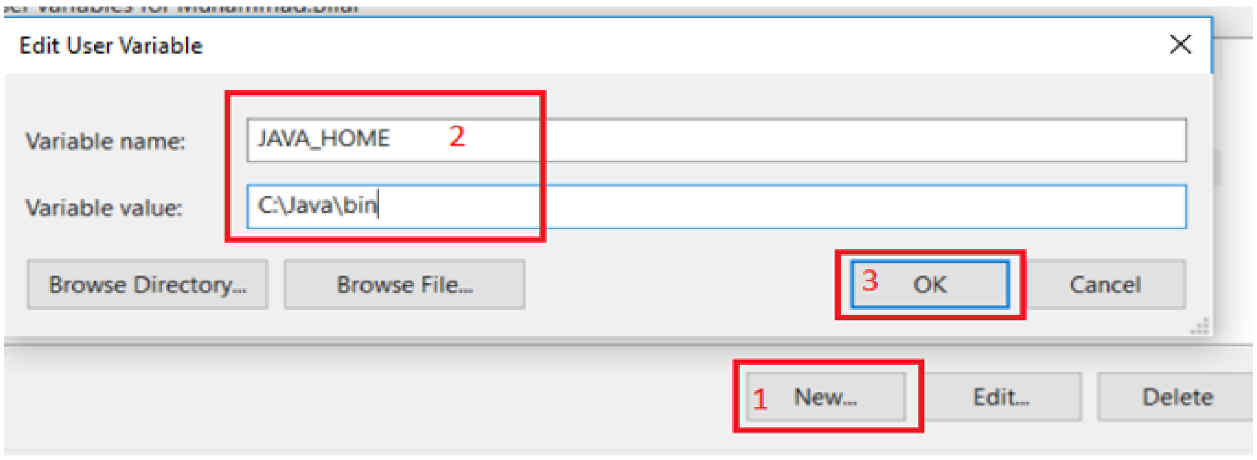


Edit User Variable

Variable name: HADOOP_HOME

Variable value: C:\Hadoop-2.8.0\bin

- Set the Java Path

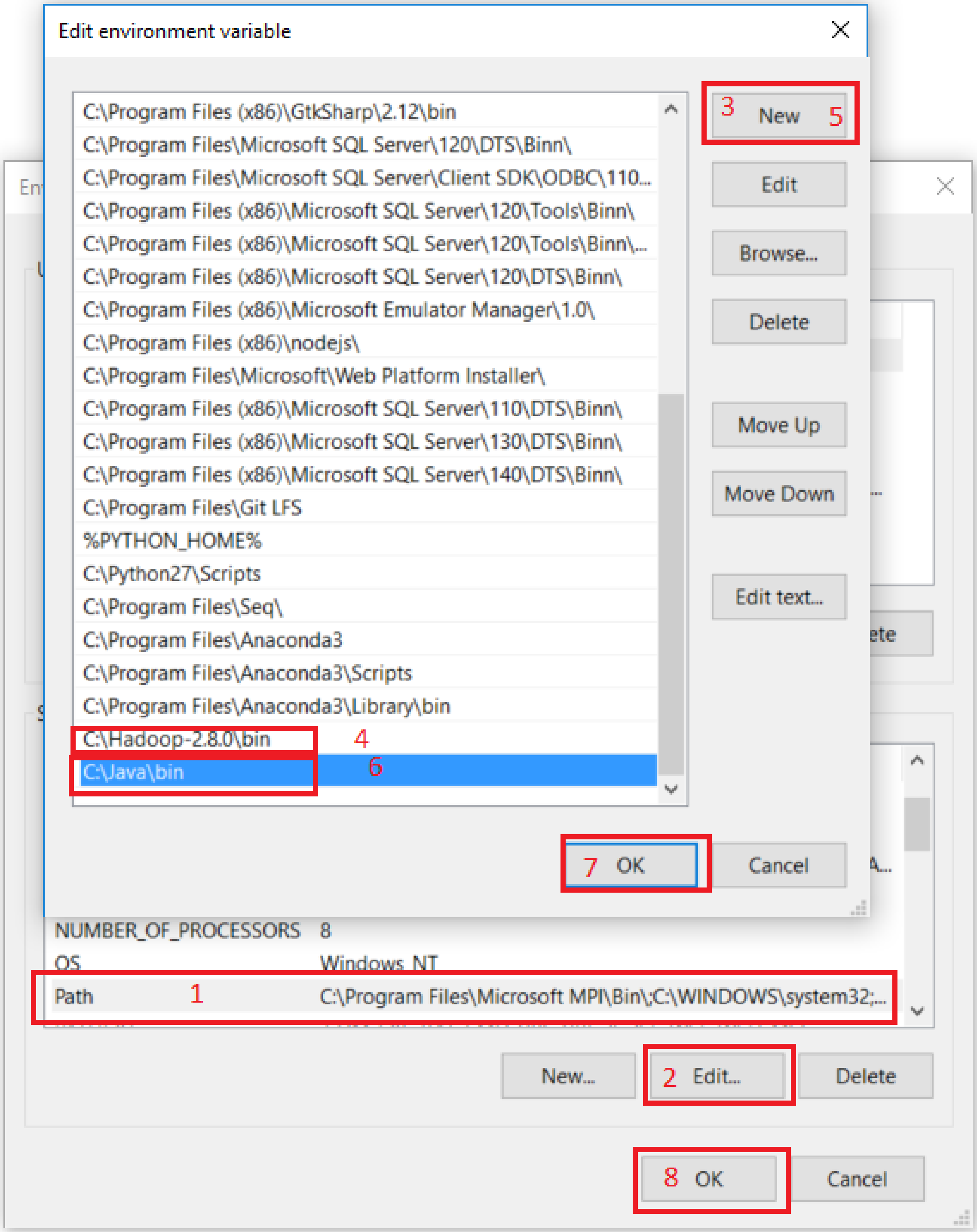


Edit User Variable

Variable name: JAVA_HOME

Variable value: C:\Java\bin

- Set the Hadoop bin, sbin & JAVA bin path in the environment variable.



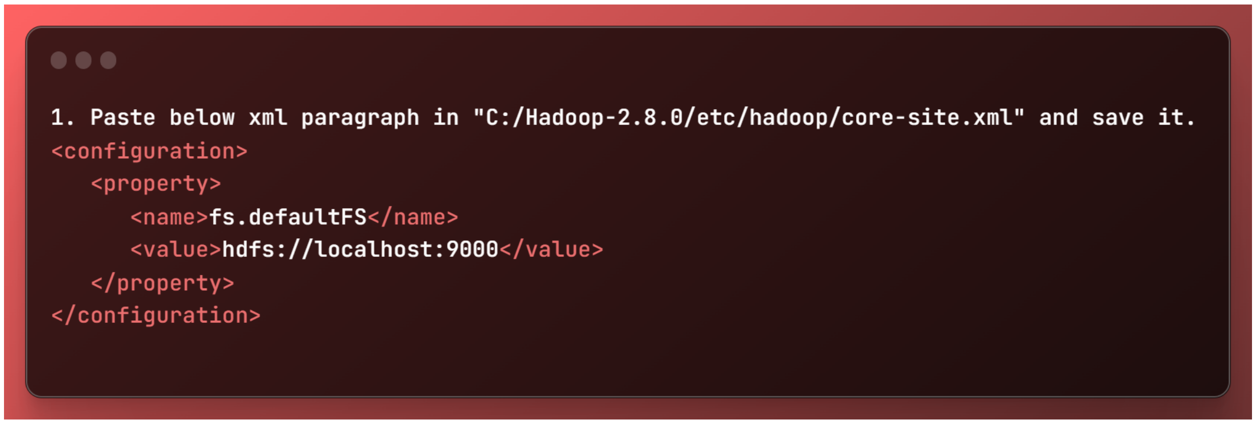
Edit environment variable

C:\Program Files (x86)\GtkSharp\2.12\bin
C:\Program Files\Microsoft SQL Server\120\DTS\Binn\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\110...
C:\Program Files (x86)\Microsoft SQL Server\120\Tools\Binn\
C:\Program Files (x86)\Microsoft SQL Server\120\Tools\Binn\...
C:\Program Files (x86)\Microsoft SQL Server\120\DTS\Binn\
C:\Program Files (x86)\Microsoft Emulator Manager\1.0\
C:\Program Files (x86)\nodejs\
C:\Program Files\Microsoft\Web Platform Installer\
C:\Program Files (x86)\Microsoft SQL Server\110\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\130\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\140\DTS\Binn\
C:\Program Files\Git LFS
%PYTHON_HOME%
C:\Python27\Scripts
C:\Program Files\Seq\
C:\Program Files\Anaconda3
C:\Program Files\Anaconda3\Scripts
C:\Program Files\Anaconda3\Library\bin
C:\Hadoop-2.8.0\bin
C:\Java\bin

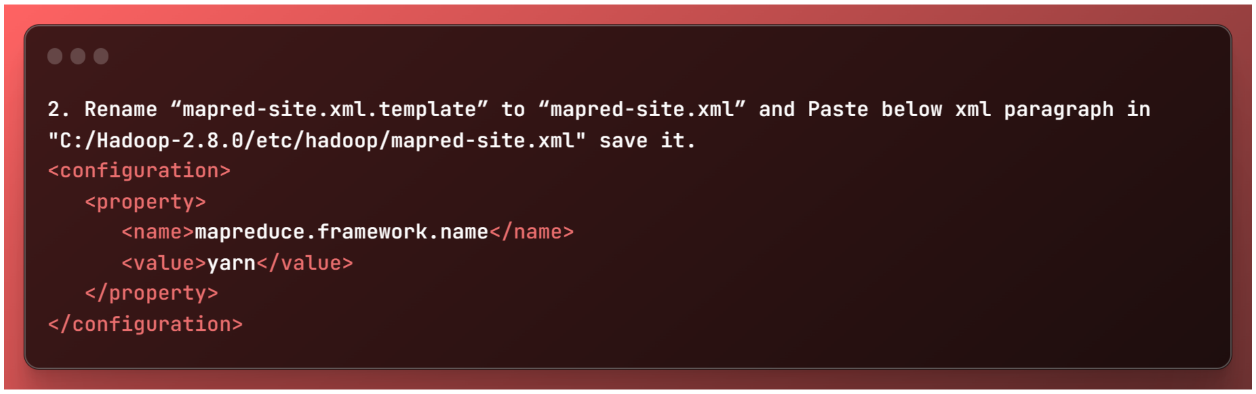
NUMBER_OF_PROCESSORS	8
OS	Windows_NT
Path	C:\Program Files\Microsoft MPI\Bin\;C:\WINDOWS\system32;...

- Create folder “data” under “C:\Hadoop-2.8.0”
    - Create folder “datanode” & “namenode” under “C:\Hadoop-2.8.0\data”

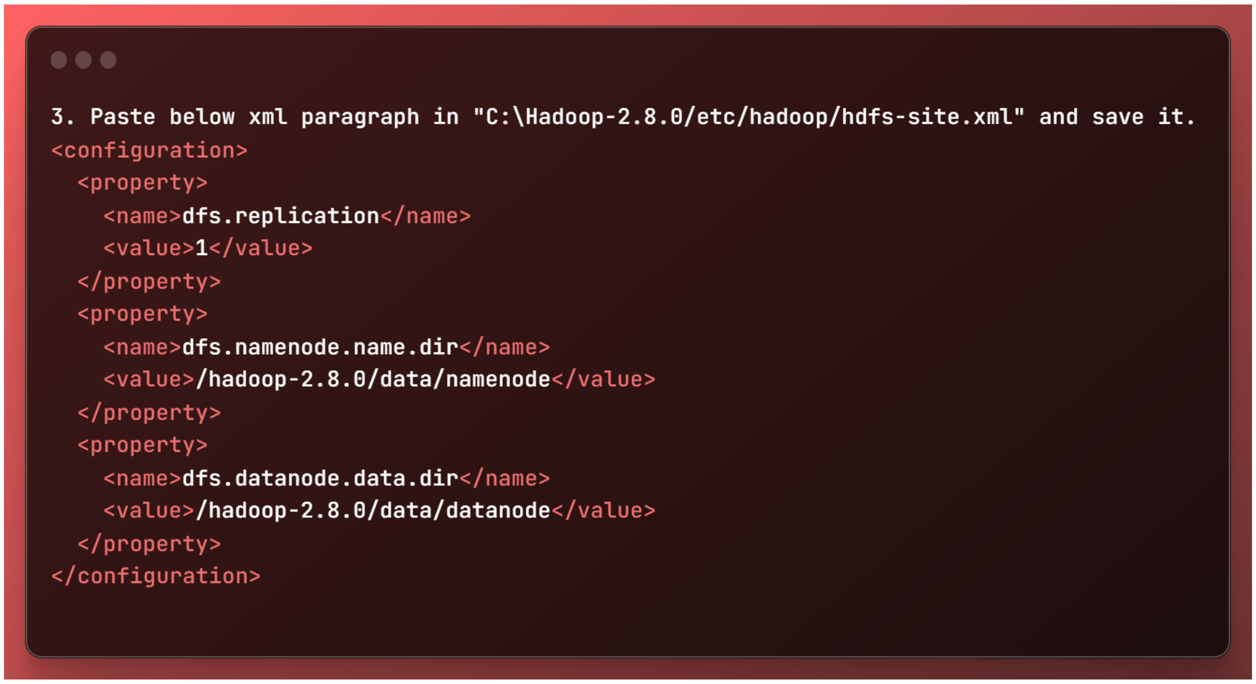
- Configuration:



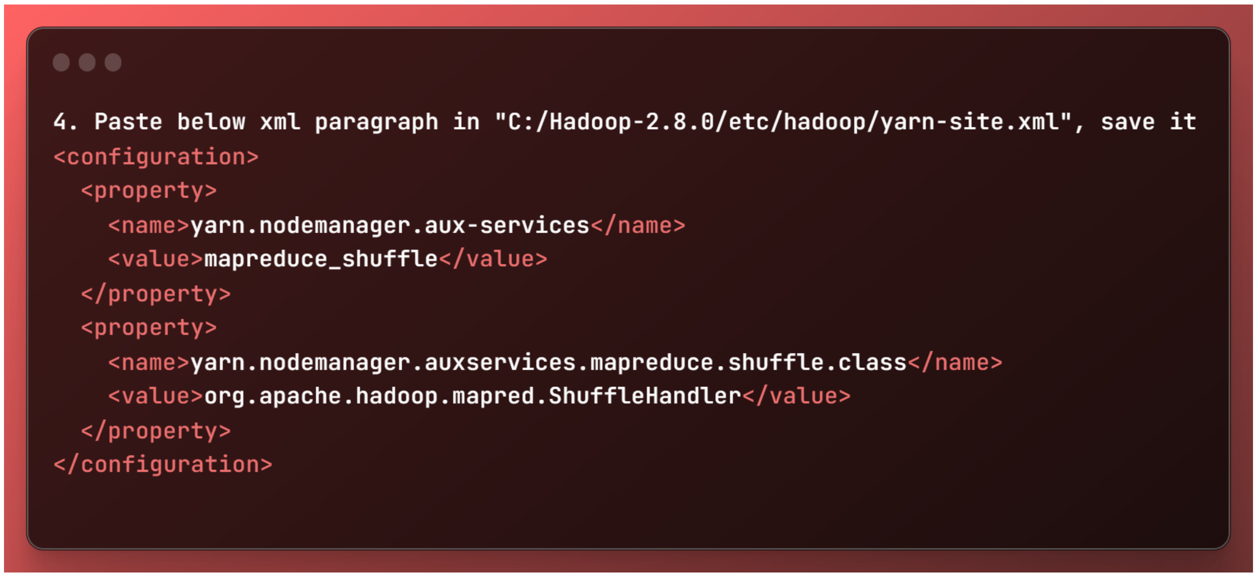
```
1. Paste below xml paragraph in "C:/Hadoop-2.8.0/etc/hadoop/core-site.xml" and save it.
<configuration>
   <property>
      <name>fs.defaultFS</name>
      <value>hdfs://localhost:9000</value>
   </property>
</configuration>
```



```
2. Rename “mapred-site.xml.template” to “mapred-site.xml” and Paste below xml paragraph in
"C:/Hadoop-2.8.0/etc/hadoop/mapred-site.xml" save it.
<configuration>
   <property>
      <name>mapreduce.framework.name</name>
      <value>yarn</value>
   </property>
</configuration>
```

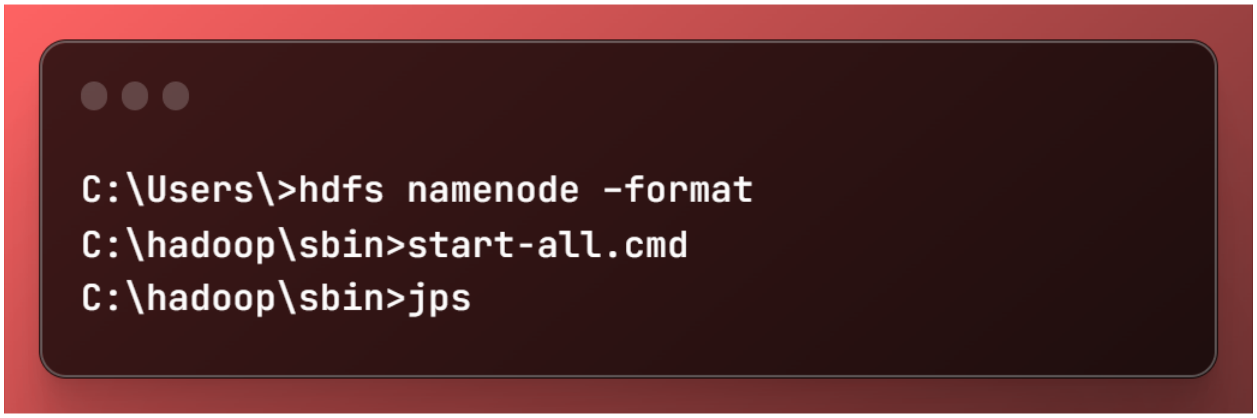


```
3. Paste below xml paragraph in "C:\Hadoop-2.8.0/etc/hadoop/hdfs-site.xml" and save it.
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/hadoop-2.8.0/data/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/hadoop-2.8.0/data/datanode</value>
  </property>
</configuration>
```



```
4. Paste below xml paragraph in "C:/Hadoop-2.8.0/etc/hadoop/yarn-site.xml", save it
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
</configuration>
```

- Open command prompt and execute the following commands



```
C:\Users\>hdfs namenode -format
C:\hadoop\sbin>start-all.cmd
C:\hadoop\sbin>jps
```

Output:

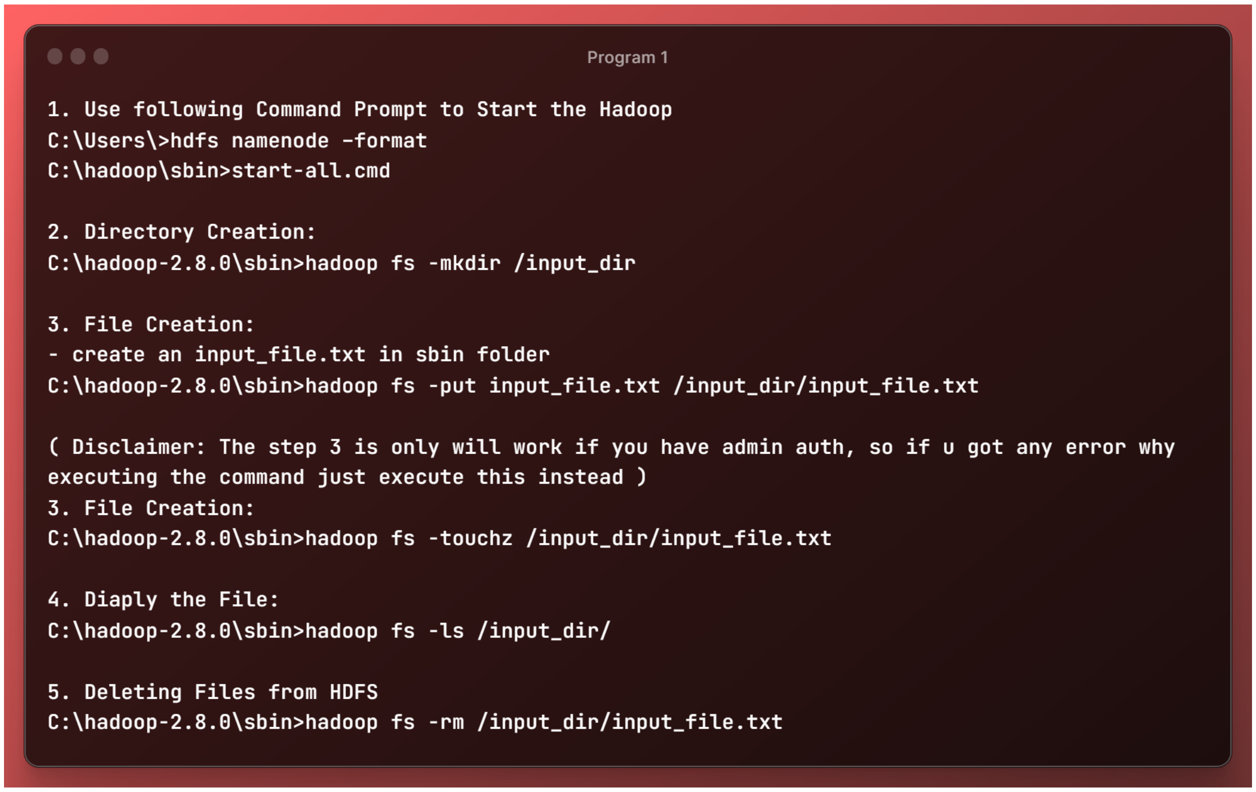
- Open: http://localhost:8088
- Open: http://localhost:50070

Question 3: Write a Hadoop command to implement file management tasks in the HDFS environment ? &
Question 10: Implement a Hadoop command to implement file management tasks, such as Adding files and directories to the HDFS environment ? &
Question 16: Implement a Hadoop command to implement file management tasks, such as Adding files to HDFS environment ? &
Question 20: Implement a Hadoop command to implement file management tasks, such as deleting files from the HDFS environment ?

Procedure / Algo:

- Just write it on your own
- By referring your output screen and program

Program:



```
1. Use following Command Prompt to Start the Hadoop
C:\Users\>hdfs namenode -format
C:\hadoop\sbin>start-all.cmd

2. Directory Creation:
C:\hadoop-2.8.0\sbin>hadoop fs -mkdir /input_dir

3. File Creation:
- create an input_file.txt in sbin folder
C:\hadoop-2.8.0\sbin>hadoop fs -put input_file.txt /input_dir/input_file.txt

( Disclaimer: The step 3 is only will work if you have admin auth, so if u got any error why
executing the command just execute this instead )
3. File Creation:
C:\hadoop-2.8.0\sbin>hadoop fs -touchz /input_dir/input_file.txt

4. Diaply the File:
C:\hadoop-2.8.0\sbin>hadoop fs -ls /input_dir/

5. Deleting Files from HDFS
C:\hadoop-2.8.0\sbin>hadoop fs -rm /input_dir/input_file.txt
```

Output:

- In Output section write the resultant text from all the executed commands from the command prompt
- And open localhost:50070/explorer.html and draw the simple layout

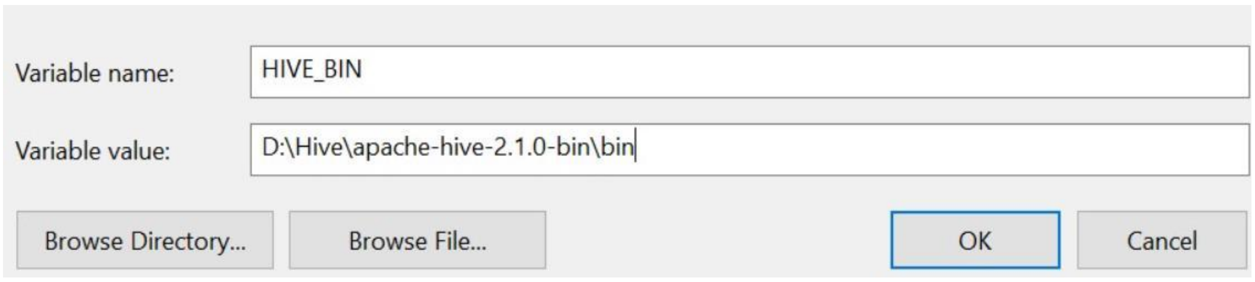
Question 9: To download and install HIVE with necessary Start-up script.

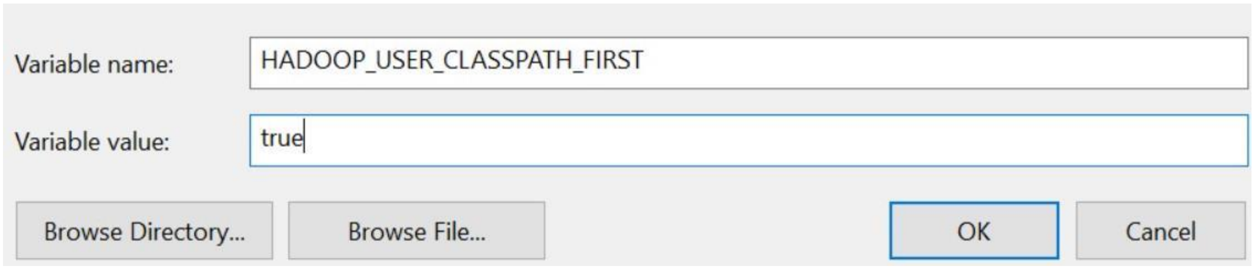
Procedure / Algo / Prepare:

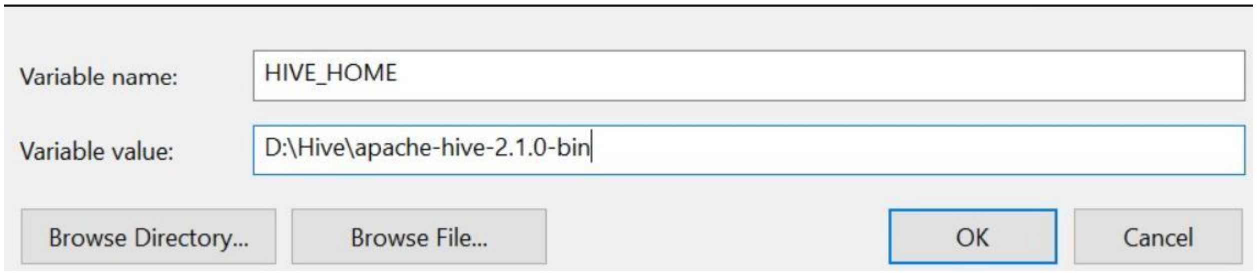
- Download Hadoop 2.8.0
- Download Java JDK
- Download Hive 2.1.0
- Download Derby Metastore
- Download hive-site.xml

Program:

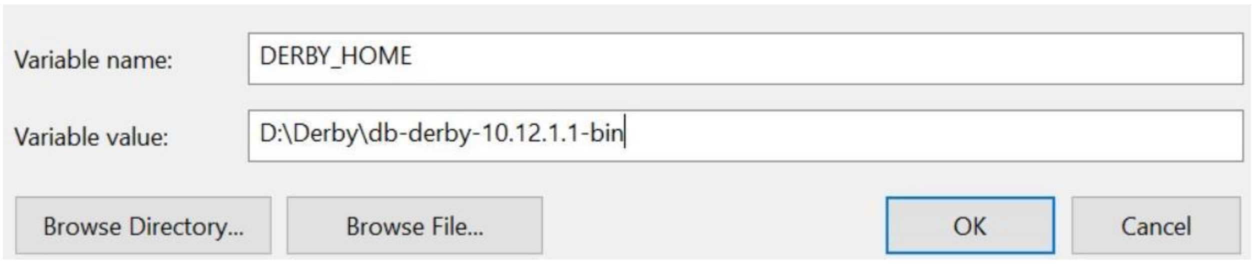
- Step 1: Download and Extract the Hive file
    - Extract that file and place under "D:\Hive"
- Step 2: Extract the Derby file
    - Extract that file and place under "D:\Derby"
- Step 3: Configuring hive-site.xml file
    - copy the “hive-site.xml” file to hive configuration location "D:\Hive\apachehive-2.1.0-bin\conf".
- Step 4: Configuring Derby libraries
    - copy all libraries from derby - "D:\Derby\db-derby-10.12.1.1-bin\lib" to hive location "D:\Hive\apache-hive2.1.0-bin\lib"
- Step 5: Add Environment variables
    - Config the following paths in Environment variables
        - HIVE_HOME as “D:\Hive\apache-hive-2.1.0-bin”
        - HIVE_BIN as “D:\Hive\apache-hive-2.1.0-bin\bin”
        - HIVE_LIB as “D:\Hive\apache-hive-2.1.0-bin\lib”
        - DERBY_HOME as “D:\Derby\db-derby-10.12.1.1-bin”
        - HADOOP_USER_CLASSPATH_FIRST as true


Variable name: HIVE_BIN
Variable value: D:\Hive\apache-hive-2.1.0-bin\bin


Variable name: HADOOP_USER_CLASSPATH_FIRST
Variable value: true

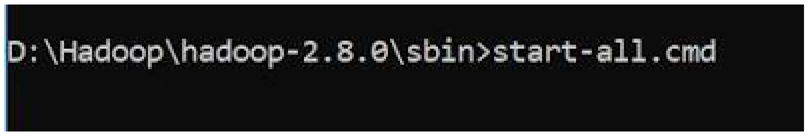

Variable name: HIVE_HOME

Variable value: D:\Hive\apache-hive-2.1.0-bin


Variable name: DERBY_HOME

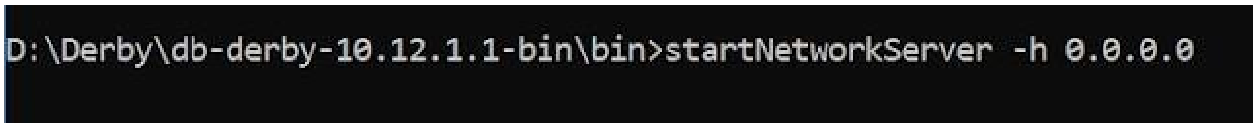
Variable value: D:\Derby\db-derby-10.12.1.1-bin

- Step 6: Start the Hadoop
    - Open command prompt and change directory to "D:\Hadoop\hadoop2.8.0\sbin" and execute
        - start-all.cmd



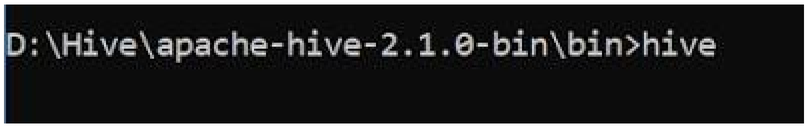
```
D:\Hadoop\hadoop-2.8.0\sbin>start-all.cmd
```

- Step 7: Start Derby server
    - After successful execution of Hadoop, Open command prompt and change directory to "D:\Derby\db-derby-10.12.1.1-bin\bin" and execute
        - startNetworkServer -h 0.0.0.0



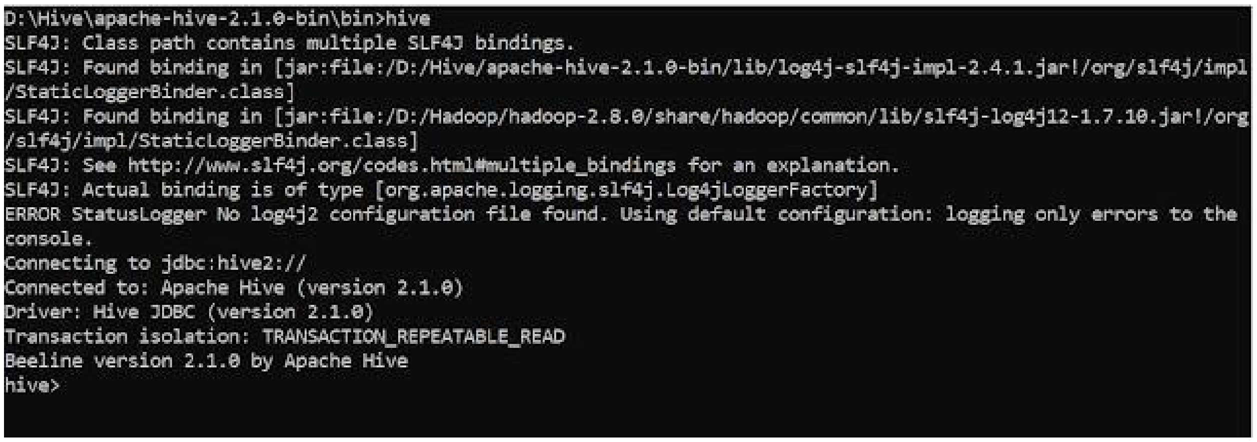
```
D:\Derby\db-derby-10.12.1.1-bin\bin>startNetworkServer -h 0.0.0.0
```

- Step 8: Start the Hive
    - After successful execution of Derby Server, Open command prompt and change directory to "D:\Hive\apache-hive-2.1.0-bin\bin" and execute
        - hive



```
D:\Hive\apache-hive-2.1.0-bin\bin>hive
```

Output:



```
D:\Hive\apache-hive-2.1.0-bin\bin>hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/D:/Hive/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl-2.4.1.jar!/org/slf4j/impl
/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/D:/Hadoop/hadoop-2.8.0/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org
/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
ERROR StatusLogger No log4j2 configuration file found. Using default configuration: logging only errors to the
console.
Connecting to jdbc:hive2://
Connected to: Apache Hive (version 2.1.0)
Driver: Hive JDBC (version 2.1.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 2.1.0 by Apache Hive
hive>
```

- Just write “hive>“ as the output of this program

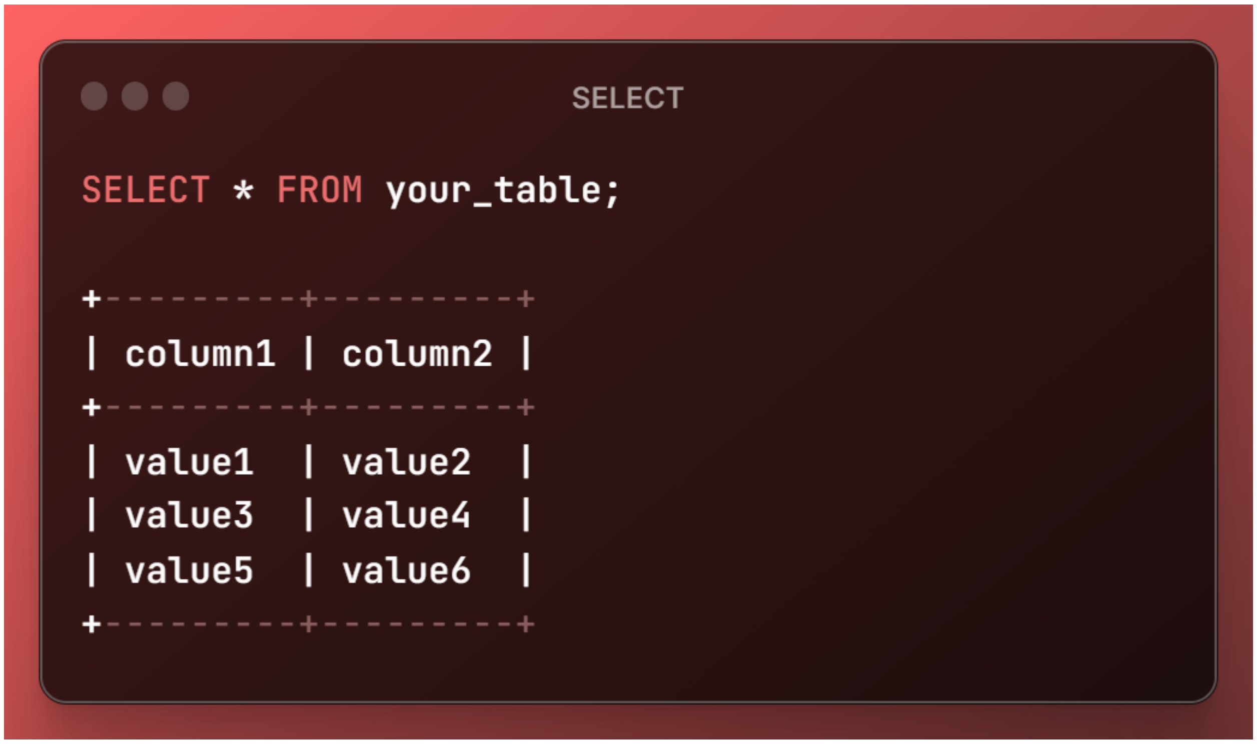
Question 8: Implement all HIVE DML commands.

Algo:

- SELECT: Retrieves data from one or more tables
- INSERT INTO: Adds data into a table from a specified source
- UPDATE: Modifies existing data in a table
- DELETE: Removes data from a table based on specified conditions
- LOAD DATA: Loads data into a table from an external file
- INSERT OVERWRITE: Writes data into a table, replacing existing data.

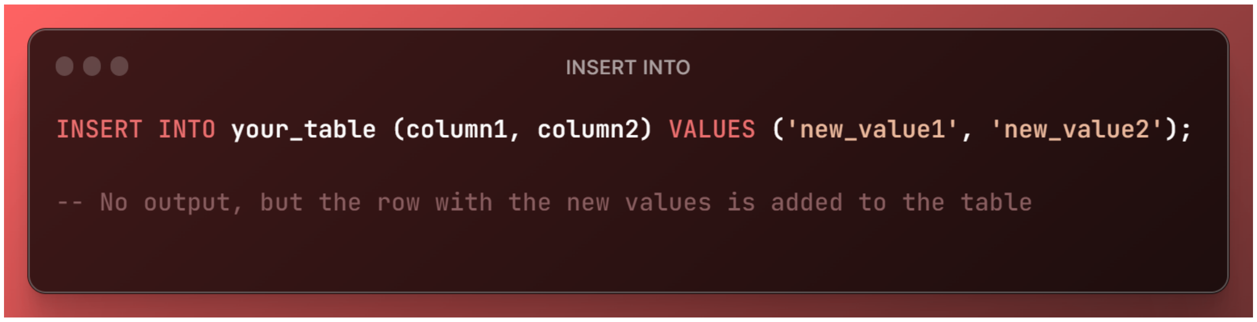
****Before Executing these commands on hive terminal, first start it by the above last 3 steps on command prompts****

Program & Output:



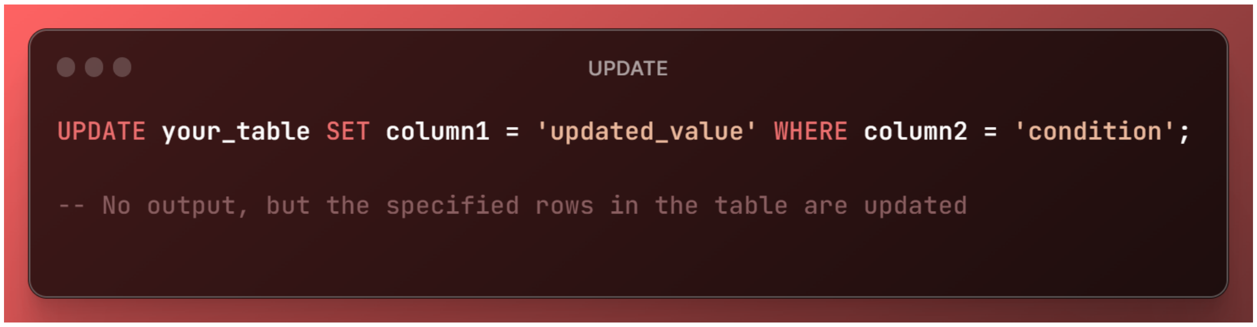
```
SELECT * FROM your_table;

+---------+---------+
| column1 | column2 |
+---------+---------+
| value1  | value2  |
| value3  | value4  |
| value5  | value6  |
+---------+---------+
```



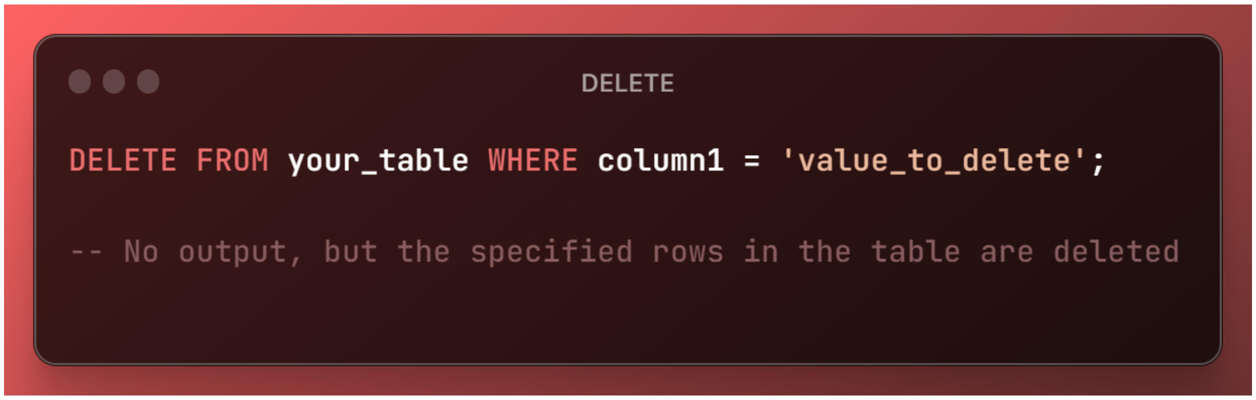
```
INSERT INTO your_table (column1, column2) VALUES ('new_value1', 'new_value2');

-- No output, but the row with the new values is added to the table
```



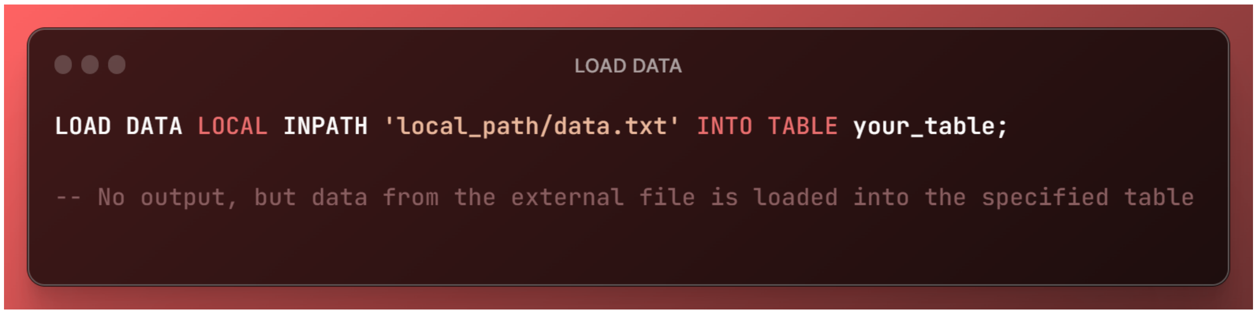
```
UPDATE your_table SET column1 = 'updated_value' WHERE column2 = 'condition';

-- No output, but the specified rows in the table are updated
```



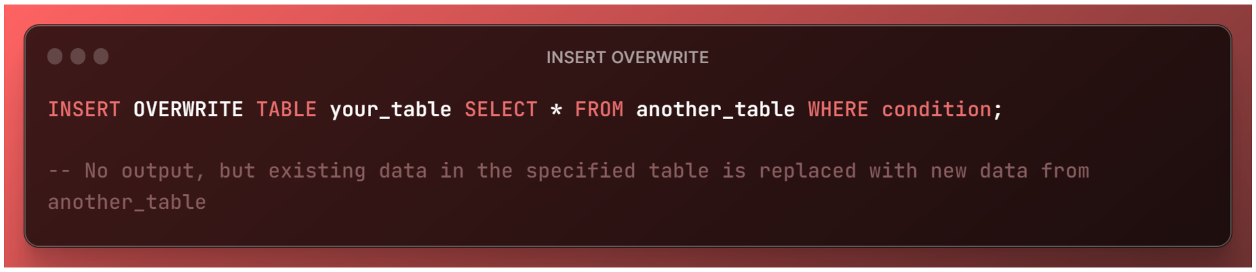
```
DELETE FROM your_table WHERE column1 = 'value_to_delete';

-- No output, but the specified rows in the table are deleted
```



```
LOAD DATA LOCAL INPATH 'local_path/data.txt' INTO TABLE your_table;

-- No output, but data from the external file is loaded into the specified table
```



```
INSERT OVERWRITE TABLE your_table SELECT * FROM another_table WHERE condition;

-- No output, but existing data in the specified table is replaced with new data from
another_table
```

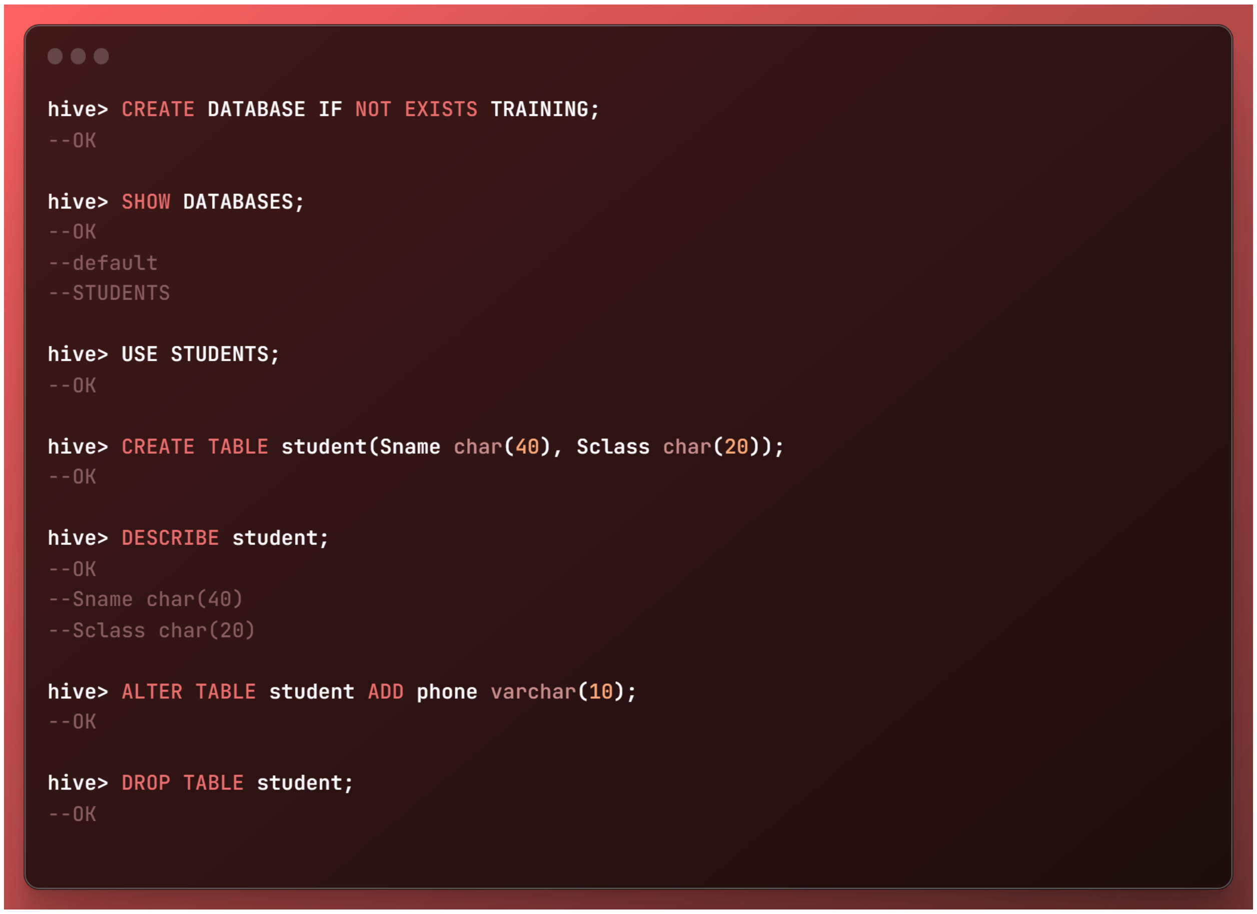
Question 13: Implement a HIVE DDL commands &
Question 15: Implement the following HIVE commands: CREATE, SHOW, DESCRIBE, USE, DROP, & ALTER

Procedure / Algo:

- CREATE - Used to create a new table in Hive.
- SHOW - Used to display all the tables in the database.
- DESCRIBE - Used to display the schema of a table
- USE - Used to select database
- DROP - Used to delete table
- ALTER - Used to modify an existing table

****Before Executing these commands on hive terminal, first start it by the above last 3 steps on command prompts****

Program & Output:



```
hive> CREATE DATABASE IF NOT EXISTS TRAINING;
--OK

hive> SHOW DATABASES;
--OK
--default
--STUDENTS

hive> USE STUDENTS;
--OK

hive> CREATE TABLE student(Sname char(40), Sclass char(20));
--OK

hive> DESCRIBE student;
--OK
--Sname char(40)
--Sclass char(20)

hive> ALTER TABLE student ADD phone varchar(10);
--OK

hive> DROP TABLE student;
--OK
```

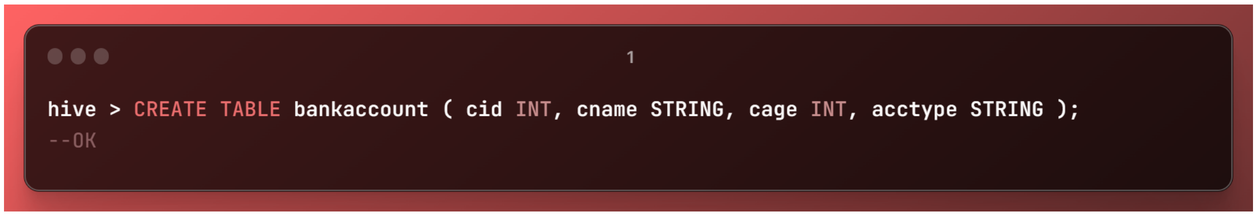
Question 19: Implement a HQL command
(I) To create a table as bankaccount with field name as cid,cname,cage,acctype
(II) Load data into table
(III) Display the table content.

Procedure / Algo:

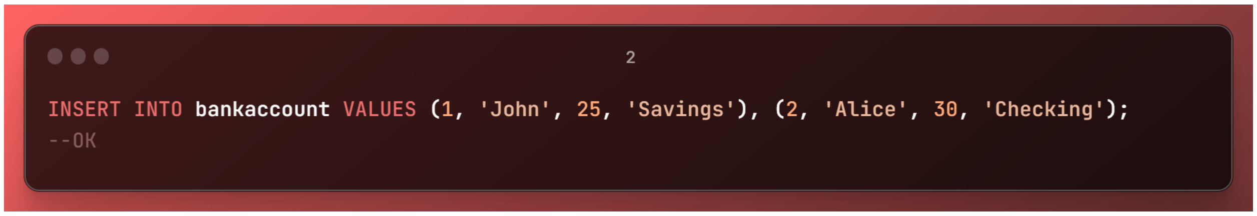
- Just write it on your own
- By referring your output screen and program

****Before Executing these commands on hive terminal, first start it by the above last 3 steps on command prompts****

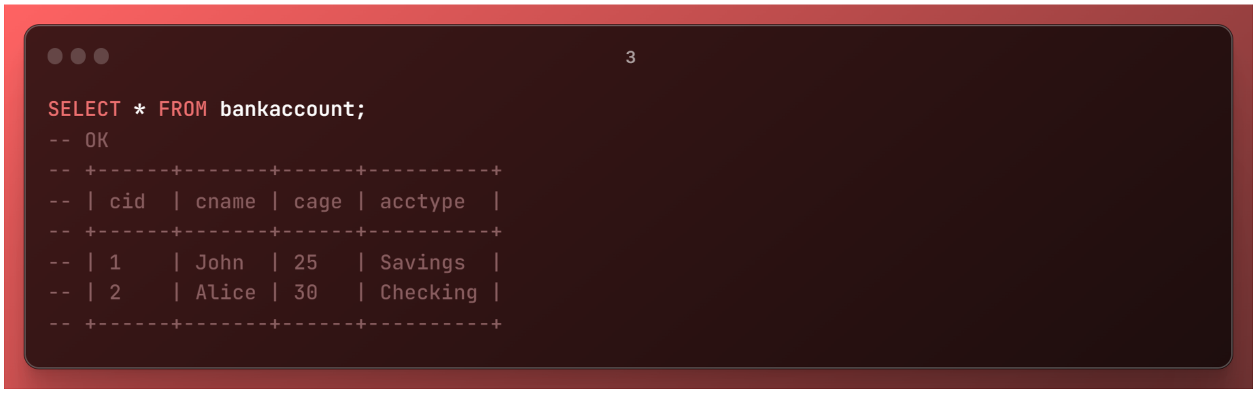
Program & Output:



```
hive > CREATE TABLE bankaccount ( cid INT, cname STRING, cage INT, acctype STRING );
--OK
```



```
INSERT INTO bankaccount VALUES (1, 'John', 25, 'Savings'), (2, 'Alice', 30, 'Checking');
--OK
```



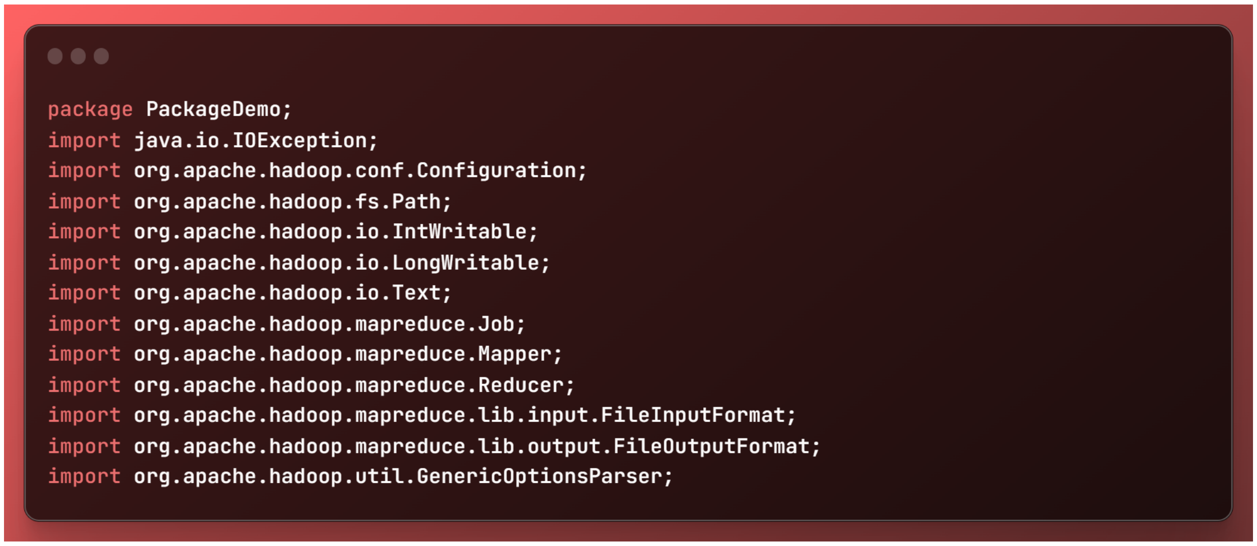
```
SELECT * FROM bankaccount;
-- OK
-- +------+-------+------+----------+
-- | cid  | cname | cage | acctype  |
-- +------+-------+------+----------+
-- | 1    | John  | 25   | Savings  |
-- | 2    | Alice | 30   | Checking |
-- +------+-------+------+----------+
```

Question 4: Using Map Reduce concept, Implement a Java program for count number of words.

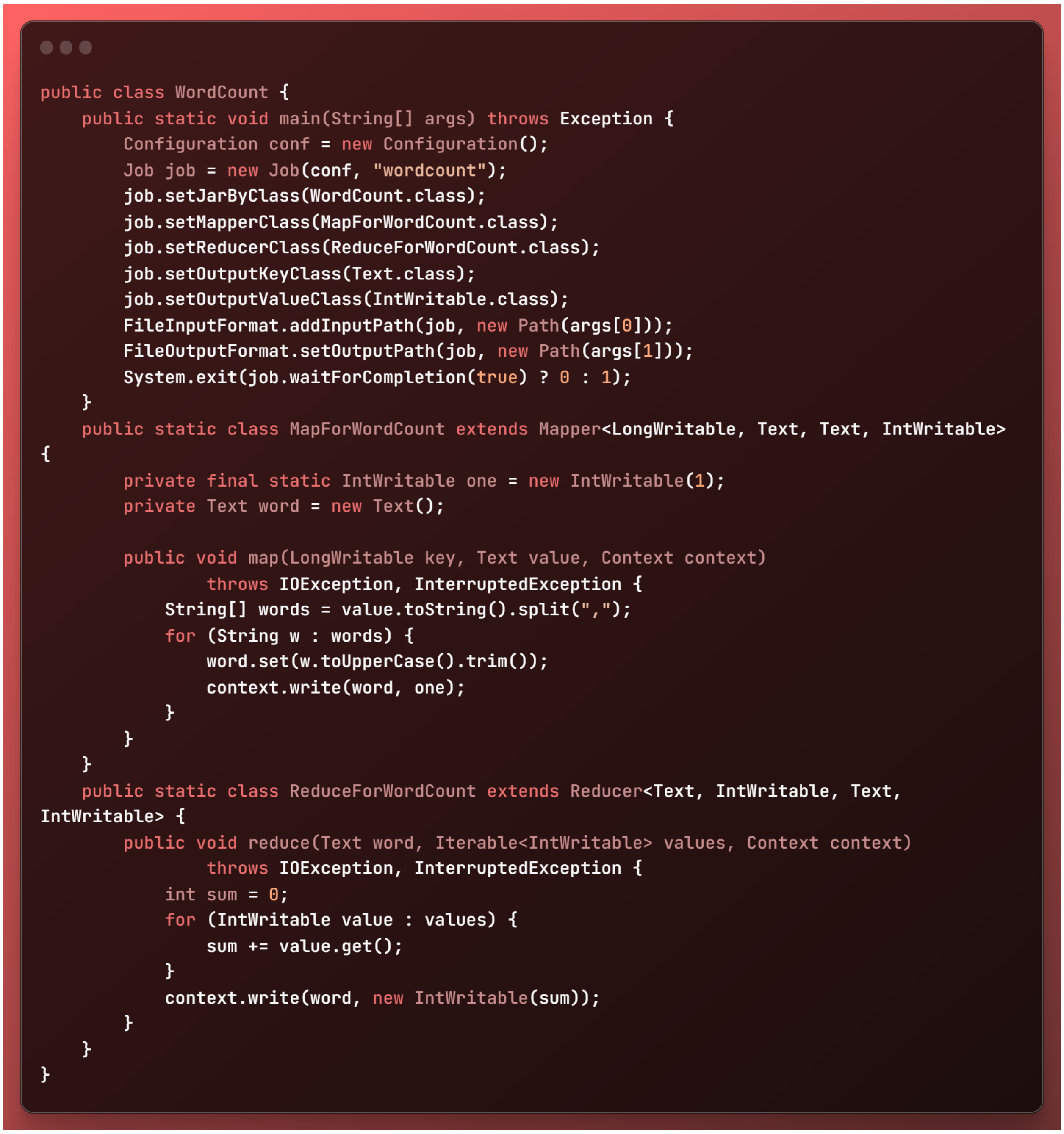
Procedure / Algo:

- Step 1: Open Eclipse> File > New > Java Project > (as - MRProgramsDemo) > Finish
- Step 2: Right Click > New > Package (as - PackageDemo) > Finish
- Step 3: Right Click on Package > New > Class (as - WordCount)
- Step 4: Add the required Reference Libraries
    - > /usr/lib/hadoop-0.20/hadoop-core.jar
    - > /usr/lib/hadoop-0.20/lib/Commons-cli-1.2.jar
- Step 5: Generated Jar File
    - Right Click on Project > Export > Select export destination as Jar File > next > Finish
- Step 6: To Move this into Hadoop directly,
    - $ hadoop fs -put wordcountFile /input_dir
- Step 7: Run Jar file
    - sbin > Hadoop jar MRProgramsDemo.jar PackageDemo.WordCount /input_dir /out

Program:



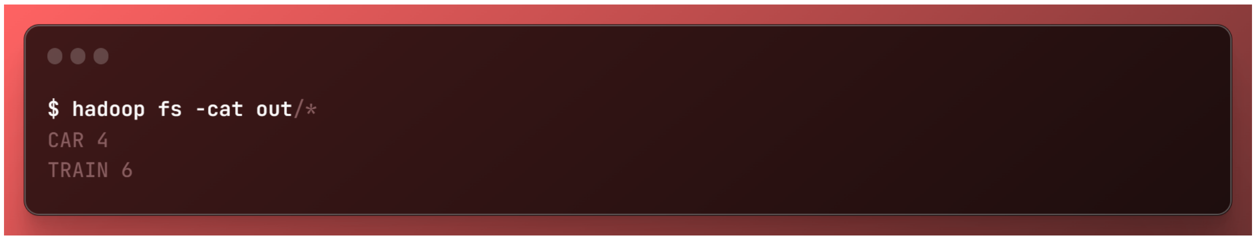
```java
package PackageDemo;
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
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;
```



```java
public class WordCount {
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = new Job(conf, "wordcount");
        job.setJarByClass(WordCount.class);
        job.setMapperClass(MapForWordCount.class);
        job.setReducerClass(ReduceForWordCount.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
    public static class MapForWordCount extends Mapper<LongWritable, Text, Text, IntWritable>
{
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(LongWritable key, Text value, Context context)
                throws IOException, InterruptedException {
            String[] words = value.toString().split(",");
            for (String w : words) {
                word.set(w.toUpperCase().trim());
                context.write(word, one);
            }
        }
    }
    public static class ReduceForWordCount extends Reducer<Text, IntWritable, Text,
IntWritable> {
        public void reduce(Text word, Iterable<IntWritable> values, Context context)
                throws IOException, InterruptedException {
            int sum = 0;
            for (IntWritable value : values) {
                sum += value.get();
            }
            context.write(word, new IntWritable(sum));
        }
    }
}
```

Output:



```
$ hadoop fs -cat out/*
CAR 4
TRAIN 6
```