BIG DATA ANALYTICS

# LAB ASSIGNMENT 6

**Mahesh Pachare**

FINAL YEAR B.TECH
IT 191080054

**Aim:** To set up and Install Apache Hive and Pig and to write the observations.

## Theory:

Apache Hive and Pig are two popular tools used in the Hadoop ecosystem for processing and analyzing large datasets stored in Hadoop clusters. Apache Hive is a data warehousing tool that provides a SQL-like interface for querying and analyzing data stored in Hadoop Distributed File System (HDFS) and other compatible file systems. Hive translates SQL-like queries, written in Hive Query Language (HQL), into MapReduce jobs, which can be executed on a Hadoop cluster. Hive also provides support for data serialization, data partitioning, and user-defined functions (UDFs), which enable users to perform complex data processing operations.

Apache Pig, on the other hand, is a data flow language used for processing and analyzing large datasets. Pig Latin is the language used to write Pig scripts, which are translated into MapReduce jobs that can be executed on a Hadoop cluster. Pig provides a rich set of operators and functions that enable users to perform a wide range of data processing operations, including filtering, grouping, joining, and aggregation.

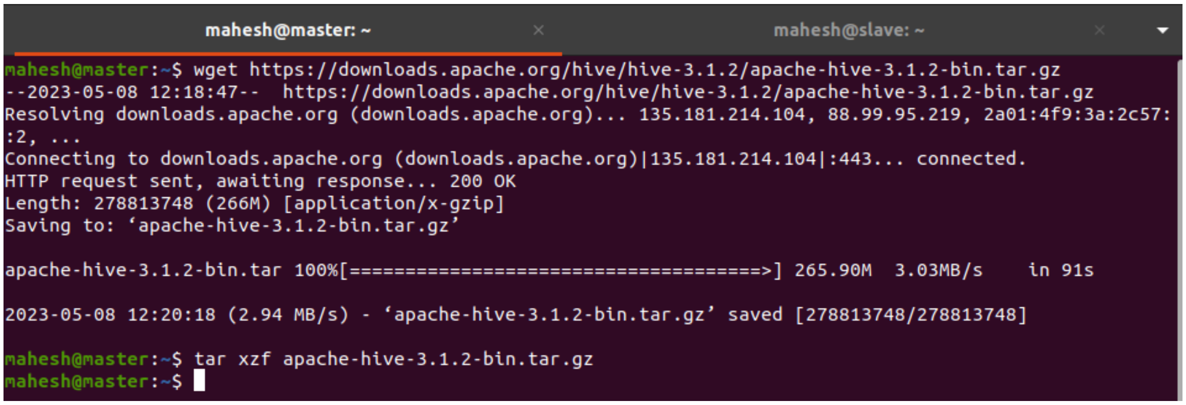
Both Hive and Pig provide a high-level abstraction over the underlying MapReduce framework, which makes it easier for users to process and analyze large datasets without having to write low-level MapReduce code. Hive and Pig also provide a way to store the processed data in various formats such as Apache Avro, ORC, and Parquet. Additionally, both tools support integration with other Hadoop ecosystem tools such as Apache Spark, Apache Kafka, and Apache HBase.

Apache Hive and Pig are two popular tools used for processing and analyzing large datasets stored in Hadoop clusters. Hive provides a SQL-like interface, whole Pig provides a data flow language. Both tools provide a high-level abstraction over the underlying MapReduce framework and support a wide range of data processing operations.

# Implementation:

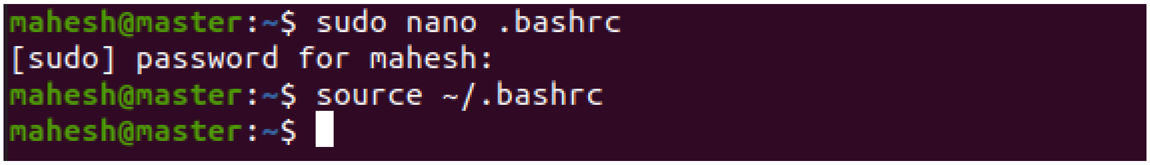
## Hive Installation:

1. Download the hive and unzip the tar file.

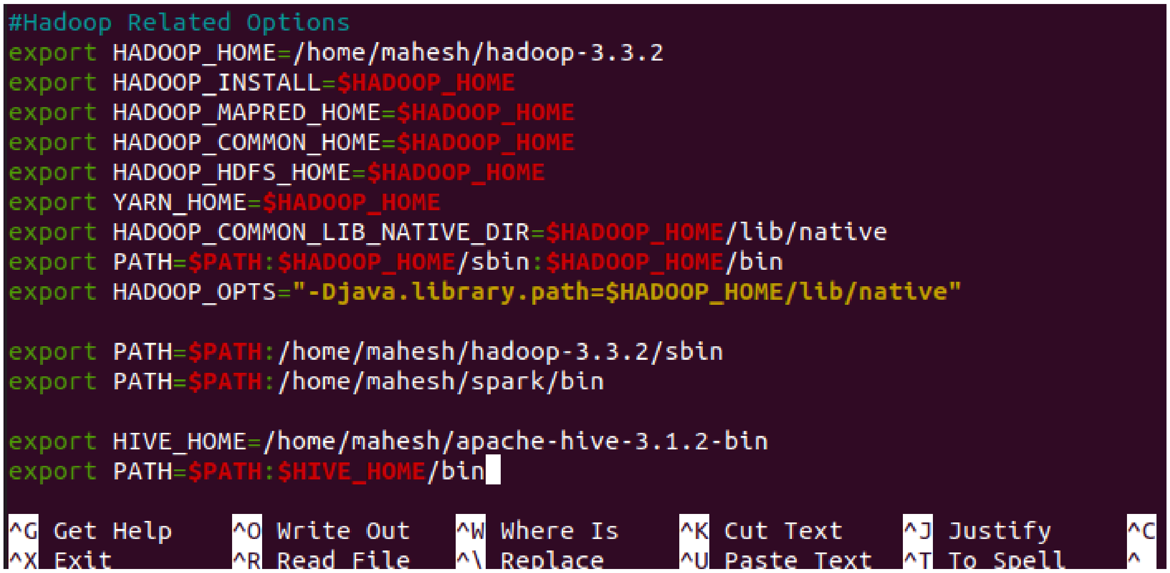


```
mahesh@master:~$ wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
--2023-05-08 12:18:47--  https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 135.181.214.104, 88.99.95.219, 2a01:4f9:3a:2c57:
:2, ...
Connecting to downloads.apache.org (downloads.apache.org)|135.181.214.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 278813748 (266M) [application/x-gzip]
Saving to: ‘apache-hive-3.1.2-bin.tar.gz’

apache-hive-3.1.2-bin.tar 100%[=====================================>] 265.90M  3.03MB/s    in 91s

2023-05-08 12:20:18 (2.94 MB/s) - ‘apache-hive-3.1.2-bin.tar.gz’ saved [278813748/278813748]

mahesh@master:~$ tar xzf apache-hive-3.1.2-bin.tar.gz
mahesh@master:~$ 
```

2. Update the .bashrc file with path variables and run it with the source command.

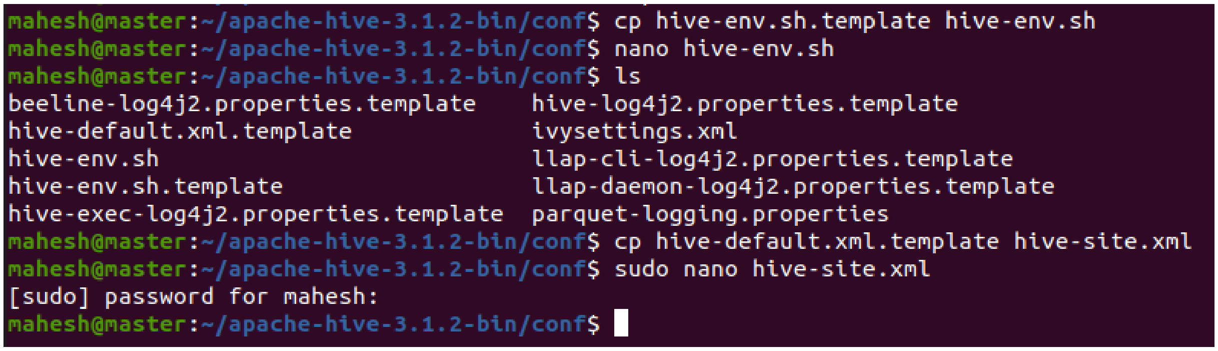


```
mahesh@master:~$ sudo nano .bashrc
[sudo] password for mahesh:
mahesh@master:~$ source ~/.bashrc
mahesh@master:~$ 
```



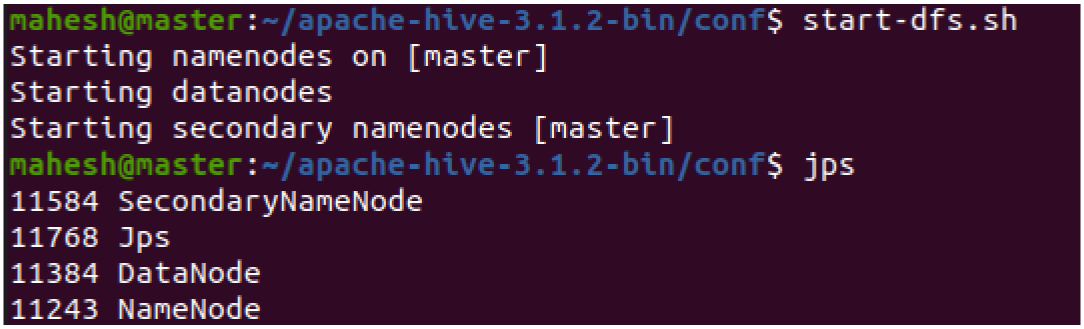
```
#Hadoop Related Options
export HADOOP_HOME=/home/mahesh/hadoop-3.3.2
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"

export PATH=$PATH:/home/mahesh/hadoop-3.3.2/sbin
export PATH=$PATH:/home/mahesh/spark/bin

export HIVE_HOME=/home/mahesh/apache-hive-3.1.2-bin
export PATH=$PATH:$HIVE_HOME/bin
```

3. Copy the env template and update the hive env file with environment variables.



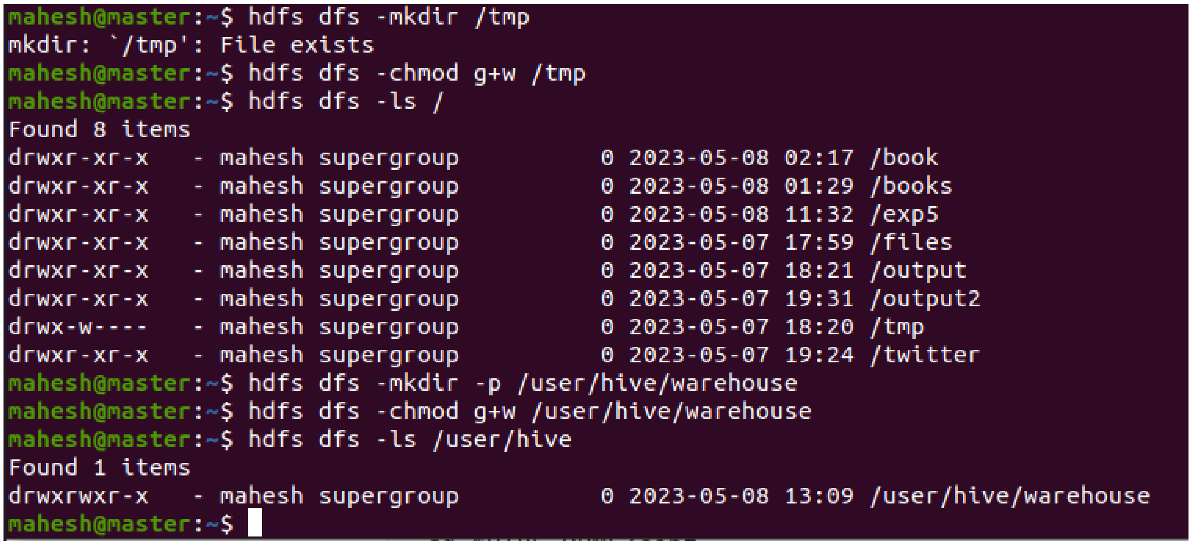
```
mahesh@master:~/apache-hive-3.1.2-bin/conf$ cp hive-env.sh.template hive-env.sh
mahesh@master:~/apache-hive-3.1.2-bin/conf$ nano hive-env.sh
mahesh@master:~/apache-hive-3.1.2-bin/conf$ ls
beeline-log4j2.properties.template    hive-log4j2.properties.template
hive-default.xml.template             ivysettings.xml
hive-env.sh                           llap-cli-log4j2.properties.template
hive-env.sh.template                  llap-daemon-log4j2.properties.template
hive-exec-log4j2.properties.template  parquet-logging.properties
mahesh@master:~/apache-hive-3.1.2-bin/conf$ cp hive-default.xml.template hive-site.xml
mahesh@master:~/apache-hive-3.1.2-bin/conf$ sudo nano hive-site.xml
[sudo] password for mahesh:
mahesh@master:~/apache-hive-3.1.2-bin/conf$ 
```

4. Starting the Hadoop file system and Verifying the status.



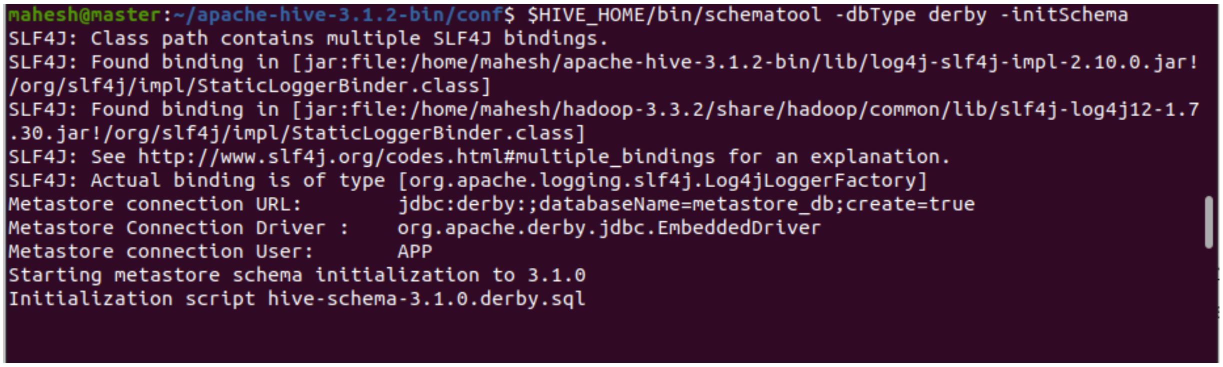
```
mahesh@master:~/apache-hive-3.1.2-bin/conf$ start-dfs.sh
Starting namenodes on [master]
Starting datanodes
Starting secondary namenodes [master]
mahesh@master:~/apache-hive-3.1.2-bin/conf$ jps
11584 SecondaryNameNode
11768 Jps
11384 DataNode
11243 NameNode
```

5. Making a directory for the hive.

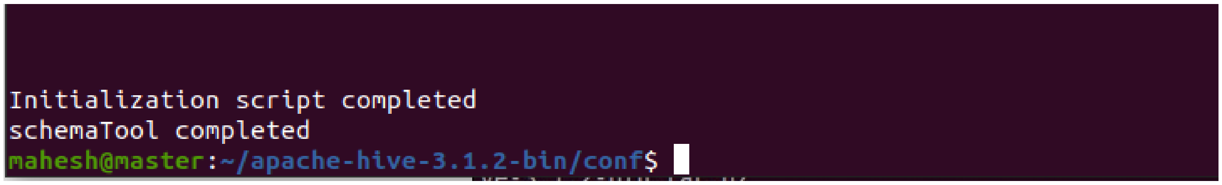


```
mahesh@master:~$ hdfs dfs -mkdir /tmp
mkdir: `/tmp': File exists
mahesh@master:~$ hdfs dfs -chmod g+w /tmp
mahesh@master:~$ hdfs dfs -ls /
Found 8 items
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 02:17 /book
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 01:29 /books
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 11:32 /exp5
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 17:59 /files
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 18:21 /output
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 19:31 /output2
drwx-w----   - mahesh supergroup          0 2023-05-07 18:20 /tmp
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 19:24 /twitter
mahesh@master:~$ hdfs dfs -mkdir -p /user/hive/warehouse
mahesh@master:~$ hdfs dfs -chmod g+w /user/hive/warehouse
mahesh@master:~$ hdfs dfs -ls /user/hive
Found 1 items
drwxrwxr-x   - mahesh supergroup          0 2023-05-08 13:09 /user/hive/warehouse
mahesh@master:~$ 
```

6. Initializing the schema using derby.

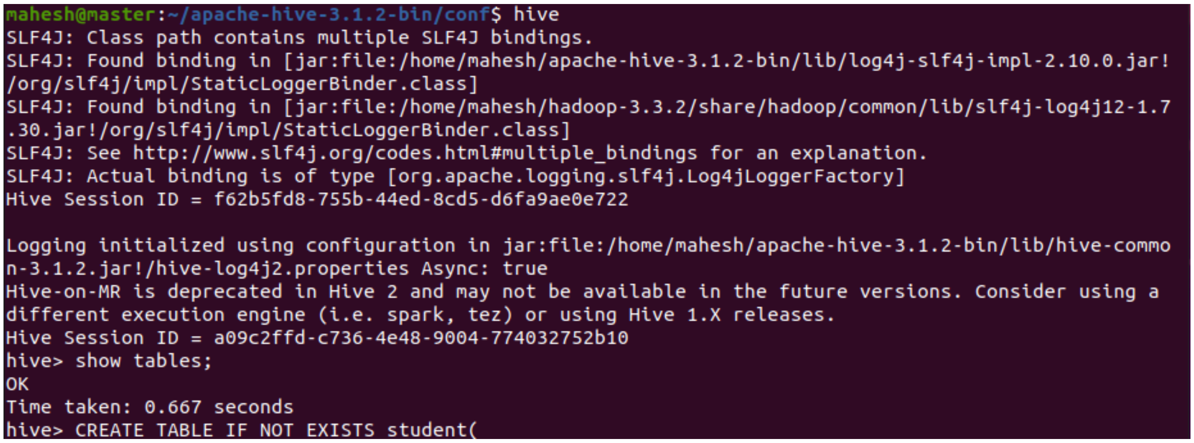


```
mahesh@master:~/apache-hive-3.1.2-bin/conf$ $HIVE_HOME/bin/schematool -dbType derby -initSchema
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/mahesh/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!
/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/mahesh/hadoop-3.3.2/share/hadoop/common/lib/slf4j-log4j12-1.7
.30.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]
Metastore connection URL:        jdbc:derby:;databaseName=metastore_db;create=true
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql
```



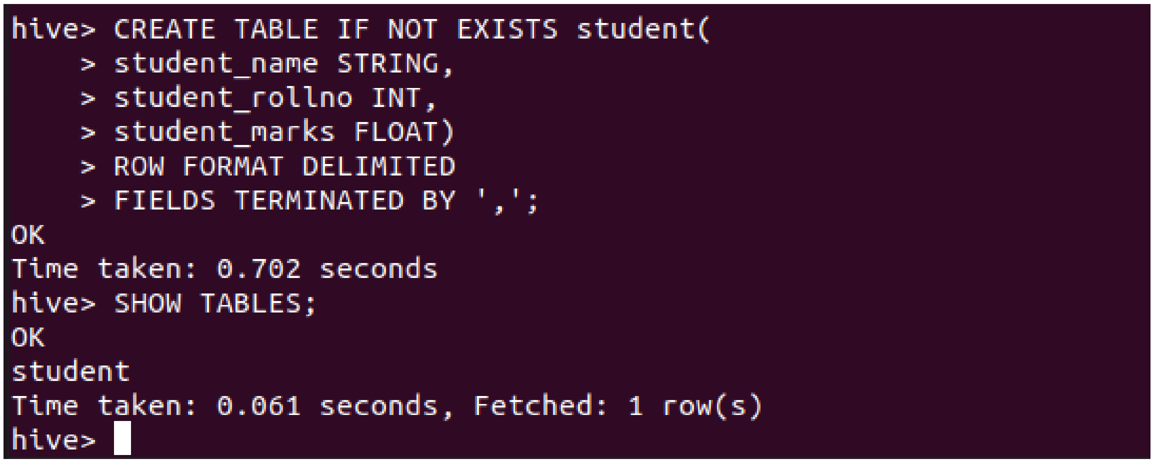
```
Initialization script completed
schemaTool completed
mahesh@master:~/apache-hive-3.1.2-bin/conf$
```

7. Starting hive and creating the testdb database.



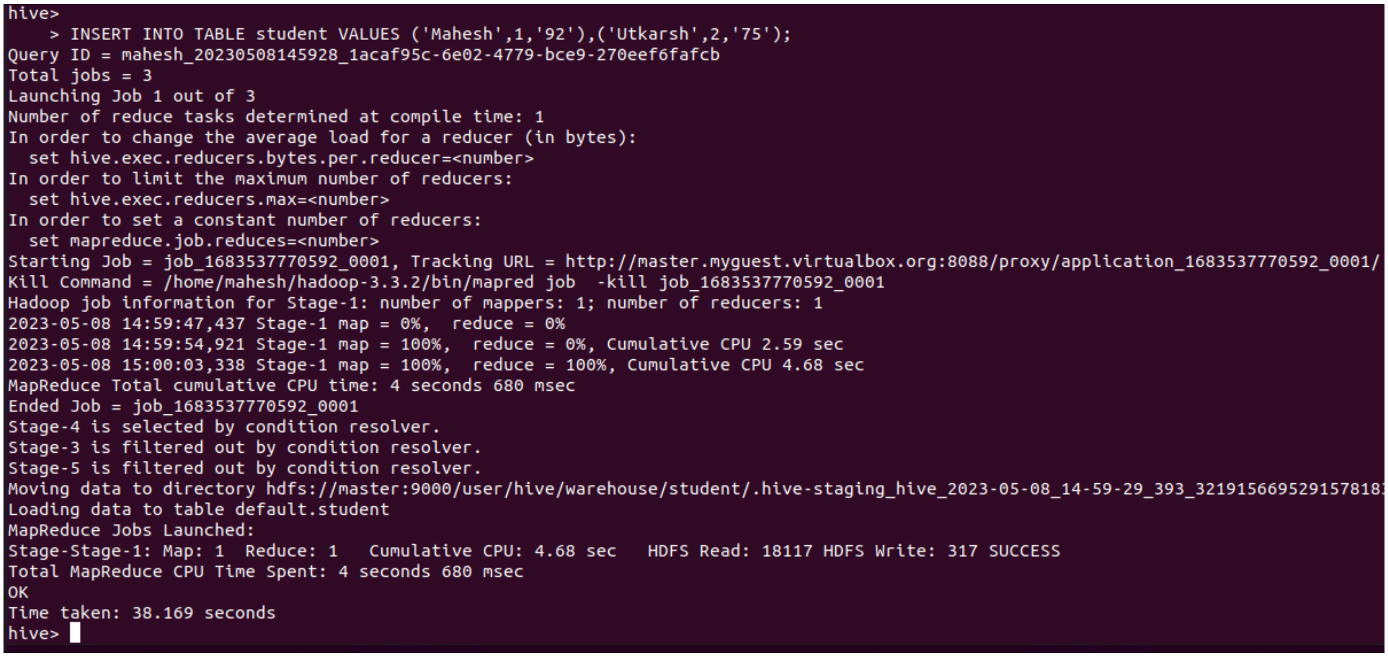
```
mahesh@master:~/apache-hive-3.1.2-bin/conf$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/mahesh/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!
/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/mahesh/hadoop-3.3.2/share/hadoop/common/lib/slf4j-log4j12-1.7
.30.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]
Hive Session ID = f62b5fd8-755b-44ed-8cd5-d6fa9ae0e722

Logging initialized using configuration in jar:file:/home/mahesh/apache-hive-3.1.2-bin/lib/hive-commo
n-3.1.2.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a
different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Hive Session ID = a09c2ffd-c736-4e48-9004-774032752b10
hive> show tables;
OK
Time taken: 0.667 seconds
hive> CREATE TABLE IF NOT EXISTS student(
```

8. Creating the student table and displaying it.



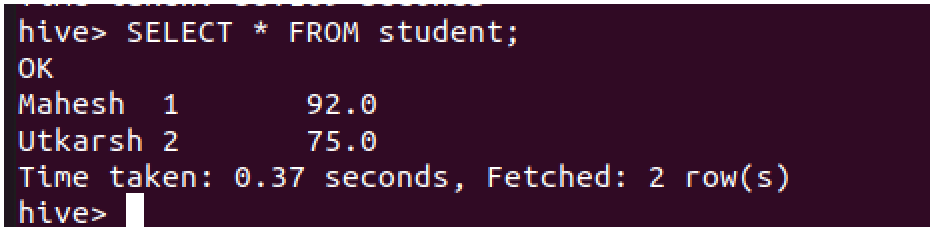
```
hive> CREATE TABLE IF NOT EXISTS student(
    > student_name STRING,
    > student_rollno INT,
    > student_marks FLOAT)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ',';
OK
Time taken: 0.702 seconds
hive> SHOW TABLES;
OK
student
Time taken: 0.061 seconds, Fetched: 1 row(s)
hive>
```

9. Inserting data into the student table.



```
hive>
    > INSERT INTO TABLE student VALUES ('Mahesh',1,'92'),('Utkarsh',2,'75');
Query ID = mahesh_20230508145928_1acaf95c-6e02-4779-bce9-270eef6fafcb
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1683537770592_0001, Tracking URL = http://master.myguest.virtualbox.org:8088/proxy/application_1683537770592_0001/
Kill Command = /home/mahesh/hadoop-3.3.2/bin/mapred job  -kill job_1683537770592_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2023-05-08 14:59:47,437 Stage-1 map = 0%,  reduce = 0%
2023-05-08 14:59:54,921 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.59 sec
2023-05-08 15:00:03,338 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.68 sec
MapReduce Total cumulative CPU time: 4 seconds 680 msec
Ended Job = job_1683537770592_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://master:9000/user/hive/warehouse/student/.hive-staging_hive_2023-05-08_14-59-29_393_3219156695291578181
Loading data to table default.student
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.68 sec   HDFS Read: 18117 HDFS Write: 317 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 680 msec
OK
Time taken: 38.169 seconds
hive>
```

10. Displaying the inserted data.

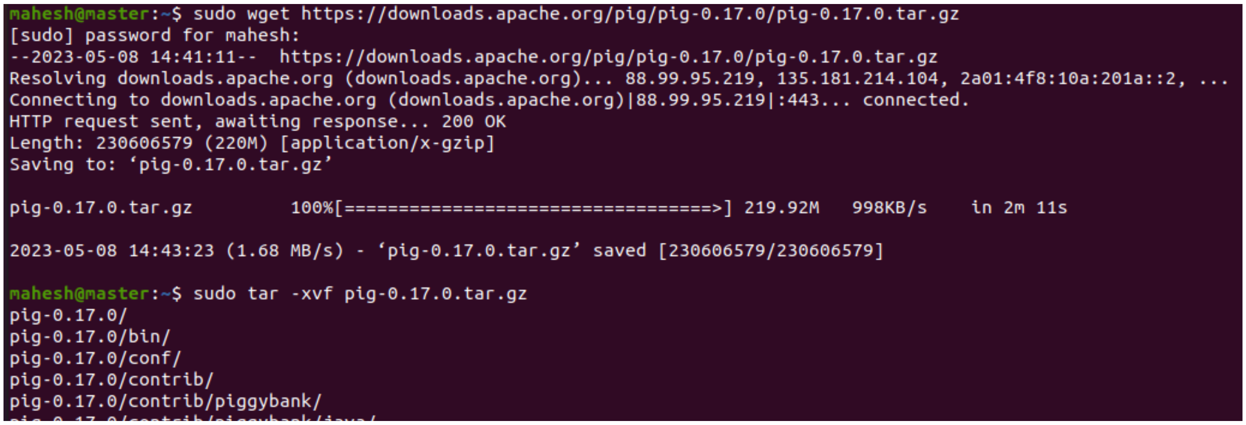


```
hive> SELECT * FROM student;
OK
Mahesh  1       92.0
Utkarsh 2       75.0
Time taken: 0.37 seconds, Fetched: 2 row(s)
hive>
```

## Installing Apache Pig:

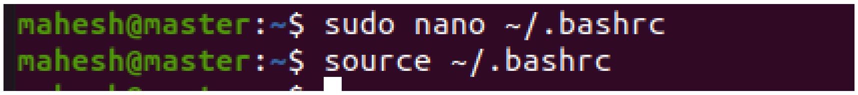
1. Downloading the Pig 0.15.0 version, unzipping the tar file

   .

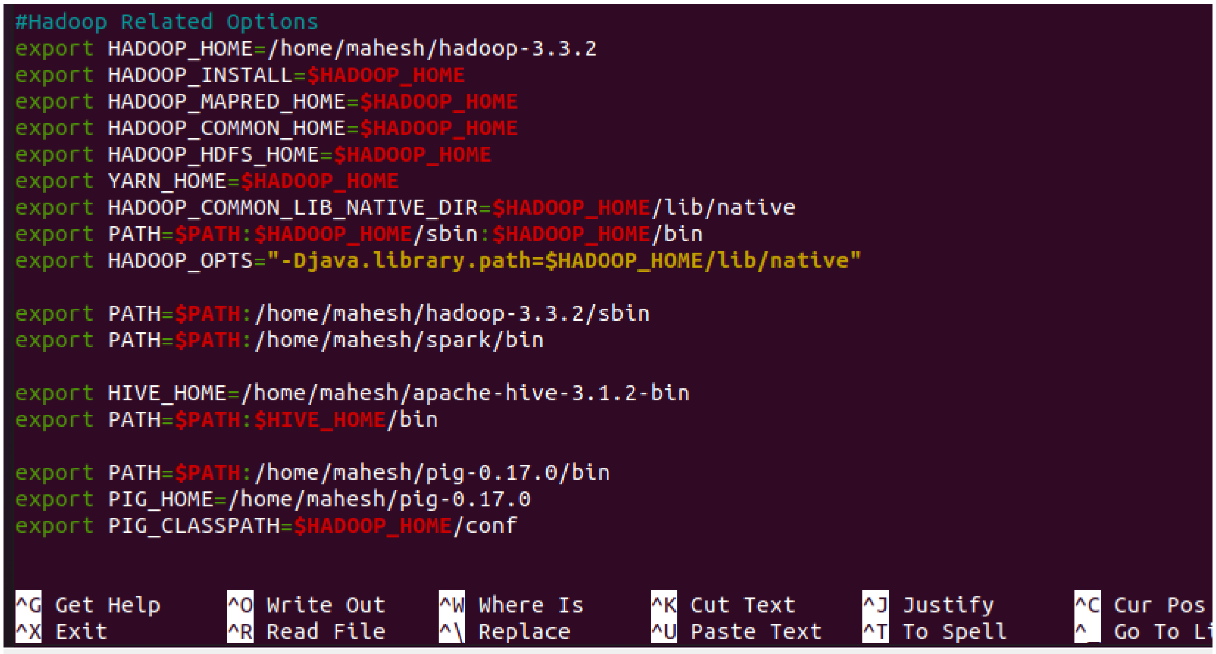


```
mahesh@master:~$ sudo wget https://downloads.apache.org/pig/pig-0.17.0/pig-0.17.0.tar.gz
[sudo] password for mahesh:
--2023-05-08 14:41:11--  https://downloads.apache.org/pig/pig-0.17.0/pig-0.17.0.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 88.99.95.219, 135.181.214.104, 2a01:4f8:10a:201a::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|88.99.95.219|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 230606579 (220M) [application/x-gzip]
Saving to: ‘pig-0.17.0.tar.gz’

pig-0.17.0.tar.gz         100%[=================================>] 219.92M   998KB/s    in 2m 11s

2023-05-08 14:43:23 (1.68 MB/s) - ‘pig-0.17.0.tar.gz’ saved [230606579/230606579]

mahesh@master:~$ sudo tar -xvf pig-0.17.0.tar.gz
pig-0.17.0/
pig-0.17.0/bin/
pig-0.17.0/conf/
pig-0.17.0/contrib/
pig-0.17.0/contrib/piggybank/
pig-0.17.0/contrib/piggybank/java/
```

2. Updating the .bashrc file with pig path variables and running it using source.

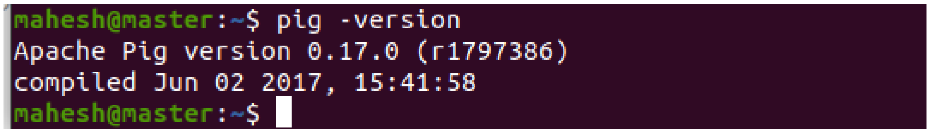


```
mahesh@master:~$ sudo nano ~/.bashrc
mahesh@master:~$ source ~/.bashrc
```



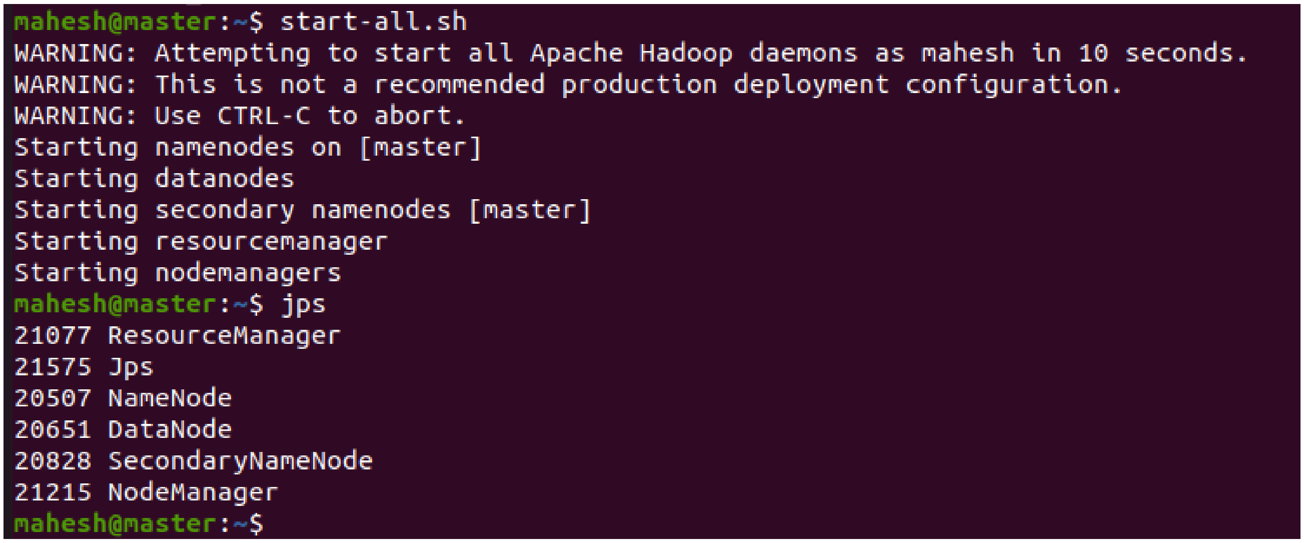
```
#Hadoop Related Options
export HADOOP_HOME=/home/mahesh/hadoop-3.3.2
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"

export PATH=$PATH:/home/mahesh/hadoop-3.3.2/sbin
export PATH=$PATH:/home/mahesh/spark/bin

export HIVE_HOME=/home/mahesh/apache-hive-3.1.2-bin
export PATH=$PATH:$HIVE_HOME/bin

export PATH=$PATH:/home/mahesh/pig-0.17.0/bin
export PIG_HOME=/home/mahesh/pig-0.17.0
export PIG_CLASSPATH=$HADOOP_HOME/conf
```

3. Verifying the pig.



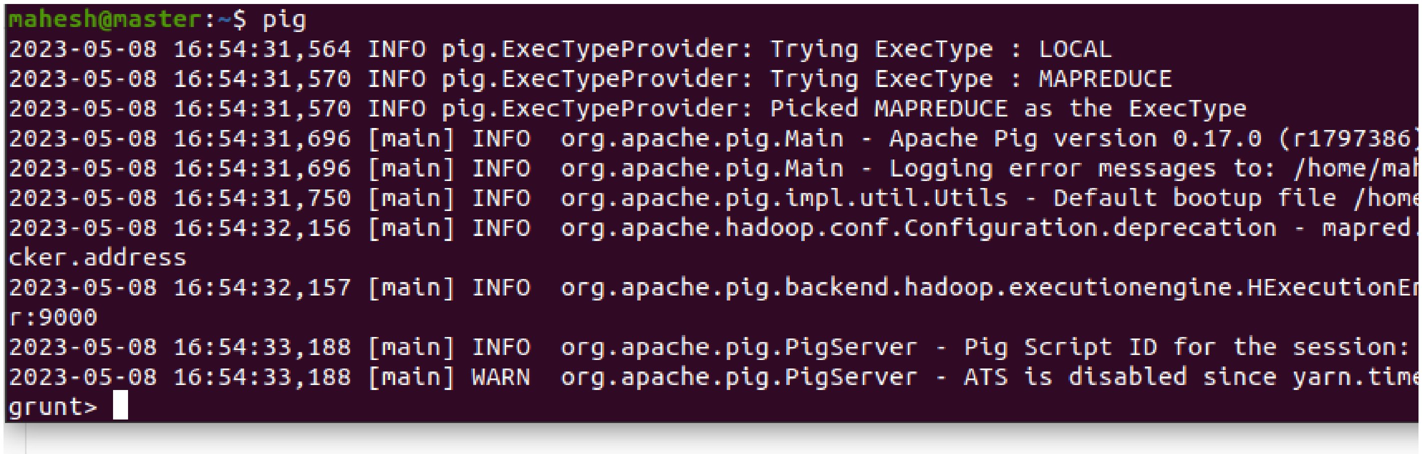
```
mahesh@master:~$ pig -version
Apache Pig version 0.17.0 (r1797386)
compiled Jun 02 2017, 15:41:58
mahesh@master:~$ 
```

4. Starting the Hadoop setup.



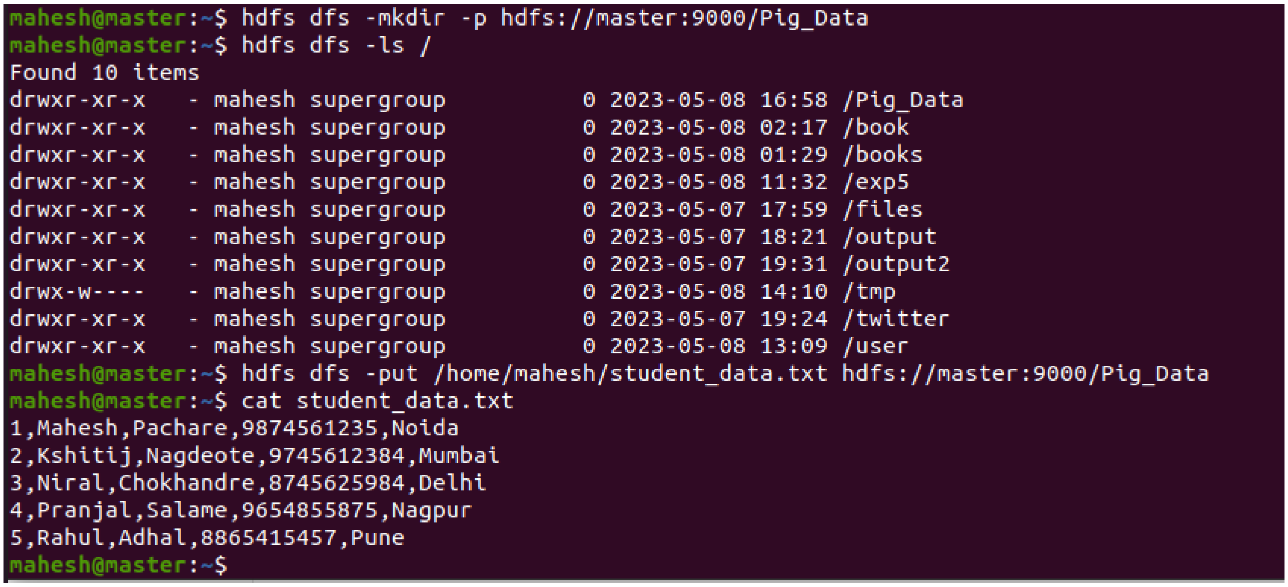
```
mahesh@master:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as mahesh in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [master]
Starting datanodes
Starting secondary namenodes [master]
Starting resourcemanager
Starting nodemanagers
mahesh@master:~$ jps
21077 ResourceManager
21575 Jps
20507 NameNode
20651 DataNode
20828 SecondaryNameNode
21215 NodeManager
mahesh@master:~$
```

5. Starting pig shell - Grunt.



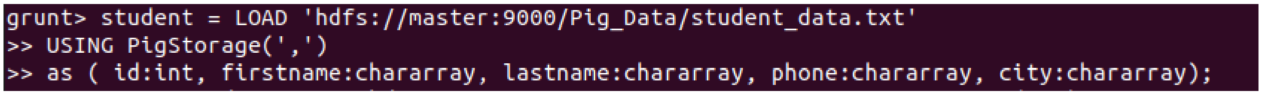
```
mahesh@master:~$ pig
2023-05-08 16:54:31,564 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2023-05-08 16:54:31,570 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
2023-05-08 16:54:31,570 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2023-05-08 16:54:31,696 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.0 (r1797386
2023-05-08 16:54:31,696 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/mah
2023-05-08 16:54:31,750 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home
2023-05-08 16:54:32,156 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.
cker.address
2023-05-08 16:54:32,157 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEn
r:9000
2023-05-08 16:54:33,188 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: 
2023-05-08 16:54:33,188 [main] WARN  org.apache.pig.PigServer - ATS is disabled since yarn.time
grunt> 
```

6. Creating the Pig_Data directory in hdfs and putting student_data.txt which has csv data in it.



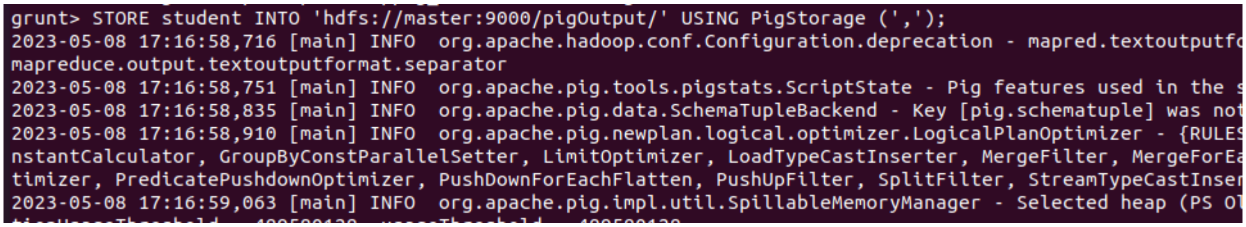
```
mahesh@master:~$ hdfs dfs -mkdir -p hdfs://master:9000/Pig_Data
mahesh@master:~$ hdfs dfs -ls /
Found 10 items
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 16:58 /Pig_Data
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 02:17 /book
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 01:29 /books
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 11:32 /exp5
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 17:59 /files
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 18:21 /output
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 19:31 /output2
drwx-w----   - mahesh supergroup          0 2023-05-08 14:10 /tmp
drwxr-xr-x   - mahesh supergroup          0 2023-05-07 19:24 /twitter
drwxr-xr-x   - mahesh supergroup          0 2023-05-08 13:09 /user
mahesh@master:~$ hdfs dfs -put /home/mahesh/student_data.txt hdfs://master:9000/Pig_Data
mahesh@master:~$ cat student_data.txt
1,Mahesh,Pachare,9874561235,Noida
2,Kshitij,Nagdeote,9745612384,Mumbai
3,Niral,Chokhandre,8745625984,Delhi
4,Pranjal,Salame,9654855875,Nagpur
5,Rahul,Adhal,8865415457,Pune
mahesh@master:~$
```

7. Loading the data using the Load command present in the Hadoop file system.

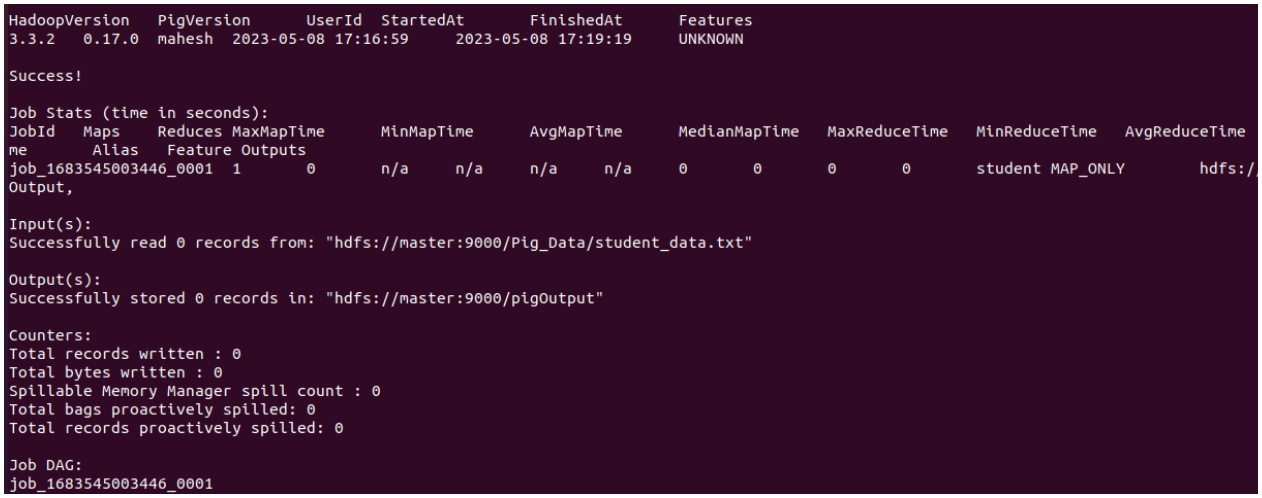


```
grunt> student = LOAD 'hdfs://master:9000/Pig_Data/student_data.txt'
>> USING PigStorage(',')
>> as ( id:int, firstname:chararray, lastname:chararray, phone:chararray, city:chararray);
```

8. Using the store command to store loaded data into the HDFS file system.

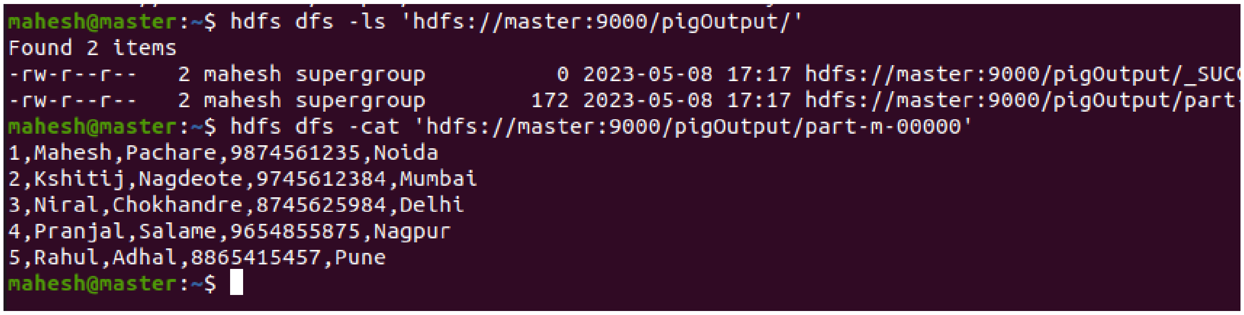


```
grunt> STORE student INTO 'hdfs://master:9000/pigOutput/' USING PigStorage (',');
2023-05-08 17:16:58,716 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.textoutputfo
mapreduce.output.textoutputformat.separator
2023-05-08 17:16:58,751 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the s
2023-05-08 17:16:58,835 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not
2023-05-08 17:16:58,910 [main] INFO  org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES
nstantCalculator, GroupByConstParallelSetter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEa
timizer, PredicatePushdownOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInser
2023-05-08 17:16:59,063 [main] INFO  org.apache.pig.impl.util.SpillableMemoryManager - Selected heap (PS Ol
```



```
HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
3.3.2   0.17.0  mahesh  2023-05-08 17:16:59     2023-05-08 17:19:19     UNKNOWN

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTime      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime
me       Alias   Feature Outputs
job_1683545003446_0001  1       0       n/a     n/a     n/a     n/a     0       0       0       0       student MAP_ONLY        hdfs:/
Output,

Input(s):
Successfully read 0 records from: "hdfs://master:9000/Pig_Data/student_data.txt"

Output(s):
Successfully stored 0 records in: "hdfs://master:9000/pigOutput"

Counters:
Total records written : 0
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1683545003446_0001
```

9. Stored file output in Hadoop.



```
mahesh@master:~$ hdfs dfs -ls 'hdfs://master:9000/pigOutput/'
Found 2 items
-rw-r--r--   2 mahesh supergroup          0 2023-05-08 17:17 hdfs://master:9000/pigOutput/_SUC
-rw-r--r--   2 mahesh supergroup        172 2023-05-08 17:17 hdfs://master:9000/pigOutput/part
mahesh@master:~$ hdfs dfs -cat 'hdfs://master:9000/pigOutput/part-m-00000'
1,Mahesh,Pachare,9874561235,Noida
2,Kshitij,Nagdeote,9745612384,Mumbai
3,Niral,Chokhandre,8745625984,Delhi
4,Pranjal,Salame,9654855875,Nagpur
5,Rahul,Adhal,8865415457,Pune
mahesh@master:~$ 
```

**Conclusion:** In this experiment we have successfully installed hive over map reduce layer, created a student table, and inserted data in it using SQL queries.

Also we have installed apache pig, loaded a sample student data, and stored it back to the Hadoop file system in the output folder. Since hive requires RDBMs to store its metadata when needed to install and run derby service which is made for hive only while executing SQL statements. Yarn should also be present in running state as it is required to run the Map Reduce Job submitted by the hive. For Pig, it has 2 modes of execution one is local and another is map reduce which is the default, so we need to run the Hadoop file system, Yarn, and MapReduce before interacting with Pig for loading and storing data after performing processing.