

Exp.No: 4

Create User Defined Function (UDF) in Apache Pig and execute it in MapReduce

AIM:

To create User Define Function in Apache Pig and execute it on map reduce.

PROCEDURE:

Step-1: Go to <https://pig.apache.org/releases.html> and copy the path of the latest version of pig that you want to install. Run the following command to download Apache Pig in Ubuntu:

wget <https://dlcdn.apache.org/pig/pig-0.16.0/pig-0.16.0.tar.gz>

Step-2: To untar pig-0.16.0.tar.gz file run the following command:

tar xvzf pig-0.16.0.tar.gz

Step 3: To create a pig folder and move pig-0.16.0 to the pig folder, execute the following command:

sudo mv /home/hadoop/pig-0.16.0 /home/hadoop/pig

Step 4: Now open the .bashrc file to edit the path and variables/settings for pig. Run the following command:

sudo nano .bashrc

Add the below given to .bashrc file at the end and save the file.

#PIG settings

export PIG_HOME=/home/hadoop/pig

export PATH=\$PATH:\$PIG_HOME/bin

export PIG_CLASSPATH=\$PIG_HOME/conf:\$HADOOP_INSTALL/etc/hadoop/

export PIG_CONF_DIR=\$PIG_HOME/conf

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdkamd64
```

```
export PIG_CLASSPATH=$PIG_CONF_DIR:$PATH
```

#PIG setting ends

Step 5: Run the following command to make the changes effective in the .bashrc file:

```
source .bashrc
```

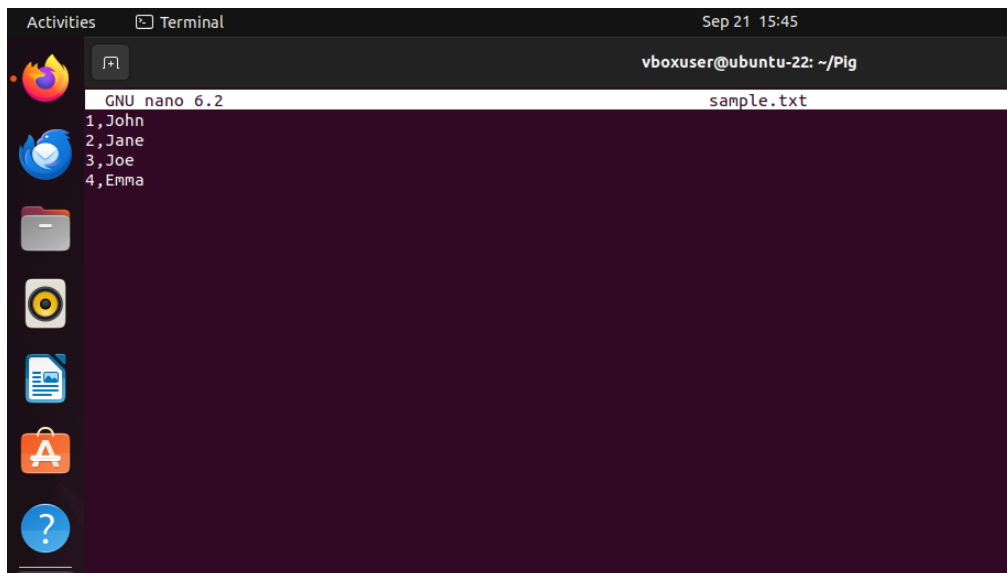
Step 6: To start all Hadoop daemons, navigate to the hadoop-3.2.1/sbin folder and run the following commands:

```
./start-dfs.sh
```

```
./start-yarn.sh
```

Step 7: Create a sample text file

```
nano sample.txt
```



Step 8: Add the text file to the Hadoop environment.

```
hadoop fs -put sample.txt /home/hadoop/piginput/
```

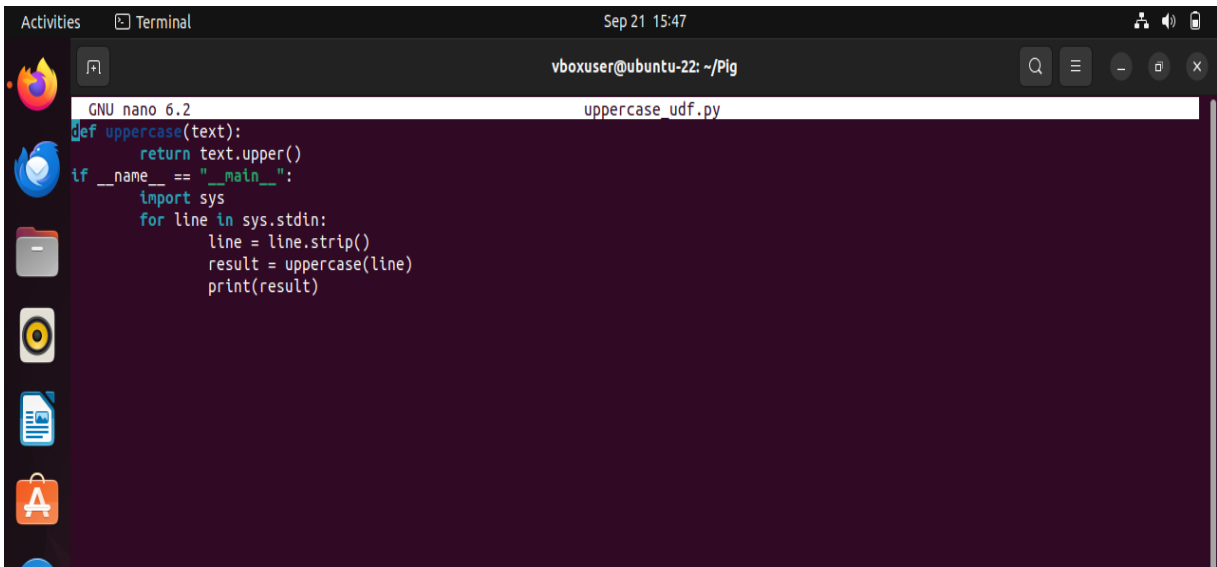
Step 9: Create PIG File

```
nano demo_pig.pig
```

A terminal window titled 'vboxuser@ubuntu-22: ~/Pig' showing a nano editor editing 'demo_pig.pig'. The script contains Pig Latin commands to load data from HDFS and dump it.

```
GNU nano 6.2 demo_pig.pig
-- Load the data from HDFS
data = LOAD '/home/vboxuser/piginput/sample.txt' USING PigStorage(',') AS (id:int,name:chararray);
-- Dump the data to check if it was loaded correctly
DUMP data;
```

Step 10: Create udf file and save as uppercase_udf.py

A terminal window titled 'vboxuser@ubuntu-22: ~/Pig' showing a nano editor editing 'uppercase_udf.py'. The code defines a Python function 'uppercase' and a main block that reads from stdin and prints the uppercase version of each line.

```
GNU nano 6.2 uppercase_udf.py
def uppercase(text):
    return text.upper()
if __name__ == "__main__":
    import sys
    for line in sys.stdin:
        line = line.strip()
        result = uppercase(line)
        print(result)
```

Step 11: Create the udfs folder on hadoop

hadoop fs -mkdir /home/hadoop/udfs

Step 12: Put the uppercase_udf.py in to the above folder

hdfs dfs -put uppercase_udf.py /home/hadoop/udfs/

Step 13: Create a file named udf_example.pig

nano udf_example.pig



```
GNU nano 6.2 udf_example.pig
-- Register the Python UDF script
REGISTER 'hdfs:///home/vboxuser/udfs/uppercase_udf.py' USING jython AS udf;
-- Load some data
data = LOAD 'hdfs:///home/vboxuser/piginput/sample.txt' AS (text:chararray);
-- Use the Python UDF
uppercased_data = FOREACH data GENERATE udf.uppercase(text) AS uppercase_text;
-- Store the result
STORE uppercased_data INTO 'hdfs:///home/vboxuser/pig_output_data';
```

Step 14: To view the output use the command below

hdfs dfs -cat /home/hadoop/pig_output_data/part-m-00000



```
vboxuser@ubuntu-22:~/Pig$ hdfs dfs -cat /home/vboxuser/pig_output_data/part-m-00000
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
1, JOHN
2, JANE
3, JOE
4, EMMA
vboxuser@ubuntu-22:~/Pig$
```

Step 15: The result in the Namenode is as follows:

The screenshot shows a web browser window at the URL `localhost:9870/explorer.html#/home/vboxuser/pig_output_data`. The interface displays a file explorer for the HDFS path `/home/vboxuser/pig_output_data`. A modal window titled "File information - part-m-00000" is open, showing details for the file `part-m-00000`. The modal includes tabs for "Download", "Head the file (first 32K)", and "Tail the file (last 32K)". The "Block information" section shows the following details:

- Block ID: 1073741964
- Block Pool ID: BP-305058674-127.0.1.1-1726117174333
- Generation Stamp: 1140
- Size: 31
- Availability:
 - ubuntu-22.4.myguest.virtualbox.org

The "File contents" section shows the following text:

```
1, JOHN
2, JANE
3, JOE
4, EMMA
```

RESULT:

Thus the program is executed successfully and output is verified.