

**HADOOP MAPREDUCE PROGRAM IN PYTHON**

\_\_\_\_\_\_\_\_\_\_\_\_**MAPPER**\_\_\_\_\_\_\_\_\_\_\_\_\_

1> make a file named mapper.py and paste below python code for mapper in it

**$ nano mapper.py**

#!/usr/bin/env python

import sys

for line in sys.stdin:

line = line.strip()

words = line.split()

for word in words:

print '%s\t%s' % (word, 1)

--------understanding above code---------------

#[ for line in sys.stdin: ] described that input comes from standard input (STDIN).

Standard input(stdin), is the source of input data for python ,

#[ line = line.strip() ] removes extra spaces

#[ words = line.split() ] splits line into words

#[ for word in words: ] increases counters

#[ print '%s\t%s' % (word, 1) ] will write result to (stdout) . This output will

input for reducer

2> Grant all permission to mapper.py

**$ chmod 777 /home/ubuntu/mapper.py**

\_\_\_\_\_\_\_\_\_\_\_\_**REDUCER**\_\_\_\_\_\_\_\_\_\_\_\_\_

3> make a file named reducer.py and paste below python code for reducer in it

**$ nano reducer.py**

#!/usr/bin/env python

from operator import itemgetter

import sys

current\_word = None

current\_count = 0

word = None

for line in sys.stdin:

line = line.strip()

word, count = line.split('\t', 1)

try:

count = int(count)

except ValueError:

continue

if current\_word == word:

current\_count += count

else:

if current\_word:

print '%s\t%s' % (current\_word, current\_count)

current\_count = count

current\_word = word

if current\_word == word:

print '%s\t%s' % (current\_word, current\_count)

----understanding above code----

#The code in reducer.py will read results of mapper.py through standard input so , output

of mapper.py and input of reducer.py must match .

#[ word, count = line.split('\t', 1) ] will parse input got from mapper

#[ try:

count = int(count)

except ValueError: ] will convert count which is in currently string format to int

because count is going to be a number , i.e int.

#The [continue] statement after the code will ignore the line if count was not the number , i.e int

#[ if current\_word == word:

current\_count += count

else:

if current\_word: ] here if works because hadoop sorts map output i.e word before it is passed to the reducer

#[ print '%s\t%s' % (current\_word, current\_count)

current\_count = count

current\_word = word] this will write result to standard output (STDOUT)

4> Grant all permission to reducer.py

**$ chmod 777 /home/ubuntu/reducer.py**

\_\_\_\_\_\_\_\_\_\_\_\_**RUNNING PYTHON CODE ON HADOOP**\_\_\_\_\_\_\_\_\_\_\_\_\_

5> first copy the files that has to be Processed from our local file system to Hadoop’s HDFS.

**$ hadoop fs -put <filename> <input>**

6> run hadoop streaming jar file which will allow python code on hadoop followed by mapper reducer input and output

**$ hadoop jar /usr/local/hadoop/contrib/streaming/hadoop-streaming-1.2.1.jar -file /home/ubuntu/mapper.py -mapper /home/ubuntu/mapper.py -file /home/ubuntu/reducer.py -reducer /home/ubuntu/reducer.py -input in -output out1**

----------Understanding above command-------------------

Here -file takes File/dir to be shipped in the Job jar file -input takes DFS input file for the Map step .

-mapper takes the streaming command to run map step . -reducer takes the streaming command to run reduce step