Exemplo MapReduce com Python: Mapper

([URL IMF Class](https://campusformacionnebrija.imf.com/general/prt/prg/leccgen/newud.php?codcurso=C7274C13&unidad=2&trozo=26&param=contenido))

Create a folder in which we will store the mapping and reduction files:

*mkdir /home/bigdata/ejemplosMapReduce*

*mkdir /home/bigdata/ejemplosMapReduce/python*

*cd /home/bigdata/ejemplosMapReduce/python*

Create the python mapping script – mapper.py:

*sudo nano mapper.py*

*#!/usr/bin/env python3*

*# file: mapper.py*

*import sys*

*# input comes from STDIN (standard input)*

*for line in sys.stdin:*

*# remove leading and trailing whitespace*

*line = line.strip()*

*# split the line into words*

*words = line.split()*

*# increase counters*

*for word in words:*

*# write the results to STDOUT (standard output);*

*# what we output here will be the input for the*

*# Reduce step, i.e. the input for reducer.py*

*# tab-delimited; the trivial word count is 1*

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

Assign the execution permissions with the following command:

*sudo chmod +x /home/bigdata/ejemplosMapReduce/python/mapper.py*

Create the python reduction script – reduction.py:

*sudo nano reducer.py*

*#!/usr/bin/env python3*

*# file: reducer.py*

*from operator import itemgetter*

*import sys*

*current\_word = None*

*current\_count = 0*

*word = None*

*# input comes from STDIN*

*for line in sys.stdin:*

*# remove leading and trailing whitespace*

*line = line.strip()*

*# parse the input we got from mapper.py*

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

*# convert count (currently a string) to int*

*try:*

*count = int(count)*

*except ValueError:*

*# count was not a number, so silently ignore/discard this line*

*continue*

*# this IF-switch only works because Hadoop sorts map output by key (here: word) before it is passed to the reducer*

*if current\_word == word:*

*current\_count += count*

*else:*

*if current\_word:*

*# write result to STDOUT*

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

*current\_count = count*

*current\_word = word*

*# do not forget to output the last word if needed!*

*if current\_word == word:*

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

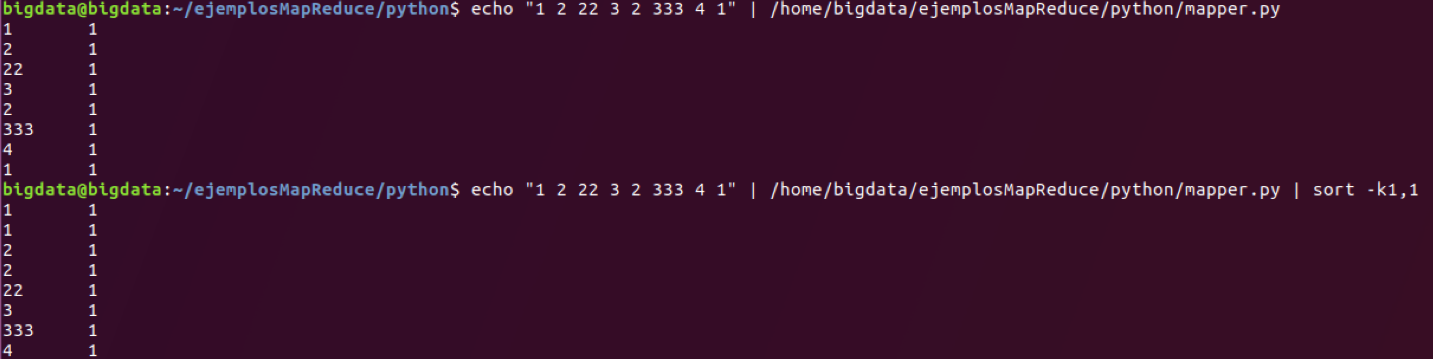
Assign the execution permissions with the following command:

*sudo chmod +x /home/bigdata/ejemplosMapReduce/python/reducer.py*

Check that they work with a simple example - concatenate sorting:

*echo "1 2 22 3 2 333 4 1" | /home/bigdata/ejemplosMapReduce/python/mapper.py*

*echo "1 2 22 3 2 333 4 1" | /home/bigdata/ejemplosMapReduce/python/mapper.py | sort -k1,1*



Creamos unos ficheros y los subimos a una nueva estructura en HDFS:

*mkdir /home/bigdata/ejemplosMapReduce/python/wc-in-local*

*echo "Hello World. Bye World." > /home/bigdata/ejemplosMapReduce/python/wc-in-local/a.txt*

*echo "Hello Hadoop Goodbye Hadoop" > /home/bigdata/ejemplosMapReduce/python/wc-in-local/b.txt*

Creamos la nueva carpeta en HDFS:

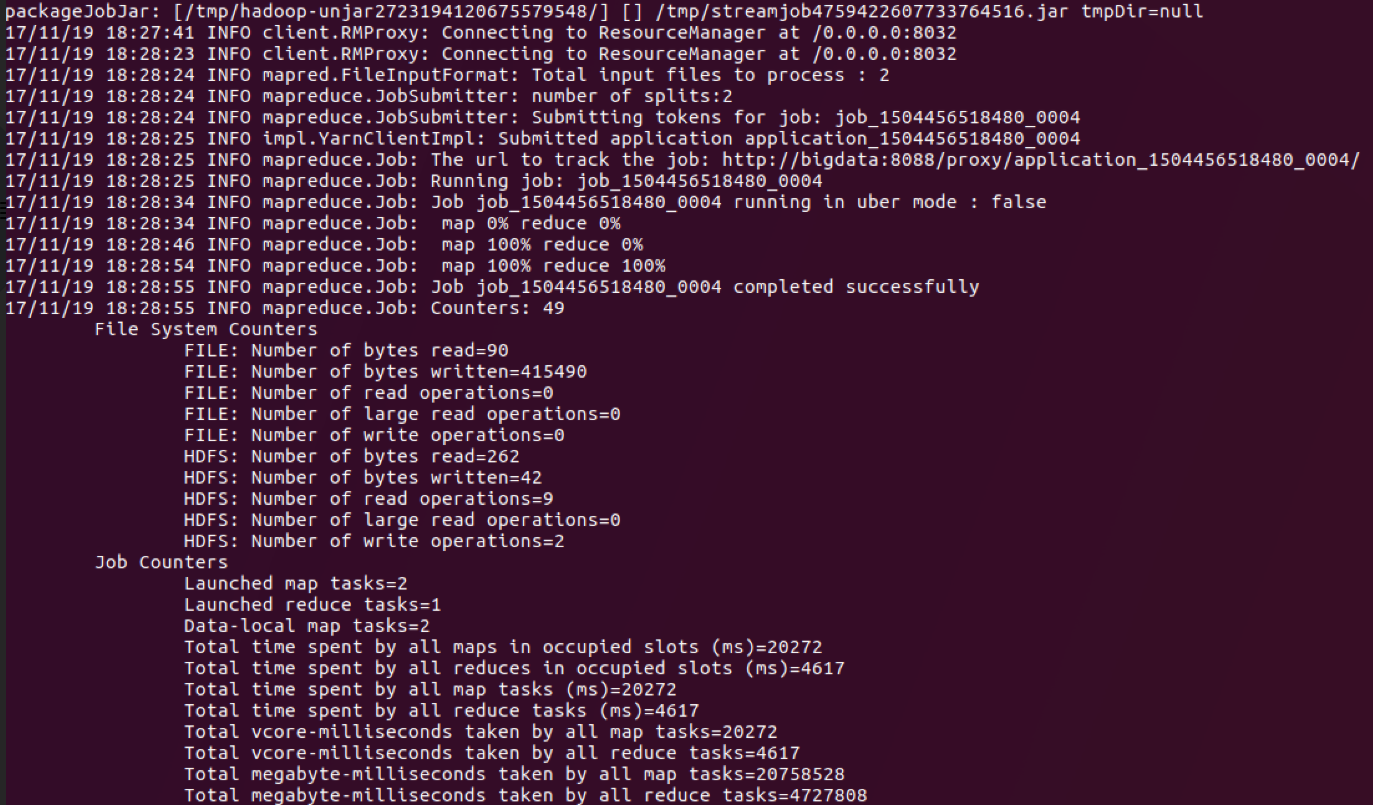
*hdfs dfs -mkdir -p /user/bigdata/wc-in-python*

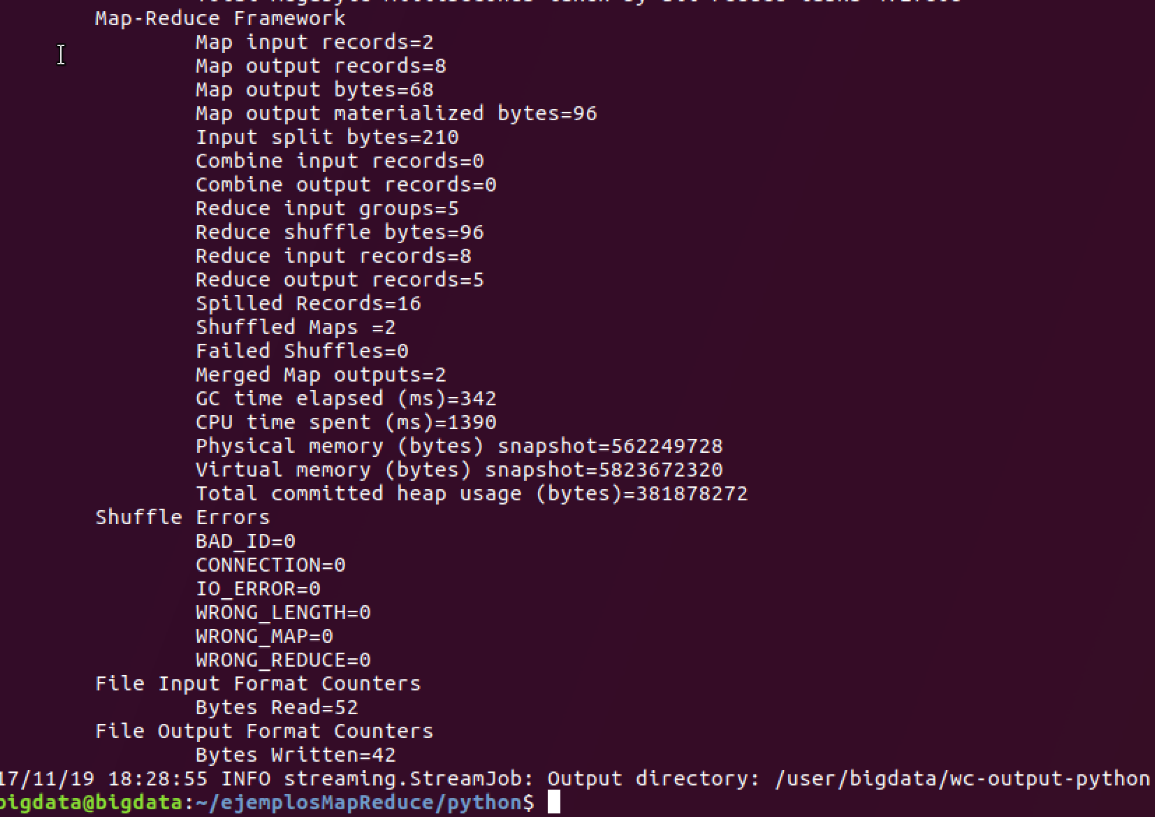
Subimos los ficheros:

*hdfs dfs -put /home/bigdata/ejemplosMapReduce/python/wc-in-local/\* /user/bigdata/wc-in-python*

And now we process them:

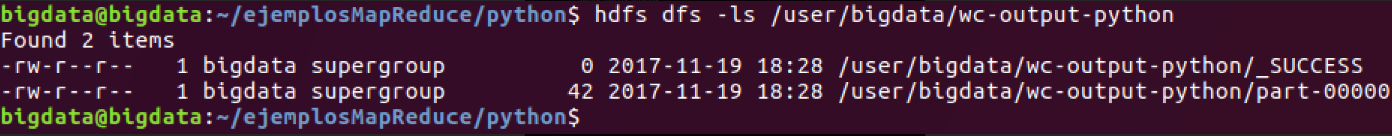
*hadoop jar /home/bigdata/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.8.0.jar -mapper /home/bigdata/ejemplosMapReduce/python/mapper.py -reducer /home/bigdata/ejemplosMapReduce/python/reducer.py -input /user/bigdata/wc-in-python/\* -output /user/bigdata/wc-output-python*





To verify if the files were generated:

*hdfs dfs -ls /user/bigdata/wc-output-python*



And see the results:

*hdfs dfs -cat /user/bigdata/wc-output-python/\**

