



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN

---

**NOMBRE DE ESTUDIANTE:** Díaz Padilla Danny Sebastián

**Laboratorio de:**

ANALÍTICA DE DATOS – BIG DATA

**Práctica Clase 20**

**Tema: Hadoop**

**Objetivos:**

- Instalar la máquina virtual de Cloudera con Hadoop.
- Crear un esquema MapReduce en Python.
- Ejecutar MapReduce sobre un archivo de texto con el libro del quijote de la mancha.

**Marco teórico:**

### **Hadoop**

Es una estructura de software de código abierto para almacenar datos y ejecutar aplicaciones en clústeres de hardware comercial. Proporciona almacenamiento masivo para cualquier tipo de datos, enorme poder de procesamiento y la capacidad de procesar tareas o trabajos concurrentes virtualmente ilimitados. [1]

### **Comandos**

Listar todo el contenido en el directorio especificado:

**`hdfs dfs -ls /user/cloudera`**

Crear directorio en un sitio específico.

**`hdfs dfs -mkdir /user/cloudera/input`**

Almacenar y poner un archivo local en el sistema de archivos Hadoop:

**`hdfs dfs -put quijote.txt /user/cloudera/input/`**

El siguiente commando utiliza una librería de map reduce llamada hadoop-streaming la cual permite realizar los trabajos de MapReduce y también con -input se agrega la entrada de los datos y con -output se establece el directorio de la salida del proceso.

**`hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar - input  
/user/cloudera/input -output /user/cloudera/output - mapper /quijote/mapper.py -  
reducer /quijote/reducer.py`**

Leer el contenido del archive resultante:

**`hdfs dfs cat /user/cloudera/output/part 00000 | head 1000`**



**ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

---

### Desarrollo de la práctica:

Para el primer paso es crear un directorio llamado “quijote” y crear un archivo bash que permitirá descargar el libro del quijote.

El script contiene la siguiente línea de código:

**curl <http://www.gutenberg.org/cache/epub/2000/pg2000.txt> -oquijote.txt**

Posteriormente le agregamos permisos de ejecución y lo ejecutamos.

```
cloudera@quickstart:~/Documents/quijote
File Edit View Search Terminal Help
[cloudera@quickstart Documents]$ mkdir quijote
[cloudera@quickstart Documents]$ cd quijote/
[cloudera@quickstart quijote]$ nano descarga.sh
[cloudera@quickstart quijote]$ chmod 777 descarga.sh
[cloudera@quickstart quijote]$ ./descarga.sh
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %             %           Dload  Upload  Total   Spent    Left   Speed
100 2147k  100 2147k    0     0  406k      0  0:00:05  0:00:05 --:--:--  460k
[cloudera@quickstart quijote]$ ls
descarga.sh  quijote.txt
[cloudera@quickstart quijote]$
```

En la anterior figura se puede apreciar la descarga del fichero.

Luego preparamos el entorno de hadoop creando un directorio de entrada y poniendo el archivo de texto con el libro de el Quijote en ese directorio.

Se crea ahora el script que realizará el mapeo a todas las palabras, está hecho en python y tiene la siguiente estructura.

```
cloudera@quickstart:~/Documents/quijote
File Edit View Search Terminal Help
[cloudera@quickstart quijote]$ hdfs dfs -ls /user/cloudera
[cloudera@quickstart quijote]$ ls
descarga.sh  quijote.txt
[cloudera@quickstart quijote]$ hdfs dfs -mkdir /user/cloudera/input
[cloudera@quickstart quijote]$ hdfs dfs -put quijote.txt /user/cloudera/input
[cloudera@quickstart quijote]$ nano mapper.py
bash: nano_mapper.py: command not found
[cloudera@quickstart quijote]$ nano mapper.py
[cloudera@quickstart quijote]$ cat mapper.py
#!/usr/bin/env python

import sys
for line in sys.stdin:
    line = line.strip()
    keys = line.split()
    for key in keys:
        value = 1
        print("%s\t%d" % (key, value))

[cloudera@quickstart quijote]$
[cloudera@quickstart quijote]$
[cloudera@quickstart quijote]$
[cloudera@quickstart quijote]$
[cloudera@quickstart quijote]$
```

Se imprime el valor de 1 por cada palabra encontrada, es decir, se le da un puntaje a las palabras recibidas.



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

Para el archivo que realizará la reducción se crea otro fichero .py llamado reducer el cual obtiene un puntaje total de todas las palabras previamente procesadas por la función de mapeo.

```
cloudera@quickstart:~/Documents/quijote
File Edit View Search Terminal Help
[cloudera@quickstart quijote]$ cat reducer.py
#!/usr/bin/env python

import sys

last_key = None
running_total = 0

for input_line in sys.stdin:
    input_line = input_line.strip()
    this_key, value = input_line.split("\t",1)
    value = int(value)

    if last_key == this_key:
        running_total += value
    else:
        if last_key:
            print( "%s\t%d" % (last_key, running_total))
            running_total = value
            last_key = this_key

if last_key == this_key:
    print( "%s\t%d" % (last_key, running_total))
[cloudera@quickstart quijote]$
```

Mediante el comando:

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar - input /user/cloudera/input -output
/user/cloudera/output - mapper /quijote/mapper.py -reducer /quijote/reducer.py
```

se intenta realizar este procedimiento.

Sin embargo, aunque la función de mapeo y reducción alcanzar el 100% ambas tareas fallan.

```
cloudera@quickstart:/
File Edit View Search Terminal Help
78944064156_0005
20/01/13 12:07:27 INFO mapreduce.Job: The url to track the job: http://quickstar
t.cloudera:8088/proxy/application_1578944064156_0005/
20/01/13 12:07:27 INFO mapreduce.Job: Running job: job_1578944064156_0005
20/01/13 12:08:05 INFO mapreduce.Job: Job job_1578944064156_0005 running in uber
mode : false
20/01/13 12:08:05 INFO mapreduce.Job: map 0% reduce 0%
20/01/13 12:08:49 INFO mapreduce.Job: map 67% reduce 0%
20/01/13 12:08:51 INFO mapreduce.Job: map 83% reduce 0%
20/01/13 12:08:52 INFO mapreduce.Job: map 100% reduce 0%
20/01/13 12:09:24 INFO mapreduce.Job: Task Id : attempt_1578944064156_0005_r_000
000_0, Status : FAILED
Error: java.lang.RuntimeException: PipeMapRed.waitOutputThreads(): subprocess fa
iled with code 1
    at org.apache.hadoop.streaming.PipeMapRed.waitOutputThreads(PipeMapRed.j
ava:325)
    at org.apache.hadoop.streaming.PipeMapRed.mapRedFinished(PipeMapRed.java
:538)
    at org.apache.hadoop.streaming.PipeReducer.close(PipeReducer.java:134)
    at org.apache.hadoop.io.IOUtils.cleanup(IOUtils.java:246)
    at org.apache.hadoop.mapred.ReduceTask.runOldReducer(ReduceTask.java:459
)
    at org.apache.hadoop.mapred.ReduceTask.run(ReduceTask.java:392)
    at org.apache.hadoop.mapred.YarnChild$2.run(YarnChild.java:164)
```



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

---

El problema puede deberse a falta de compatibilidad o agotamiento de recursos.

En una ejecución normal, el resultado se podría ver como en la siguiente figura

```
[cloudera@quickstart ~]$ sudo -i
[root@quickstart ~]# cd ../../
[root@quickstart /]# cd quijote
[root@quickstart quijote]# nano mapper.py
[root@quickstart quijote]# nano reducer.py
[root@quickstart quijote]# hdfs dfs -ls /user/cloudera/output
[root@quickstart quijote]# hdfs dfs -ls /user/cloudera/output4
Found 2 items
-rw-r--r--  1 root cloudera          0 2020-01-08 16:51 /user/cloudera/output
_SUCCESS
-rw-r--r--  1 root cloudera 4005150 2020-01-08 16:51 /user/cloudera/output
part-000000
```

Y para mostrar todos los resultados se debe utilizar el comando: `hdfs dfs -cat /user/cloudera/output/*`

El cual muestra un conteo de cada palabra

```
única  2
única  3
única  4
única  5
única  5
única, 1
única, 1
única. 1
única. 1
único  1
único  2
único  3
único  4
único  5
único  6
único  7
único  8
único  9
único 10
único 11
único 11
```



**ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

### Análisis de resultados:

El error generado parece ser ajeno a los archivos creados. Entrando al enlace proporcionado por el mismo “trabajo” se puede ver esa ejecución en tiempo real:

Sin embargo, no da una descripción específica del error y la línea de comandos solo señala líneas dentro de los scripts de hadoop.

Este problema puede ser causado por incompatibilidad o problemas de recursos ya que la máquina virtual ejecutada solicita 4GB de RAM y la máquina utilizada apenas cumple con el requisito

### Conclusiones y recomendaciones:

- Se logró instalar la máquina virtual de Cloudera con Hadoop.
  - Se creó un esquema MapReduce en Python.
  - Se ejecutó MapReduce sobre un archivo de texto con el libro del quijote de la mancha, pero no se pudo cumplir la operación.
  - El procesamiento distribuido otorgado por Hadoop permite aumentar la velocidad a la que se trata una cantidad gigante de datos.
  - Hadoop proporciona un tratamiento del problema utilizando multihilos y todos los núcleos físicos disponibles en la máquina.
- 
- Se recomienda utilizar una computadora con un buen procesador y una cantidad de RAM mayor o igual a 8GB.
  - Para evitar problemas de permisos se recomienda ejecutar los comandos en un directorio fuera de root.

### Bibliografía

[1]"¿Qué es Hadoop?", *Sas.com*. [Online]. Available: [https://www.sas.com/es\\_pe/insights/big-data/hadoop.html](https://www.sas.com/es_pe/insights/big-data/hadoop.html). [Accessed: 13- Jan- 2020].