

Hadoop

x

Jupyter

Big Data Aplicado

Alejandro Cruz Aguilar

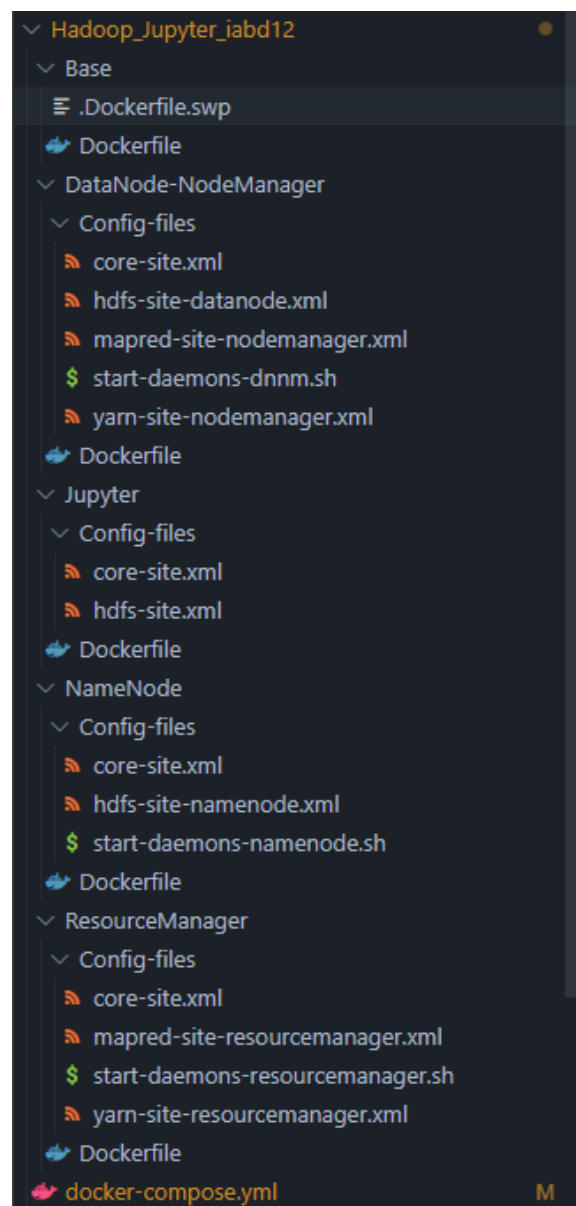
Índice

Preparación del Entorno	2
Uso de Hadoop mediante en consola.....	4
Creación de los scripts para contar	5
Lanzamiento de los Scripts	7
Lanzamiento de scripts en Jupyter	8
Descargar el Jupyter Notebook.....	10

Preparación del Entorno

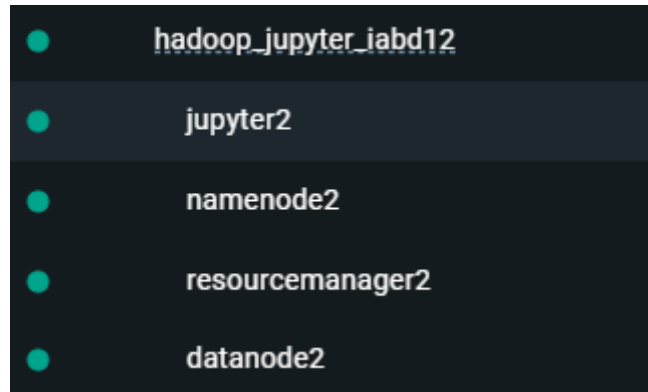
Antes de crear el cluster, necesitaremos lo siguiente:

- Una imagen BASE, de la cual el resto de los contenedores van a funcionar. Este debe tener instalado el HADOOP.
- Una imagen de JUPYTER, el cual tendrá instalada las librerías básicas, como Python, PIP...
- Una imagen de DATANODE; NAMENODE; RESOURCEMANAGER. Estos, para poder subir el archivo mediante HADOOP.
- Por último, un docker-compose, el cual se encargará de enlazar todos estos contenedores.



Uso de Hadoop mediante en consola

Ahora que hemos creado el cluster con todo lo que necesitamos, (jupyter, namenode, datanode, resourcemanager), podemos continuar subiendo el archivo “el_quijote.txt” a hadoop.



La sintaxis del comando es “hdfs dfs -put <ruta_archivo> <ruta_hadoop>”

```
$ hdfs dfs -put el_quijote.txt /user/hdadmin
$
```

Ahora podemos comprobar desde el mismo namenode, que el archivo a sido subido sin ningun problema.

Browse Directory

/user/hdadmin

Go!

Show

25

entries

Search:

<input type="checkbox"/>	<div><div></div></div> Permission	<div><div></div></div> Owner	<div><div></div></div> Group	<div><div></div></div> Size	<div><div></div></div> Last Modified	<div><div></div></div> Replication	<div><div></div></div> Block Size	<div><div></div></div> Name	<div><div></div></div>
<input type="checkbox"/>	-rw-r--r--	hdadmin	supergroup	1.01 MB	Nov 29 17:20	3	128 MB	el_quijote.txt	<div><div></div></div>

Showing 1 to 1 of 1 entries

Previous

1

Next

Hadoop, 2023.

Tambien, pro si queremos asegurarnos, podemos comprobarlo mediante la consola.

Con la sintaxis “!hadoop fs -ls /user/hdadmin”, nos muestra el archivo y sus permisos.

```
$ !hadoop fs -ls /user/hdadmin
Found 1 items
-rw-r--r--  3 hdadmin supergroup  1060259 2024-11-29 16:20 /user/hdadmin/el_quijote.txt
$
```

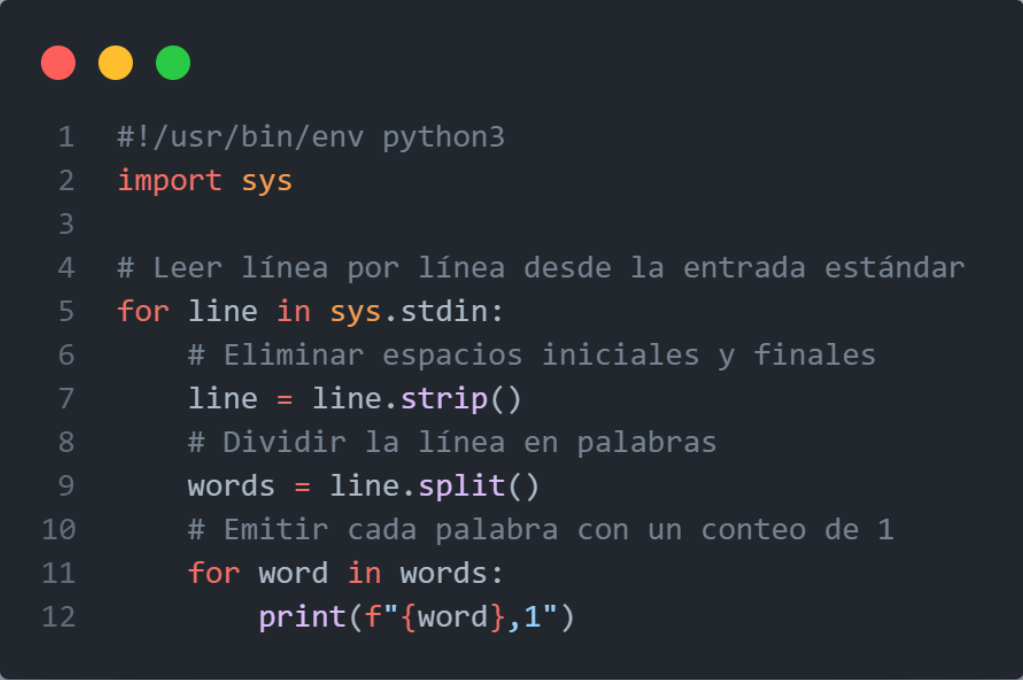
Creación de los scripts para contar

Ya que hemos subido el archivo a hadoop, tendremos que contar cuanta cantidad hay de cada palabra. Para ello, necesitaremos un mapreduce, lo cual lo podemos dividir en 2 partes, el mapper, y el reducer.

El código del mapper es el siguiente.

```
1  #!/usr/bin/env python3
2  import sys
3
4  current_word = None
5  current_count = 0
6  word = None
7
8  # Leer línea por línea desde la entrada estándar
9  for line in sys.stdin:
10     # Eliminar espacios iniciales y finales
11     line = line.strip()
12     # Dividir la línea en palabra y valor
13     word, count = line.split(",", 1)
14     try:
15         count = int(count)
16     except ValueError:
17         # Ignorar líneas mal formateadas
18         continue
19
20     # Sumar conteos si la palabra es la misma
21     if current_word == word:
22         current_count += count
23     else:
24         # Si es una nueva palabra, imprimir la anterior
25         if current_word:
26             print(f"{current_word},{current_count}")
27         current_word = word
28         current_count = count
29
30 # Imprimir la última palabra si existe
31 if current_word == word:
32     print(f"{current_word},{current_count}")
```

Y el código del reducer es este:

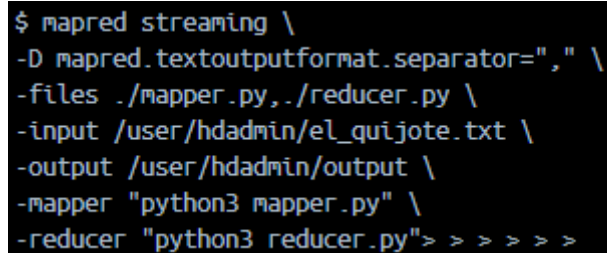


```
1  #!/usr/bin/env python3
2  import sys
3
4  # Leer línea por línea desde la entrada estándar
5  for line in sys.stdin:
6      # Eliminar espacios iniciales y finales
7      line = line.strip()
8      # Dividir la línea en palabras
9      words = line.split()
10     # Emitir cada palabra con un conteo de 1
11     for word in words:
12         print(f"{word},1")
```

Lanzamiento de los Scripts

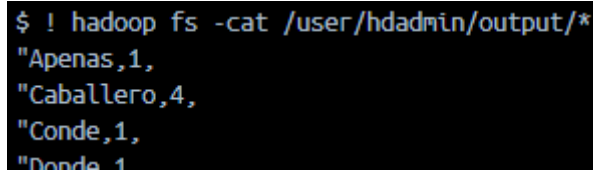
Ahora, con estos scripts creados, mediante la consola de docker. Para ello, tendremos que lanzar el siguiente comando en la consola:

```
mapred streaming \  
-D mapred.textoutputformat.separator="," \  
-files ./mapper.py,./reducer.py \  
-input /user/hdadmin/el_quijote.txt \  
-output /user/hdadmin/output \  
-mapper "python3 mapper.py" \  
-reducer "python3 reducer.py"
```



```
$ mapred streaming \  
-D mapred.textoutputformat.separator="," \  
-files ./mapper.py,./reducer.py \  
-input /user/hdadmin/el_quijote.txt \  
-output /user/hdadmin/output \  
-mapper "python3 mapper.py" \  
-reducer "python3 reducer.py"> > > > >
```

Este creara un directorio en la “/user/hdadmin” llamado “output”. Este directorio contiene el archivo el cual guarda la cuenta de cada palabra contada y su cantidad en total.



```
$ ! hadoop fs -cat /user/hdadmin/output/*  
"Apenas,1,  
"Caballero,4,  
"Conde,1,  
"Donde,1
```

Lanzamiento de scripts en Jupyter

Ya hemos comprobado que el contenedor de jupyter, puede lanzar e interactuar con los archivos subidos en hadoop.

Ahora, tenemos que comprobar que desde jupyter notebooks, también funciona.

Primero, tendremos que comprobar donde está nuestro mapper y reducer. Aparte, también comprobamos en que directorio está actualmente jupyter.

```
[1]: !ls /opt/bd/jupyter-data/jupyter/Tema_1
    Plantilla_Tarea1_Evaluable_RA1_BDA.ipynb  el_quijote.txt  mapper.py  reducer.py

[2]: pwd

[2]: '/opt/bd/hadoop-3.3.6'

[3]: !hdfs dfs -rm -r /user/hdadmin/output
    Deleted /user/hdadmin/output
```

Ya tenemos el directorio, así que habrá que modificar ligeramente el código para que detecte los archivos Python y ejecute. (Esto depende de tus rutas)

```
[4]: !mapred streaming \
    -D mapred.textoutputformat.separator="," \
    -files /opt/bd/jupyter-data/jupyter/Tema_1/mapper.py,/opt/bd/jupyter-data/jupyter/Tema_1/reducer.py \
    -input /user/hdadmin/el_quijote.txt \
    -output /user/hdadmin/output \
    -mapper "python3 /opt/bd/jupyter-data/jupyter/Tema_1/mapper.py" \
    -reducer "python3 /opt/bd/jupyter-data/jupyter/Tema_1/reducer.py"

Reducer Output: 1000000
Spilled Records=374036
Shuffled Maps =1
Failed Shuffles=0
Merged Map outputs=1
GC time elapsed (ms)=0
Total committed heap usage (bytes)=632291328
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=1060259
File Output Format Counters
Bytes Written=209755
2024-12-02 18:07:43,338 INFO streaming.StreamJob: Output directory: /user/hdadmin/output
```


Ya creado el fichero, podemos leerlo y comprobar que los ha contado correctamente.

```
[5]: !hadoop fs -cat /user/hdadmin/output/*
```

```
-Cuando,1,  
-Cuatro,1,  
-Dadme,1,  
-De,3,  
-Debe,2,  
-Debes,1,  
-Del,1,  
-Desa,1,  
-Deso,1,  
-Después,2,  
-Deténgome,1,  
-Déjeme,2,  
-Déjeseme,1,  
-Di,1,  
-Dice,1,  
-Dichosa,1,  
-Digo,7,  
-Dilas,1,  
-Digolo,2,
```

Descargar el Jupyter Notebook

Ya con el notebook creado, tendremos que bajarlo para tenerlo en la maquina anfitriona. Primero, habrá que localizarlo.






```
$ pwd
/opt/bd/hadoop-3.3.6
$ ls
LICENSE-binary LICENSE.txt NOTICE-binary NOTICE.txt PruebasElQuijote.ipynb README.txt
$
```

Ya con el notebook localizado, tendremos que lanzar el siguiente comando desde la consola de la maquina anfitriona, ya que si lo hacemos desde la virtual este no funciona;

```
docker cp <contenedor>:<ruta> / <archivo> <rutaAnfitriona>
```

```
PS C:\Users\IABD12> docker cp jupyter2:/opt/bd/hadoop-3.3.6/PruebasElQuijote.ipynb C:/Users/IABD12/Documents/GitHub/ClaseIABD/David/Jupyter/Tema_1
Successfully copied 2.05kB to C:\Users\IABD12\Documents\GitHub\ClaseIABD\David\Jupyter\Tema_1
```

Ya que nos dice que se ha descargado satisfactoriamente, podemos comprobar que, en el directorio anfitrión, se ha descargado. Y ya con ello habríamos terminado.

Nombre	Fecha de modificación	Tipo	Tamaño
 el_quijote.txt	04/11/2024 19:26	Documento de te...	1.036 KB
 mapper.py	29/11/2024 17:24	Archivo de origen ...	1 KB
 Plantilla_Tarea1_Evaluable_RA1_BDA.ipynb	29/11/2024 17:04	Archivo de origen ...	11 KB
 PruebasElQuijote.ipynb	02/12/2024 19:39	Archivo de origen ...	1 KB
 reducer.py	29/11/2024 17:24	Archivo de origen ...	1 KB