

Reto 3.2: HIVE y SparkSQL, GESTIÓN DE DATOS VIA SQL

1. Conectarse a la consola de HUE via AWS EMR, Conectate al nodo mates mediante SSH `ssh -i bigdata-key-pair.pem hadoop@ec2-100-26-198-124.compute-1.amazonaws.com`
2. EJECUTA `pyspark` (esta es una variable que ya existe)

```
[hadoop@ip-10-0-6-239 ~]$ pyspark
Python 3.7.16 (default, Aug 30 2023, 20:37:53)
[GCC 7.3.1 20180712 (Red Hat 7.3.1-15)] on linux
Type "help", "copyright", "credits" or "license()" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/11/25 02:59:33 WARN HiveConf: HiveConf of name hive.server2.thrift.url does not exist
23/11/25 02:59:36 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.

Welcome to
  ____              _
 / ___|  _ \   ___| | | |
 \___ \ |_) | / __| |_| |
  ___) ||  __/|___|  __/
 /____/|___||___|_|_|_|

version 3.4.1-amzn-2

Using Python version 3.7.16 (default, Aug 30 2023 20:37:53)
Spark context Web UI available at http://ip-10-0-6-239.ec2.internal:4040
Spark context available as 'sc' (master = yarn, app id = application_1700874674194_0010).
SparkSession available as 'spark'.
>>>
```

3. Una vez allí puedes ejecutar los siguientes comandos, asegúrate de modificarlos a como están en tus archivos según el reto 3.1

```
>>> files_rdd = sc.textFile("hdfs:///datasets/gutenberg-small/*.txt")
>>> files_rdd = sc.textFile("s3://st0263datasets/gutenberg-small/*.txt")
>>> wc_unsort = files_rdd.flatMap(lambda line: line.split()).map(lambda word: (word,
1)).reduceByKey(lambda a, b: a + b)
>>> wc = wc_unsort.sortBy(lambda a: -a[1])
>>> for tupla in wc.take(10):
>>>     print(tupla)
>>> wc.saveAsTextFile("hdfs:///tmp/wcout1")
```

4. También se puede realizar empleando un archivo .py para ello vamos a crearlo

```
$ nano wc-pyspark.py
```

pegamos el siguiente contenido:

```
from pyspark.sql import SparkSession
```

puede no necesitar instanciar las variables spark y sc si esta ejecutando en AWS EMR:

```
spark = SparkSession.builder.appName("WordCount").getOrCreate()
```

```
sc = spark.sparkContext
```

```
files_rdd = sc.textFile("s3://st0263datasets/gutenberg-small/*.txt")
#files_rdd = sc.textFile("hdfs:///datasets/gutenberg-small/*.txt")
wc_unsort = files_rdd.flatMap(lambda line: line.split()).map(lambda word: (word,
1)).reduceByKey(lambda a, b: a + b)
wc = wc_unsort.sortBy(lambda a: -a[1])
wc.coalesce(1).saveAsTextFile("hdfs:///tmp/wcout1")
```

y guardamos el archivo y lo ejecutamos con:

```
$ spark-submit --master yarn --deploy-mode cluster wc-pyspark.py
```

