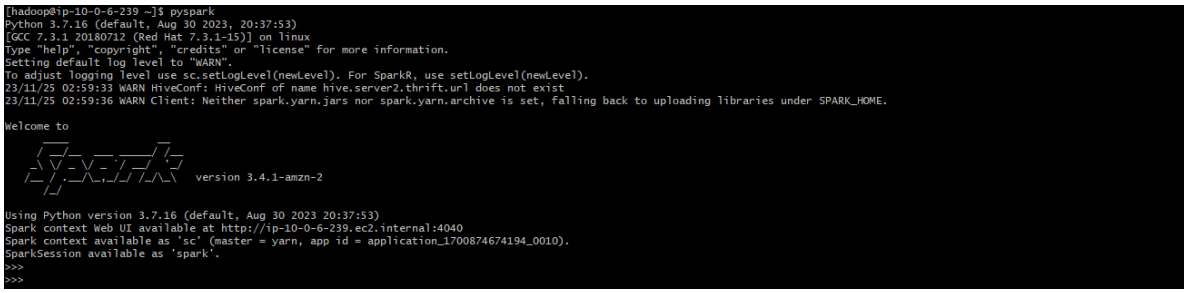
Universidad Eafit

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)



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. Tambien 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**