(RABDA.2 / CEBDA.2b / 2p)

Vuelve a contar las palabras que tiene El Quijote, pero haciendo usos de los scripts Python, teniendo en cuenta que el proceso de mapeo va a limpiar las palabras de signos ortográficos (quitar puntos, comas, paréntesis) y en el reducer vamos a considerar que las palabras en mayúsculas y minúsculas son la misma palabra.

Tip: para la limpieza, puedes utilizar el método de string translate de manera que elimine las string.punctuation.

Debes ejecutar ambos script como procesos MapReduce mediante Hadoop Streaming y comprobar en HDFS el archivo que se ha creado.

Este ejercicio lo tengo hecho en clase, voy a utilizar el código para que lo puedas ver y si tienes alguna sugerencia estaré encantado de conocerla (imparto el módulo de Big Data Aplicado).

Utilizo la librería nltk así elimino signos de puntuación y artículos, preposiciones....

Estoy con un Mac M1, utilizo como sistema de virtualización un software llamado UTM, gracias al cual puedo emular una debian 64bits pero me va muy mal.

Lanzar el mapreduce a través de hadoop son unos 10-15 minutos...



```
hadoop@debianh:~/quijote$ cat mapper.py
#!/usr/bin/env python3
"""mapper.py"""
import sys
import string
import nltk
nltk.download('stopwords',quiet=True)
from nltk.corpus import stopwords
stop_words = set(stopwords.words('spanish'))
punctuations = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''
# 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:
        word = word.translate(str.maketrans('','',string.punctuation)).lower()
        if word not in stop_words:
            print('%s\t%s' % (word, 1))
         else:
            print('Eliminada "%s"' % (word) )
#### Instalar nltk
#### python3.11 -m pip install nltk
#### apt install python3-nltk
```

Reducer

En mi caso para el reducer no utilizo diccionario ya que Hadoop después del mapper hace un sort y ya no es necesario, por lo que cuento las palabras que van saliendo línea a línea (ya que están ordenadas).

Para ejecutarlo en bash emulo el sort de Hadoop:

```
$ cat quijote.txt | python3 mapper.py | sort | python3 reducer.py > result.txt
```

```
hadoop@debianh:~/quijote$ tail result.txt
zoroástrica 1
zorra 1
zorras 1
zorruna 1
zuecos 1
zulema 1
zumban 1
zurdo 2
zurrón 1
zuzaban 1
```

```
#!/usr/bin/env python3
"""reducer.py""
import sys
word = None
current_word = None
current_count = 0
# input comes from STDIN (standard input)
for line in sys.stdin:
        data = line.strip().split("\t")
        if len(data) != 2:
          continue
        word, count = data
          count = int(count)
        except ValueError:
          continue
        if current_word == word:
           current_count += count
        else:
           if current_word: # Primera iteración es None
  print ('%s\t%s' % (current_word, current_count))
           current_word = word
          current_count = 1
if current_word == word:
  print ('%s\t%s' % (current_word, current_count))
```

Ejecución a través de Hadoop

hdfs dfs -put quijote.txt /

\$ hadoop jar /opt/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.6.jar -file mapper.py -mapper mapper.py -file reducer.py -reducer reducer.py -input /quijote.txt -output /quijote_salidaX

```
2024-04-03 22:43:33,229 INFO mapred.FileInputFormat: Total input files to process: 1
2024-04-03 22:43:33,971 INFO mapreduce.JobSubmitter: number of splits:2
2024-04-03 22:43:37,658 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1712175529914_0002
2024-04-03 22:43:39,648 INFO conf.Configuration: resource-types.xml not found
2024-04-03 22:43:39,652 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'
2024-04-03 22:43:39,652 INFO impl.YarnClientImpl: Submitted application application_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizerion_fizer
```

(RABDA.4 / CEBDA.4a / 1p) Entra en Hadoop UI y en YARN, y visualiza los procesos que se han ejecutado en las actividades 1 y 2, comprobando la configuración tanto en Hadoop UI como en YARN así como la ejecución de los jobs arrancando el Job History Server.



luster

odes ode Labels pplications

cheduler

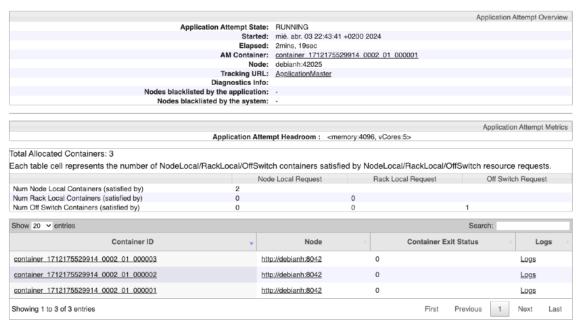
ools

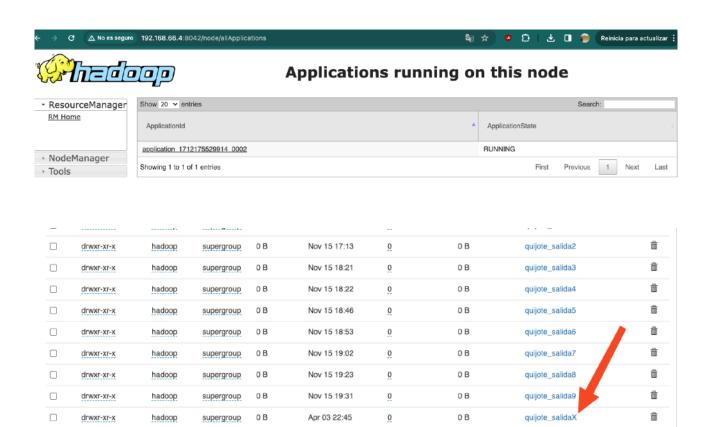
NEW SAVING SUBMITTED ACCEPTED RUNNING FINISHED FAILED

bout

Application Attempt appattempt_1712175529914_0002_000001

Logged in as: dr.who





(RABDA.3 / CEBDA.3a y CEBDA.3c / 0.5p)

En este ejercicio vamos a practicar los comandos básicos de HDFS. Una vez arrancado Hadoop:

Ejecuto alguno con time para que puedas ver la velocidad de la MV...

Crea la carpeta /user/iabd/ejercicios.

time hdfs dfs -mkdir -p /user/iabd/ejercicios

```
hadoop@debianh:~/quijote$ time hdfs dfs -mkdir -p /user/iabd/ejercicios

real 0m23,189s
user 0m20,083s
```

Sube el archivo el_quijote.txt a la carpeta creada.

```
hadoop@debianh:~/quijote$ time hdfs dfs -put el_quijote.txt /user/iabd/ejercicios/
real 0m26,007s
user 0m21,298s
sys 0m1,630s
```

Crea una copia en HDFS y llámala el_quijote2.txt.

hdfs dfs -cp /user/iabd/ejercicios/el_quijote.txt /user/iabd/ejercicios/el_quijote2.txt

Recupera el principio del fichero el quijote2.txt.

hdfs dfs -head /user/iabd/ejercicios/el_quijote2.txt

Renombra el_quijote2.txt a el_quijote_copia.txt.

hdfs dfs -mv /user/iabd/ejercicios/el_quijote2.txt /user/iabd/ejercicios/el_quijote_copia.txt

Descarga en local el quijote copia.txt con su código CRC.

hdfs dfs -get -crc /user/iabd/ejercicios/el quijote copia.txt

Adjunta una captura desde el interfaz web donde se vean ambos archivos.



Vuelve al terminal y elimina la carpeta con los archivos contenidos mediante un único comando.

hdfs dfs -rm -rf /user/iabd/ejercicios