

Tarea evaluable

CE_5075 7.1

Big data aplicado



APARTADO 2

Crea un cuaderno de Jupyter en el nodo con JupyterLab del clúster que has creado y escribe el código con PySpark y RDDs (no puedes usar DataFrames de Spark SQL) para contar las ocurrencias de todas las palabras que aparecen en *El Quijote* que empiezan y terminan con vocal (mayúscula o minúscula).

Dejaré el cuaderno de Jupyter que contiene el código ejecutado entre los archivos de la entrega. Pero aún así aquí muestro algunas capturas del cuaderno.

```
from pyspark import SparkContext
import re

# Inicializar SparkContext
sc = SparkContext('spark://spark-master:7077', 'ContarPalabrasQuijote')
sc
```

/usr/local/lib/python3.9/dist-packages/pyspark/bin/load-spark-env.sh: line 68: ps: command not found
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
25/03/30 17:46:06 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

SparkContext

Spark UI

Version	v3.5.5
Master	spark://spark-master:7077
AppName	ContarPalabrasQuijote

```
# Carga el archivo como RDD
rdd = sc.textFile('quijote.txt')
# Expresión regular para palabras que comienzan y terminan con vocal (no distingue mayúsculas de minúsculas)
vowel_pattern = r'(?i)^[aeiouáéíóú].*[aeiouáéíóú]$'

word_counts = (
    rdd.flatMap(lambda line: re.findall(r'\b\w+\b', line)) # Pasa todas las palabras a una lista
        .filter(lambda word: re.match(vowel_pattern, word)) # Filtra las palabras que coinciden con la regex
        .map(lambda word: (word.lower(), 1)) # Pasa a minúsculas y establece el contador a 1
        .reduceByKey(lambda x, y: x + y) # Reduce por palabra (clave) sumando los valores
)

# Muestra el contador por palabra
for word, count in word_counts.collect():
    print(f'{word}: {count}')
```

[Stage 1:> (0 + 2) / 2]

adulacio: 2
ora: 252
asi: 576
esto: 497
espanto: 12
enviado: 7
encuentro: 7
este: 394
alando: 4
alguna: 210
acabo: 36
osaba: 6
aquella: 219
arabada: 18


Y a continuación adjunto las capturas de pantalla del nodo master y de uno de los worker para comprobar que la aplicación está en funcionamiento.

Untitled1.ipynb - Jupyter | x

Spark Master at spark://s/ x

+

localhost:8080

 3.5.5

Spark Master at spark://spark-master:7077

URL: spark://spark-master:7077

Alive Workers: 3

Cores in use: 12 Total, 12 Used

Memory in use: 14.3 GiB Total, 3.0 GiB Used

Resources in use:

Applications: 1 [Running](#), 0 [Completed](#)

Drivers: 0 Running, 0 Completed

Status: ALIVE

Workers (3)

Worker Id	Address	State	Cores	Memory	Resources
worker-20250330162511-172.18.0.4-37507	172.18.0.4:37507	ALIVE	4 (4 Used)	4.8 GiB (1024.0 MiB Used)	
worker-20250330162511-172.18.0.5-43539	172.18.0.5:43539	ALIVE	4 (4 Used)	4.8 GiB (1024.0 MiB Used)	
worker-20250330162511-172.18.0.6-39511	172.18.0.6:39511	ALIVE	4 (4 Used)	4.8 GiB (1024.0 MiB Used)	

Running Applications (1)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20250330174607-0000 (kill)	ContarPalabrasQuijote	12	1024.0 MiB		2025/03/30 17:46:07	root	RUNNING	6 s

Completed Applications (0)


Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

apartado_2_c... - Jupyter | x

Spark Worker at 172.18.0 x

+

No seguro 172.18.0.4:8081

 3.5.5

Spark Worker at 172.18.0.4:37507

ID: worker-20250330162511-172.18.0.4-37507

Master URL: spark://spark-master:7077

Cores: 4 (4 Used)

Memory: 4.8 GiB (1024.0 MiB Used)

Resources:

[Back to Master](#)

Running Executors (1)

ExecutorID	State	Cores	Memory	Resources	Job Details	Logs
1	RUNNING	4	1024.0 MiB		ID: app-20250330174607-0000 Name: ContarPalabrasQuijote User: root	stdout stderr