

## Modo 1 – Análise de Contagem de Palavras com Spark em Jupyter Lab

Baixando a imagem do Docker:

```
docker pull jupyter/all-spark-notebook
```

Executando um contêiner Docker com a imagem:

```
docker run -it -p 8888:8888 jupyter/all-spark-notebook
```

Copie a URL com o token para acessar o Jupyter Lab. Você verá a URL no terminal após a inicialização do Jupyter Lab:

```
http://127.0.0.1:8888/lab?token=50fd79522dfecf123bb66ce1f83cbef6bae2d4f09e0de801
```

Adicionei o README.md com a função Upload Files do Jupyter Lab

Dentro do jupyter lab Inicie o PySpark:

```
pyspark
```

Carregue o conteúdo do arquivo "README.md" em um RDD:

```
rdd = sc.textFile("README.md")
```

Contagem de palavras no RDD:

```
word_counts = rdd.flatMap(lambda line: line.split(" ")).countByValue()
```


Formatando a contagem de palavras:

```
formatted_word_counts = [(word, count) for word, count in word_counts.items()]
```

Imprimindo as palavras e suas contagens:

```
for word, count in formatted_word_counts:
```

```
    print("{}: {}".format(word, count))
```



Filter files by name

Name	Last Modified
work	7 days ago
README.md	19 minutes ago

```

(base) jovyan@93799fd79f36:~ - X +
README.md x
jovyan@93799fd79f36:~$ pyspark
Python 3.11.5 | packaged by conda-forge | (main, Aug 27 2023, 03:34:09) [GCC 12.3.0] 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).
23/09/25 23:42:04 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Welcome to
      ____
     / __ \_____/ 
    / /_/ //___/   version 3.5.0
   /_/_/ 

Using Python version 3.11.5 (main, Aug 27 2023 03:34:09)
Spark context Web UI available at http://93799fd79f36:4043
Spark context available as 'sc' (master = local[*], app id = local-1695685325460).
SparkSession available as 'spark'.
>>> rdd = sc.textFile("README.md")
>>> word_counts = rdd.flatMap(lambda line: line.split(" ")).countByValue()
>>> formatted_word_counts = [(word, count) for word, count in word_counts.items()]

>>> for word, count in formatted_word_counts:
...     print("{}: {}".format(word, count))
...
#: 3
Apresentação: 1
■ 1
: 22
Olá!: 1
Meu: 1
nome: 1
é: 1
Marlos: 1
Igor: 1
Paulino: 1
Barros: 1
e: 29
sou: 1
natural: 1
da: 9
cidade: 1
de: 42
Águas: 1
Belas.: 1
Atualmente,: 1
estou: 3
cursando: 1
o: 11
quarto: 1
semestre: 1
no: 3
Centro: 1
Universitário: 1
Maurício: 1
Nassau.: 1
Tenho: 1
experiência: 1
nas: 2
áreas: 1
tecnologia:, 1
com: 5
conhecimentos: 2
em: 12
Java,: 1
Spring,: 1
Microserviços,: 1
SQL: 2
AWS.: 3
Sprints: 1
🐼 1
--: 24
[👉] 7
Cultura: 1
Ágil: 1
Segurança: 1
[CulturaX20AgilX20%20Segurança]:: 1
Conclui: 7
curso: 7
"Segurança": 1
Aplicações: 1
Web,.: 1
qual: 2
me: 7
proporcionou: 1
uma: 6
comprensão: 1
abrangente: 1
dos: 3
principais: 1
conceitos: 3
técnicas: 2
segurança: 1
para: 7
desenvolvimento: 2
aplicações: 1

```

OU

## Modo 2 – Análise de Contagem de Palavras com Spark no Terminal

Iniciando o PySpark no terminal:

```
pyspark
```

Carregue o conteúdo do arquivo "README.md" em um RDD:

```
rdd = sc.textFile("/home/marlos-igor/Área de trabalho/Marlos-Igor-Compass.UOL/README.md")
```

Contagem de palavras no RDD:

```
word_counts = rdd.flatMap(lambda line: line.split(" ")).countByValue()
```

Formatando a contagem de palavras:

```
formatted_word_counts = [(word, count) for word, count in word_counts.items()]
```

Imprimindo as palavras e suas contagens:

```
for word, count in formatted_word_counts:
```

```
    print("{}: {}".format(word, count))
```

```
marlos-igor@marlos:~$ pyspark
Python 3.11.4 (main, Jun  9 2023, 07:59:55) [GCC 12.3.0] 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).
23/09/25 21:40:28 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Welcome to

  ____      _
 / ___|  _ \| | | |
 \___ \ |_) | | | |
  ___) | | | | | | |
 /___ \_| .|_|_|_|_|
        |_|_|_|_|_|_|_| version 3.5.0

Using Python version 3.11.4 (main, Jun  9 2023 07:59:55)
Spark context Web UI available at http://marlos:4040
Spark context available as 'sc' (master = local[*], app id = local-1695688828943).
SparkSession available as 'spark'.
>>> 23/09/25 21:40:41 WARN GarbageCollectionMetrics: To enable non-built-in garbage collector(s) List(G1 Concurrent GC), users should configure it(them) to sp
ack eventing or Metrics usingGenerationGarbageCollectors or spark.eventing.orMetrics.addGenerationGarbageCollectors
```

```
rd = sc.textFile("/home/marlos-igor/Área de trabalho/Marlos-Igor-Compass.UOL/README.md")
>>> rdd = sc.textFile("/home/marlos-igor/Área de trabalho/Marlos-Igor-Compass.UOL/README.md")
>>> word_counts = rdd.flatMap(lambda line: line.split(" ")).countByValue()
>>> formatted_word_counts = [(word, count) for word, count in word_counts.items()]
>>> for word, count in formatted_word_counts:
...     print("{}: {}".format(word, count))
...
# 3
Apresentação: 1
: 1
: 22
Olá!: 1
Meu: 1
nome: 1
é: 1
Marlos: 1
Igor: 1
Paulino: 1
Barros: 1
é: 37
sou: 1
natural: 1
da: 10
cidade: 1
de: 53
Águas: 1
Belas.: 1
Atualmente,: 1
estou: 3
curstando: 1
o: 13
quarto: 1
semestre: 1
no: 4
Centro: 1
Universitário: 1
Maurício: 1
Nassau.: 1
Tenho: 1
experiência: 1
nas: 2
áreas: 1
tecnologia,: 1
com: 9
conhecimentos: 2
em: 18
Java,: 1
Spring,: 1
Microserviços,: 1
SQL: 2
AWS.: 3
Sprints: 1
👨‍💻: 1
.: 24
[☑️]: 8
Cultura: 1
Agil: 1
Segurança: 1
☑️(Cultura%20Agil%20e%20Segurança):: 1
Conclui: 8
curso: 9
"Segurança: 1
Aplicações: 1
Web": 1
qual: 2
me: 11
proporcionou: 1
uma: 7
compreensão: 1
abrangente: 1
dos: 4
principais: 1
conceitos: 3
técnicas: 2
segurança: 1
para: 11
desenvolvimento: 2
aplicações: 1
web.: 2
Além: 2
```