Laboratório sobre Spark/Kafka

A) Objetivo: O objetivo deste experimento é que o aluno compreenda as características de uma aplicação em cluster usando o Spark integrado ao broker Kafka.

B) Roteiro do laboratório

1. Configurando a interface *notebook* para uso do cluster:

a) Abra uma console e crie um proxy vinculado à porta 7776 com o comando abaixo:

```
$ ssh -D 7776 -p 13508 <user>@chococino.naquadah.com.br
```

b) Na estação de trabalho, chame o Mozilla Firefox, instale a extensão foxyproxy e configure a porta 7776 para o proxy do cluster.

c) Escolha uma das máquinas do cluster e ative o notebook com os comandos a seguir:

```
$ python3 -m notebook password
$ python3 -m notebook —ip cmx, onde cmx é o nome do nó escolhido para rodar o servidor
```

2. Testando o *broker* Kafka:

a) Faça testes de criação e listagem de tópicos no Kafka

```
$ kafka-topics.sh --list --bootstrap-server cm4:9092
$ kafka-topics.sh --create --topic student-a19000-saida --bootstrap-server cm4:9092
```

b) Crie um diálogo produtor/consumidor usando o tópico criado

```
$ kafka-console-consumer.sh -topic student-a19000-saida --bootstrap-server cm4:9092
$ kafka-console-producer.sh -topic student-a19000-saida --bootstrap-server cm4:9092
```

c) Na console do produtor, digite algo e veja como as mensagens são ecoadas no consumidor.

d) Abra mais um terminal e veja se esse mecanismo comporta mais de um consumidor para um mesmo produtor

3. Integrando Kafka com Spark:

Para compreender essa integração, vamos alterar o experimento anterior, de modo que o consumidor seja um *script* do Spark, usando um notebook

a) Utilizando a interface notebook, crie as seguintes células:

- `from pyspark.sql import SparkSession`
- `spark = SparkSession.builder.master("spark://cm1:7077").appName("teste kafka”).getOrCreate()`
- ```
  lines = spark.readStream.format("kafka") \
    .option("kafka.bootstrap.servers", "cm4:9092") \
    .option("subscribe", “student-a19000-saida“) \
    .option('includeTimestamp', 'true') \
    .load()
  ```
- `from pyspark.sql.functions import explode, split, col`
- ```
  words = lines.select(
    explode(split(col("value"), "\s+")).alias("word"),lines.timestamp)
  ```
- `words.writeStream.format('console').option('truncate', 'false').start()`

b) Agora retorne à console do produtor kafka e digite novos textos para ver o processamento no spark (veja as mensagens com timestamp na console do spark)

C) Atividade de entrega: Crie uma aplicação Spark que *(i)* receba as palavras de entrada via socket, *(ii)* contabilize o número de ocorrências de cada palavra, associadas a um *timestamp* e, *(iii)* mostre as contabilizações nas consoles de todos os consumidores que estiverem inscritos no canal kafka cujo nome é `student-<a_sua_matric>-saida`. (Gerar o notebook relativo e postar no Moodle)