Stream Processing Exercise 5 - Consuming from Kafka

January 28, 2021

0.1 Consuming Streaming data from Users topic

Purpose of this exercise is to analyze the data that is populated on Users topic following same approach than Exercise 4.

Finally we want to obtain how many times a user has accessed every 2 minutes over last 5 minutes.

[2]:

```
from pyspark import sql
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *

spark = SparkSession \
    .builder \
    .appName("UsersConsumer") \
    .getOrCreate()


dfUsersStream = (
    spark
    .readStream
    .format("kafka")
    .option("kafka.bootstrap.servers", "broker:29092")
    .option("subscribe", "users")
    .load()
)

dfUsers = (
    dfUsersStream
    .selectExpr("CAST(key AS STRING)", "CAST(value AS STRING)", "timestamp")
    .withColumn("_tmp", split(col("value"), "\\,"))
    .select((col("_tmp").getItem(0).cast("long") / lit(1000)).cast("timestamp").
 ↪alias("viewtime"),
            col("_tmp").getItem(1).alias("userid"),
            col("_tmp").getItem(2).alias("regionid"),
            col("_tmp").getItem(3).alias("gender"),
            col("timestamp"))
)
```

```
dfUsers.printSchema()
```

```
root
 |-- viewtime: timestamp (nullable = true)
 |-- userid: string (nullable = true)
 |-- regionid: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- timestamp: timestamp (nullable = true)
```

[3]:

```
dfUsers.writeStream.format("memory").outputMode("append").queryName("Users").
 ↪start()
```

[3]: `<pyspark.sql.streaming.StreamingQuery at 0x7ff3046fb190>`

[4]:

```
spark.sql("describe Users").show()
```

```
+---------+---------+-------+
| col_name|data_type|comment|
+---------+---------+-------+
| viewtime|timestamp|   null|
|   userid|   string|   null|
| regionid|   string|   null|
|   gender|   string|   null|
|timestamp|timestamp|   null|
+---------+---------+-------+
```

[5]:

```
dfUsersWindow=dfUsers.groupBy(window("timestamp", "5 minutes", "2 minutes"),␣
 ↪"userid").count()

dfUsersWindow.printSchema()
```

```
root
 |-- window: struct (nullable = true)
 |    |-- start: timestamp (nullable = true)
 |    |-- end: timestamp (nullable = true)
 |-- userid: string (nullable = true)
 |-- count: long (nullable = false)
```

[6]:

```
dfUsersWindow.writeStream.format("memory").outputMode("complete").
 ↪queryName("usersWindow").start()
```

[6]: `<pyspark.sql.streaming.StreamingQuery at 0x7ff2f439a5d0>`

[7]:

```
spark.sql("select * from usersWindow").show()
```

```
+--------------------+------+-----+
|              window|userid|count|
+--------------------+------+-----+
|[2021-01-28 19:20…|User_9|   53|
|[2021-01-28 19:18…|User_5|   39|
|[2021-01-28 19:18…|User_3|   55|
|[2021-01-28 19:18…|User_9|   53|
|[2021-01-28 19:20…|User_7|   47|
|[2021-01-28 19:20…|User_4|   52|
|[2021-01-28 19:18…|User_8|   45|
|[2021-01-28 19:18…|User_7|   47|
|[2021-01-28 19:20…|User_1|   40|
|[2021-01-28 19:20…|User_2|   44|
|[2021-01-28 19:18…|User_6|   47|
|[2021-01-28 19:20…|User_6|   47|
|[2021-01-28 19:20…|User_3|   55|
|[2021-01-28 19:18…|User_1|   40|
|[2021-01-28 19:20…|User_8|   45|
|[2021-01-28 19:18…|User_2|   44|
|[2021-01-28 19:18…|User_4|   52|
|[2021-01-28 19:20…|User_5|   39|
+--------------------+------+-----+
```

[ ]: