PROGRAMAÇÃO DE ENTRADAS STREAMING EM CLUSTERS

Segundo projeto de pesquisa

Equipe

Eduardo Afonso

19/0012307

Thiago Paiva

19/0020377

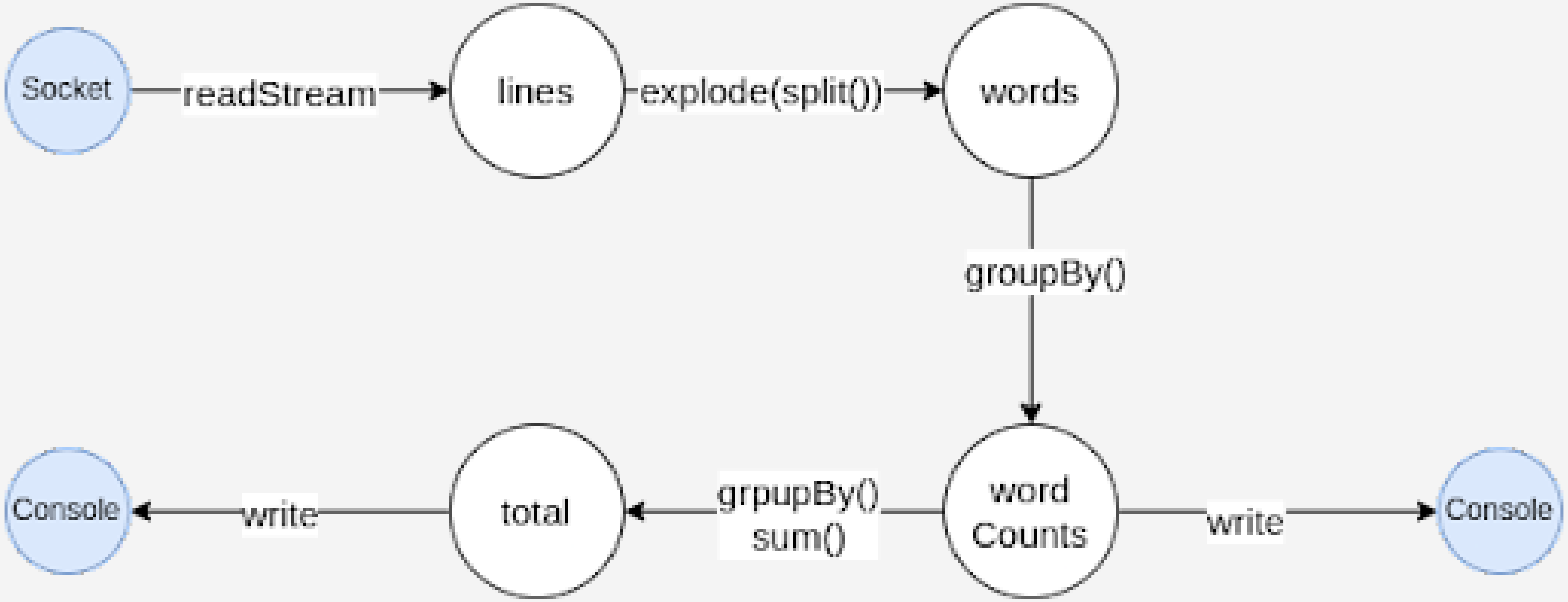
Rafael Cleydson

19/0019085

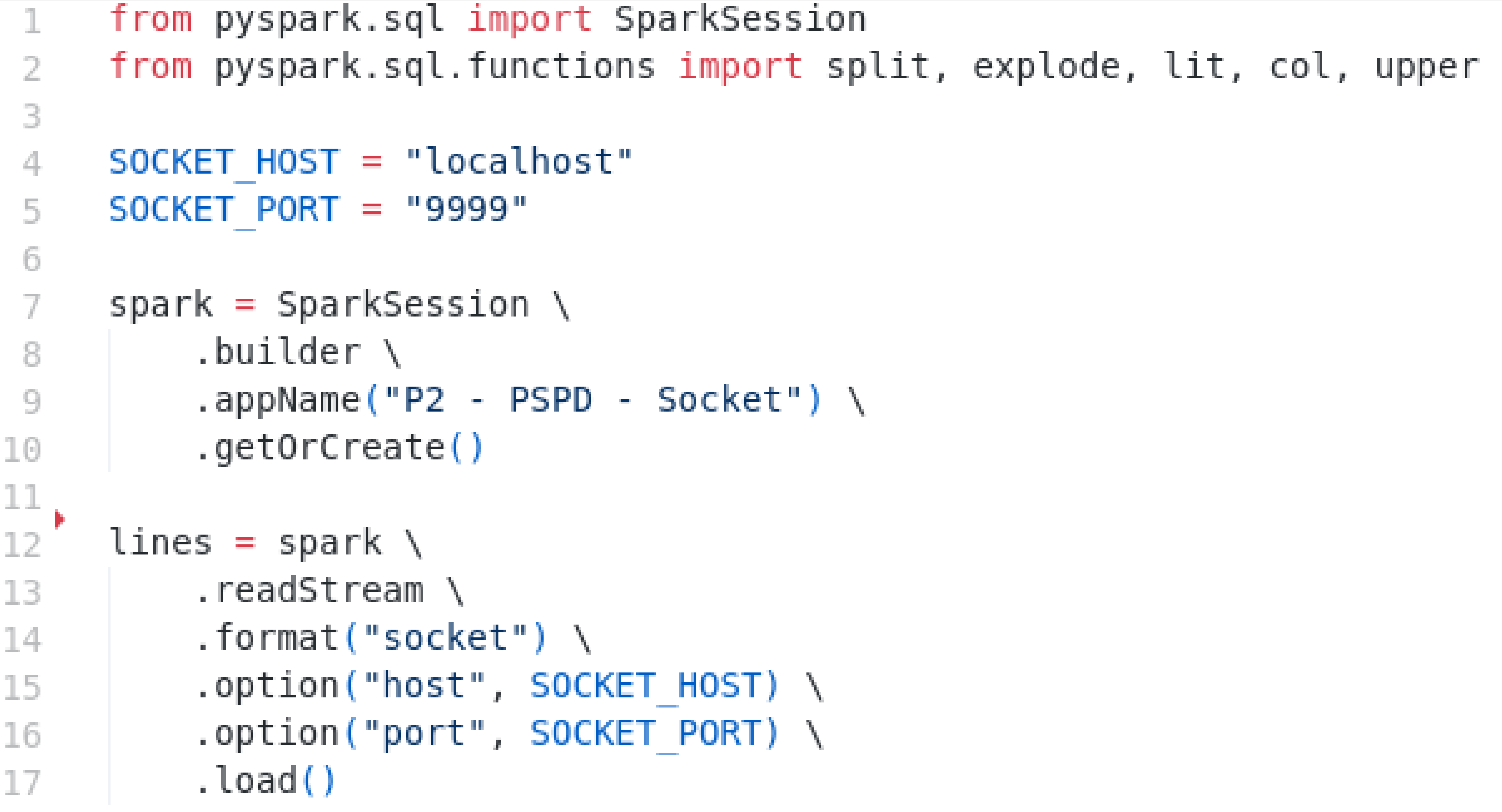
Problema

Contar palavras
Contar palavras que começam com S
Contar palavras que começam com P
Contar palavras que começam com R
Contar palavras com 6 caracteres
Contar palavras com 8 caracteres
Contar palavras com 11 caracteres

Solução Socket - Parte 1

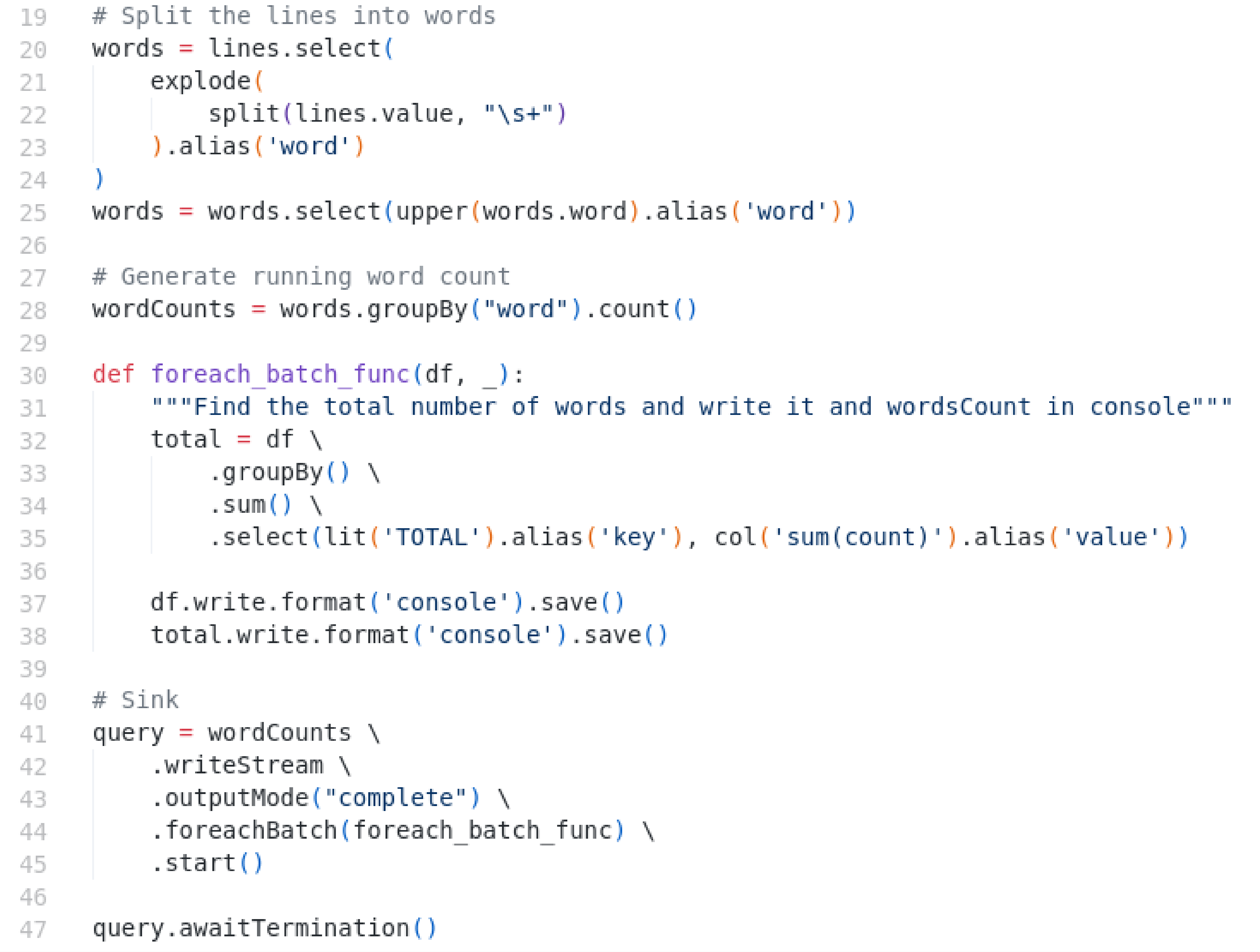


Solução Socket - Conexão



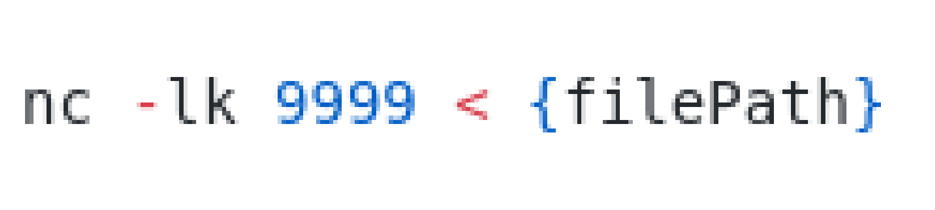
```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import split, explode, lit, col, upper

SOCKET_HOST = "localhost"
SOCKET_PORT = "9999"

spark = SparkSession \
    .builder \
    .appName("P2 - PSPD - Socket") \
    .getOrCreate()

lines = spark \
    .readStream \
    .format("socket") \
    .option("host", SOCKET_HOST) \
    .option("port", SOCKET_PORT) \
    .load()
```

Solução Socket - Contagem

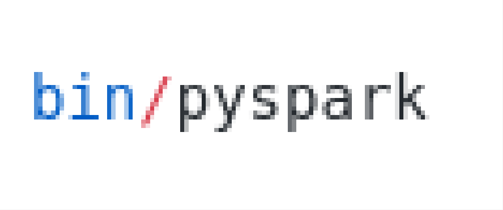


```
# Split the lines into words
words = lines.select(
    explode(
        split(lines.value, "\s+")
    ).alias('word')
)
words = words.select(upper(words.word).alias('word'))

# Generate running word count
wordCounts = words.groupBy("word").count()

def foreach_batch_func(df, _):
    """Find the total number of words and write it and wordsCount in console"""
    total = df \
        .groupBy() \
        .sum() \
        .select(lit('TOTAL').alias('key'), col('sum(count)').alias('value'))

    df.write.format('console').save()
    total.write.format('console').save()

# Sink
query = wordCounts \
    .writeStream \
    .outputMode("complete") \
    .foreachBatch(foreach_batch_func) \
    .start()

query.awaitTermination()
```

Comandos Exec Socket

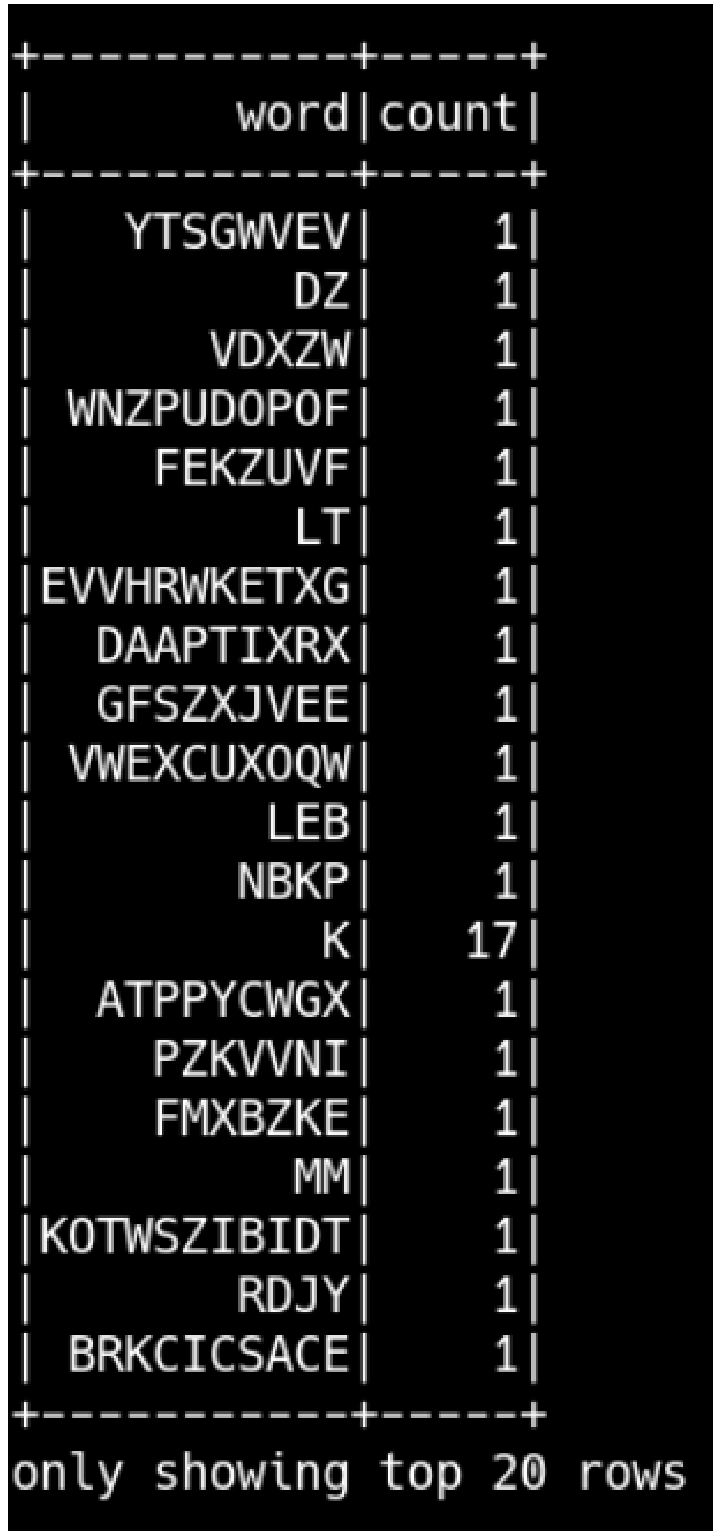


```
nc -lk 9999 < {filePath}
```

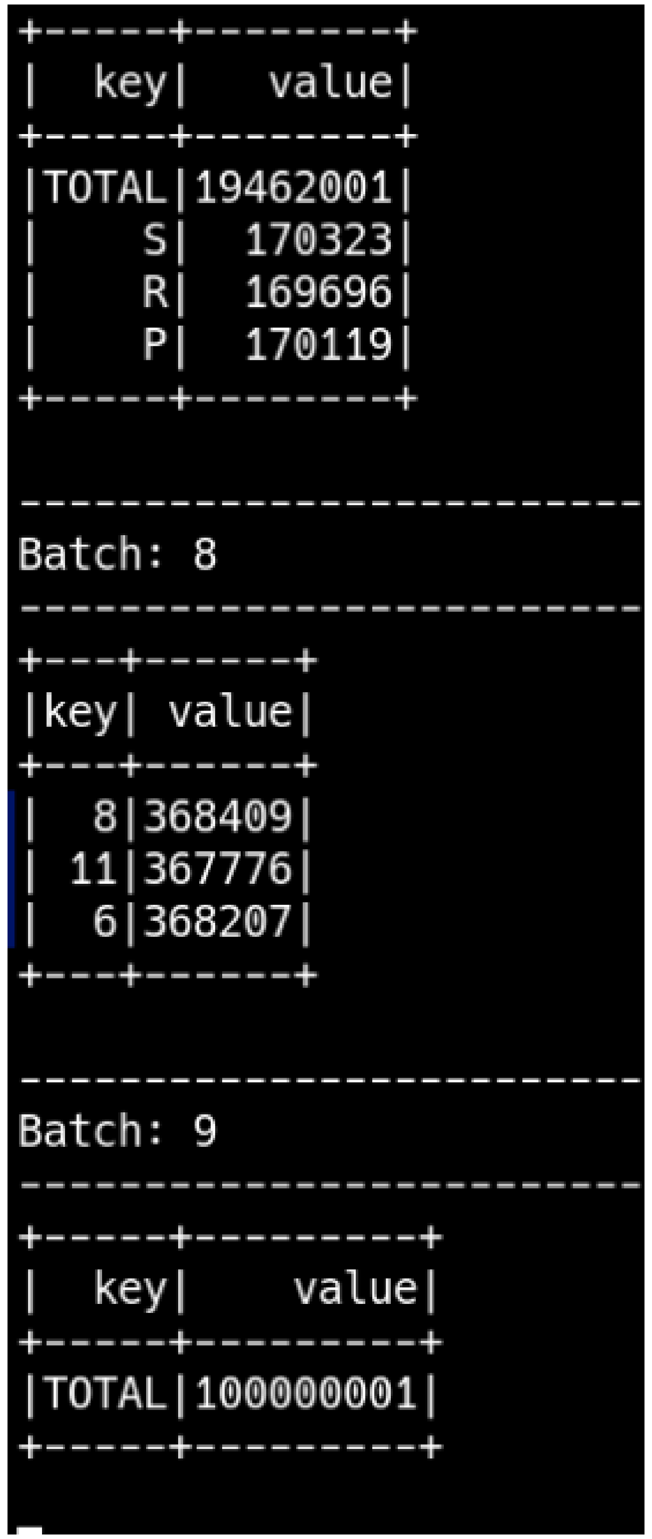


```
bin/pyspark
```

Solução Socket Saida

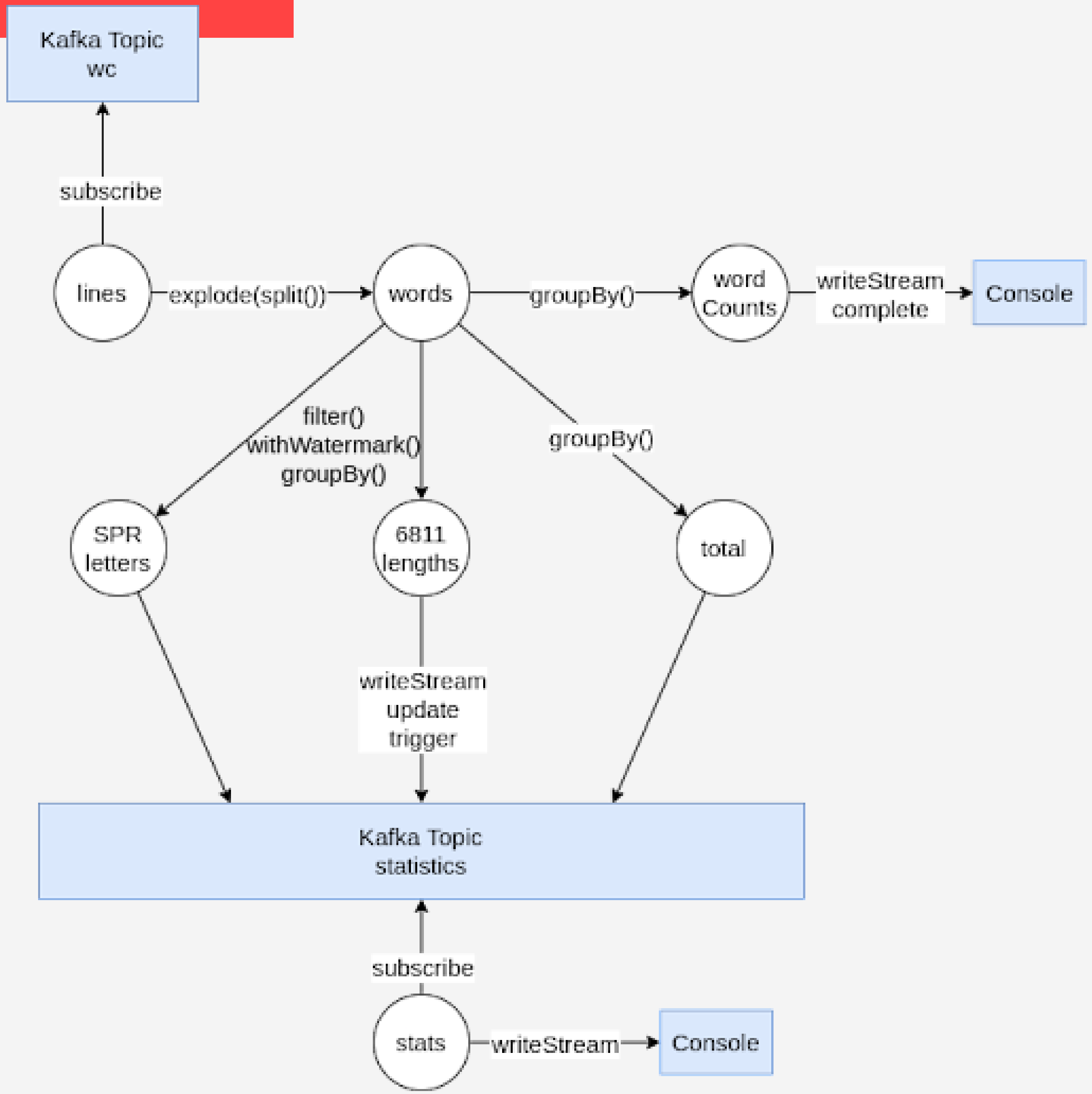


```
+-----------+-----+
|       word|count|
+-----------+-----+
|   YTSGWVEV|    1|
|         DZ|    1|
|      VDXZW|    1|
|  WNZPUDOPOF|    1|
|    FEKZUVF|    1|
|         LT|    1|
|EVVHRWKETXG|    1|
|   DAAPTIXRX|    1|
|   GFSZXJVEE|    1|
|  VWEXCUXOQW|    1|
|        LEB|    1|
|       NBKP|    1|
|          K|   17|
|   ATPPYCWGX|    1|
|    PZKVVNI|    1|
|     FMXBZKE|    1|
|         MM|    1|
|KOTWSZIBIDT|    1|
|       RDJY|    1|
|  BRKCICSACE|    1|
+-----------+-----+
only showing top 20 rows
```

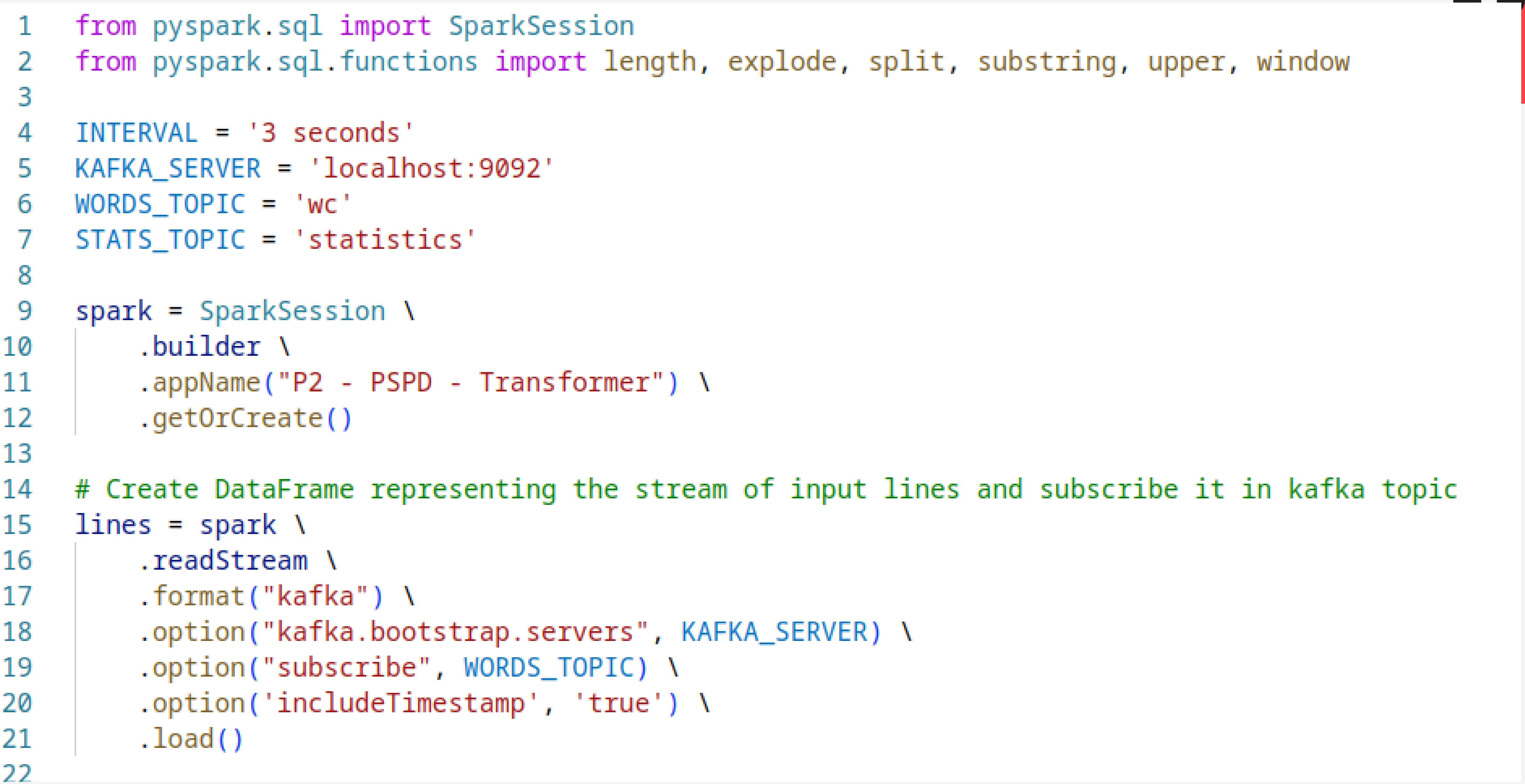


```
+-----+--------+
|  key|   value|
+-----+--------+
|TOTAL|19462001|
|    S|  170323|
|    R|  169696|
|    P|  170119|
+-----+--------+

-------------------------------------------
Batch: 8
-------------------------------------------
+---+------+
|key| value|
+---+------+
|  8|368409|
| 11|367776|
|  6|368207|
+---+------+

-------------------------------------------
Batch: 9
-------------------------------------------
+-----+---------+
|  key|    value|
+-----+---------+
|TOTAL|100000001|
+-----+---------+
```

Solução Kafka - Parte 2

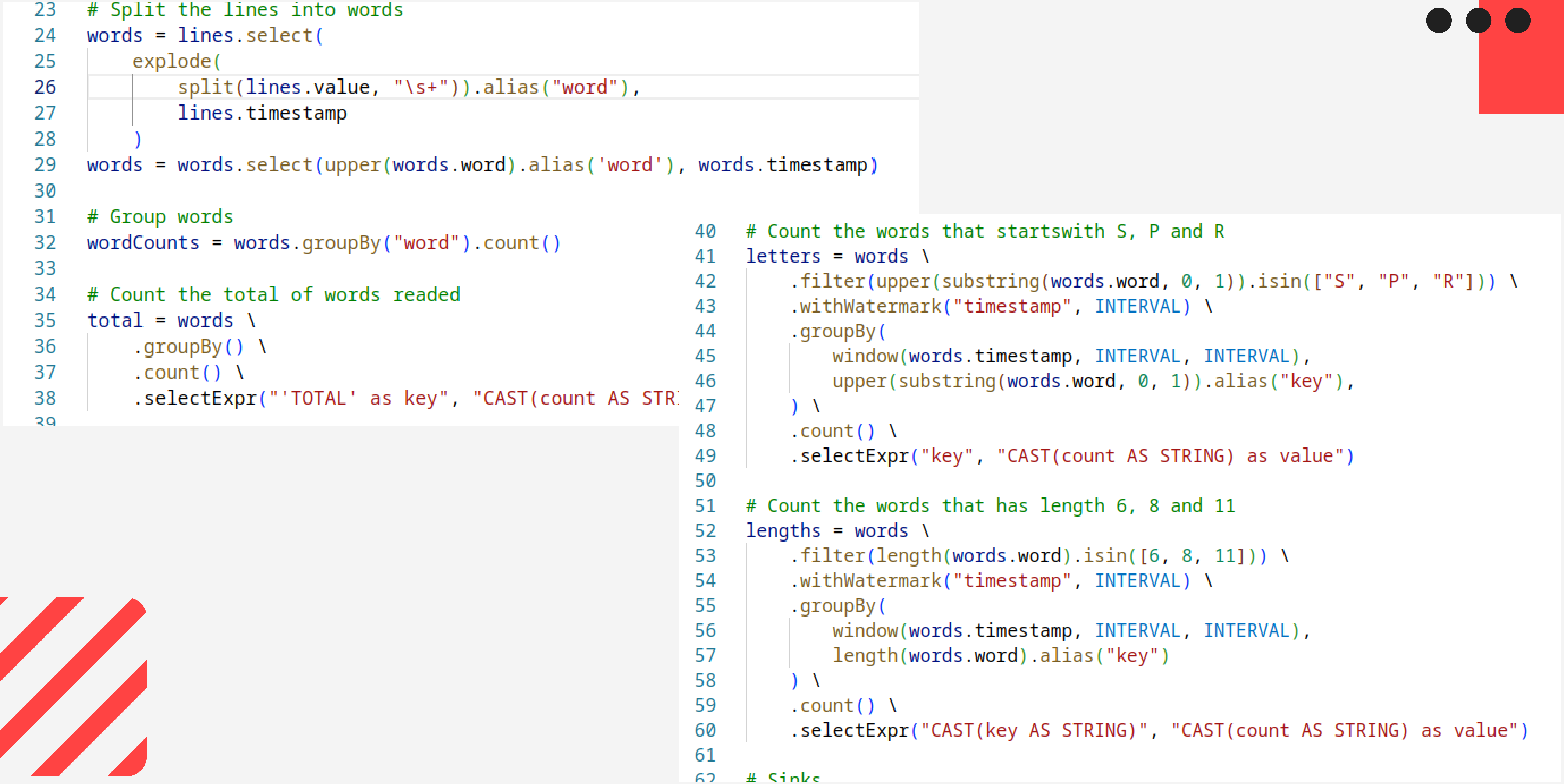


Solução Kafka - Conexao



```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import length, explode, split, substring, upper, window

INTERVAL = '3 seconds'
KAFKA_SERVER = 'localhost:9092'
WORDS_TOPIC = 'wc'
STATS_TOPIC = 'statistics'

spark = SparkSession \
    .builder \
    .appName("P2 - PSPD - Transformer") \
    .getOrCreate()

# Create DataFrame representing the stream of input lines and subscribe it in kafka topic
lines = spark \
    .readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_SERVER) \
    .option("subscribe", WORDS_TOPIC) \
    .option('includeTimestamp', 'true') \
    .load()

```

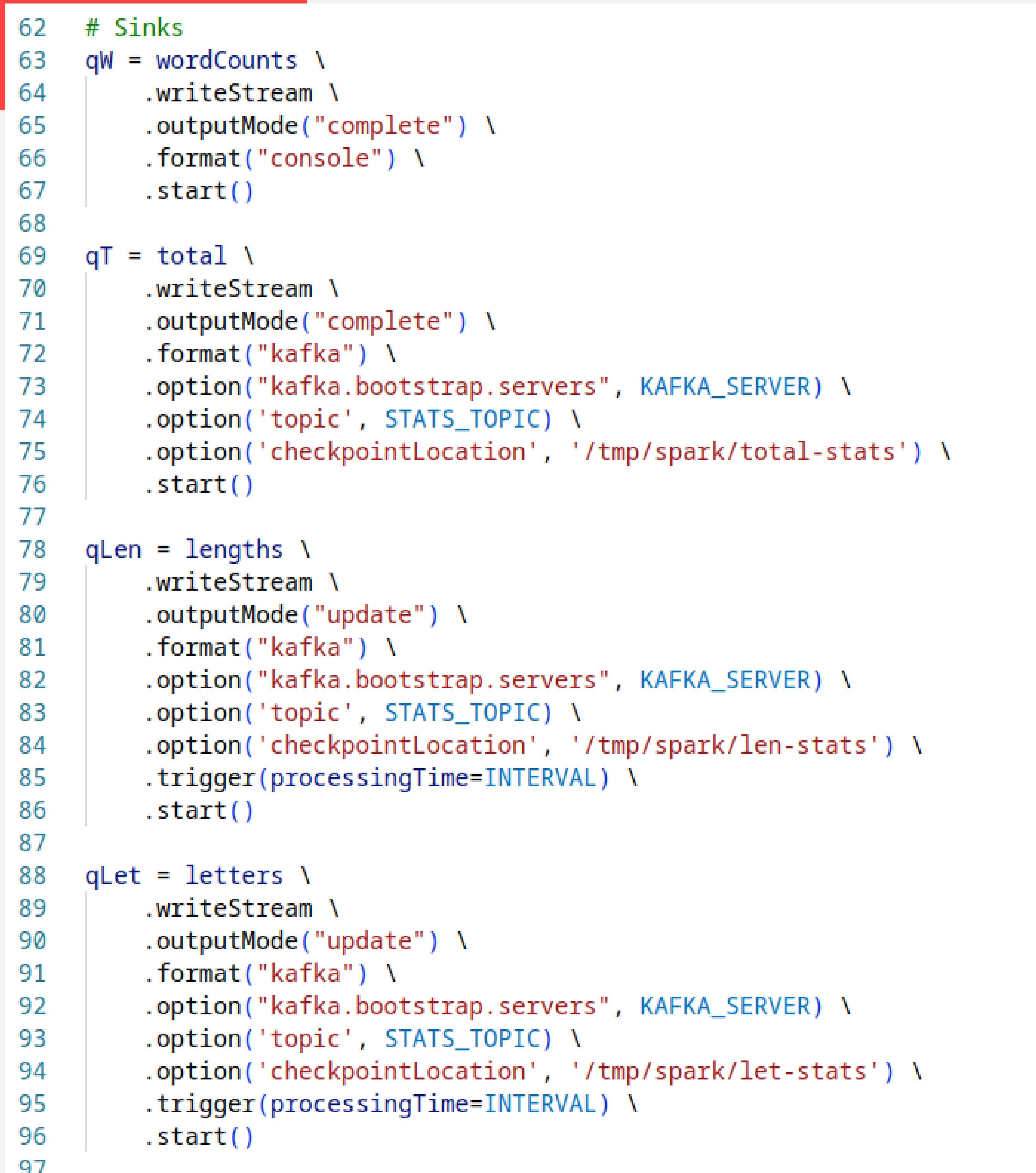
Solução Kafka - Contagem



```
# Split the lines into words
words = lines.select(
    explode(
        split(lines.value, "\s+")).alias("word"),
        lines.timestamp
    )
words = words.select(upper(words.word).alias('word'), words.timestamp)

# Group words
wordCounts = words.groupBy("word").count()

# Count the total of words readed
total = words \
    .groupBy() \
    .count() \
    .selectExpr("'TOTAL' as key", "CAST(count AS STR
```

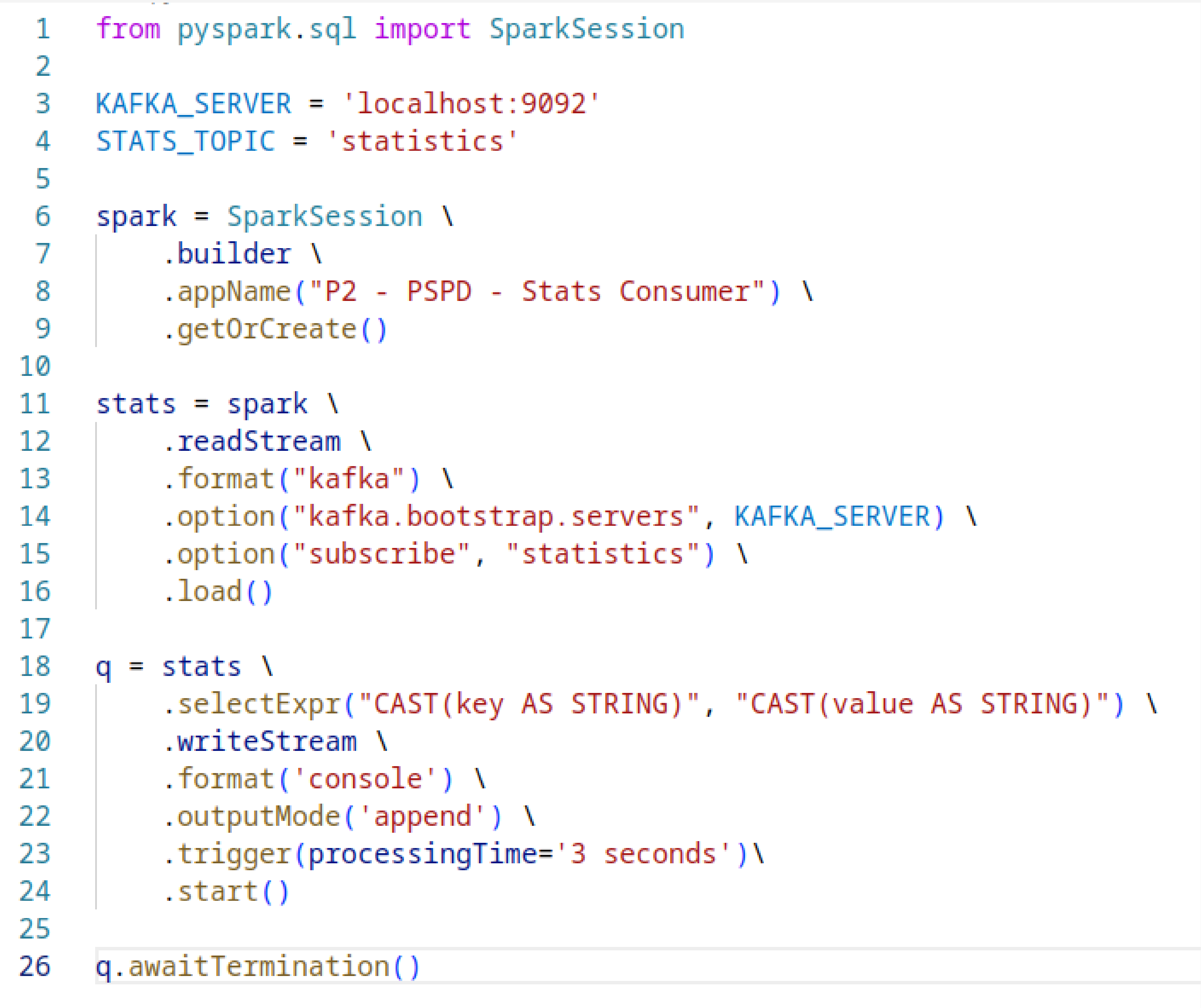
```
# Count the words that startswith S, P and R
letters = words \
    .filter(upper(substring(words.word, 0, 1)).isin(["S", "P", "R"])) \
    .withWatermark("timestamp", INTERVAL) \
    .groupBy(
        window(words.timestamp, INTERVAL, INTERVAL),
        upper(substring(words.word, 0, 1)).alias("key"),
    ) \
    .count() \
    .selectExpr("key", "CAST(count AS STRING) as value")

# Count the words that has length 6, 8 and 11
lengths = words \
    .filter(length(words.word).isin([6, 8, 11])) \
    .withWatermark("timestamp", INTERVAL) \
    .groupBy(
        window(words.timestamp, INTERVAL, INTERVAL),
        length(words.word).alias("key")
    ) \
    .count() \
    .selectExpr("CAST(key AS STRING)", "CAST(count AS STRING) as value")

# Sinks
```

Solução Kafka - Sink



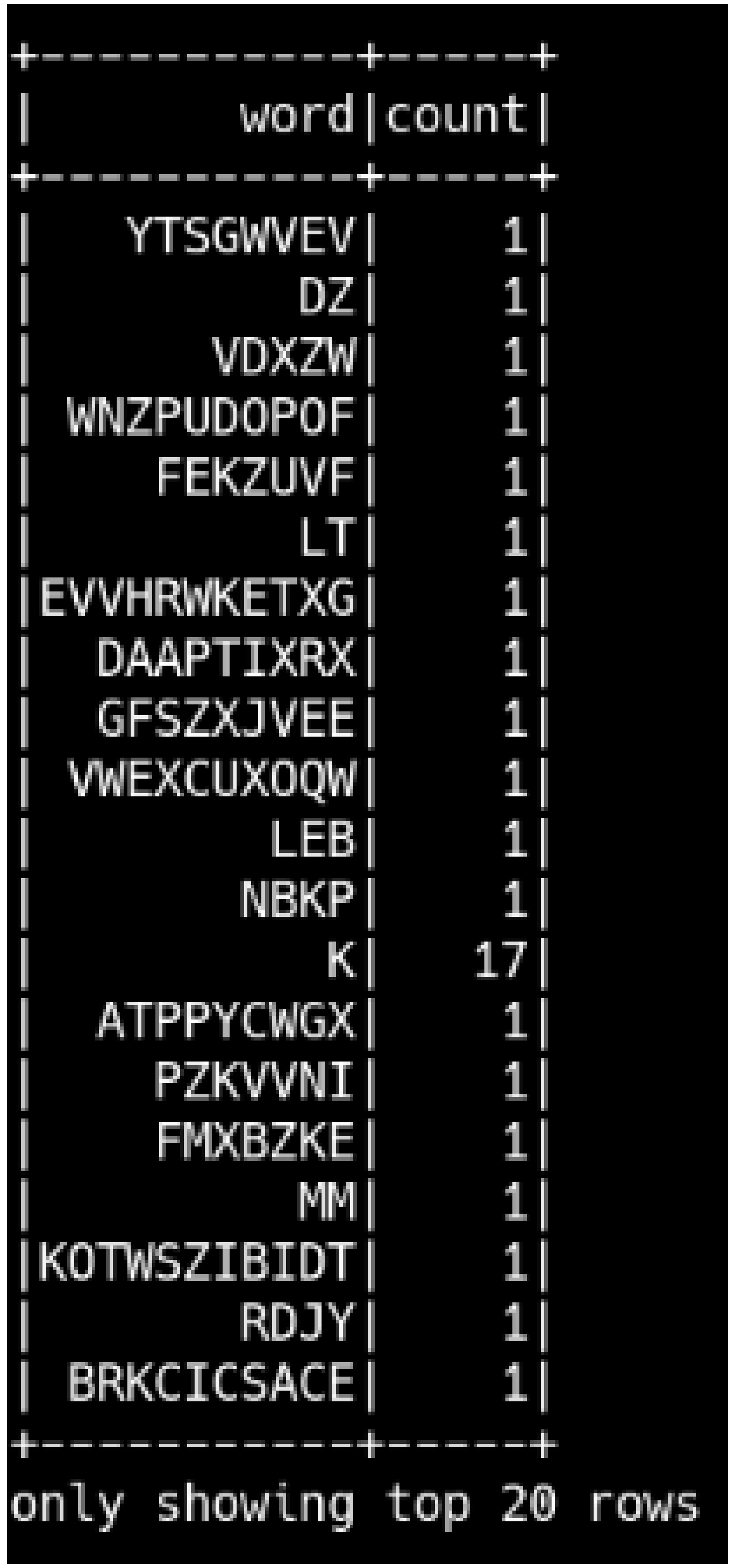
```
# Sinks
qW = wordCounts \
    .writeStream \
    .outputMode("complete") \
    .format("console") \
    .start()

qT = total \
    .writeStream \
    .outputMode("complete") \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_SERVER) \
    .option('topic', STATS_TOPIC) \
    .option('checkpointLocation', '/tmp/spark/total-stats') \
    .start()

qLen = lengths \
    .writeStream \
    .outputMode("update") \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_SERVER) \
    .option('topic', STATS_TOPIC) \
    .option('checkpointLocation', '/tmp/spark/len-stats') \
    .trigger(processingTime=INTERVAL) \
    .start()

qLet = letters \
    .writeStream \
    .outputMode("update") \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_SERVER) \
    .option('topic', STATS_TOPIC) \
    .option('checkpointLocation', '/tmp/spark/let-stats') \
    .trigger(processingTime=INTERVAL) \
    .start()
```

Solução Kafka - Visualização das Estatisticas

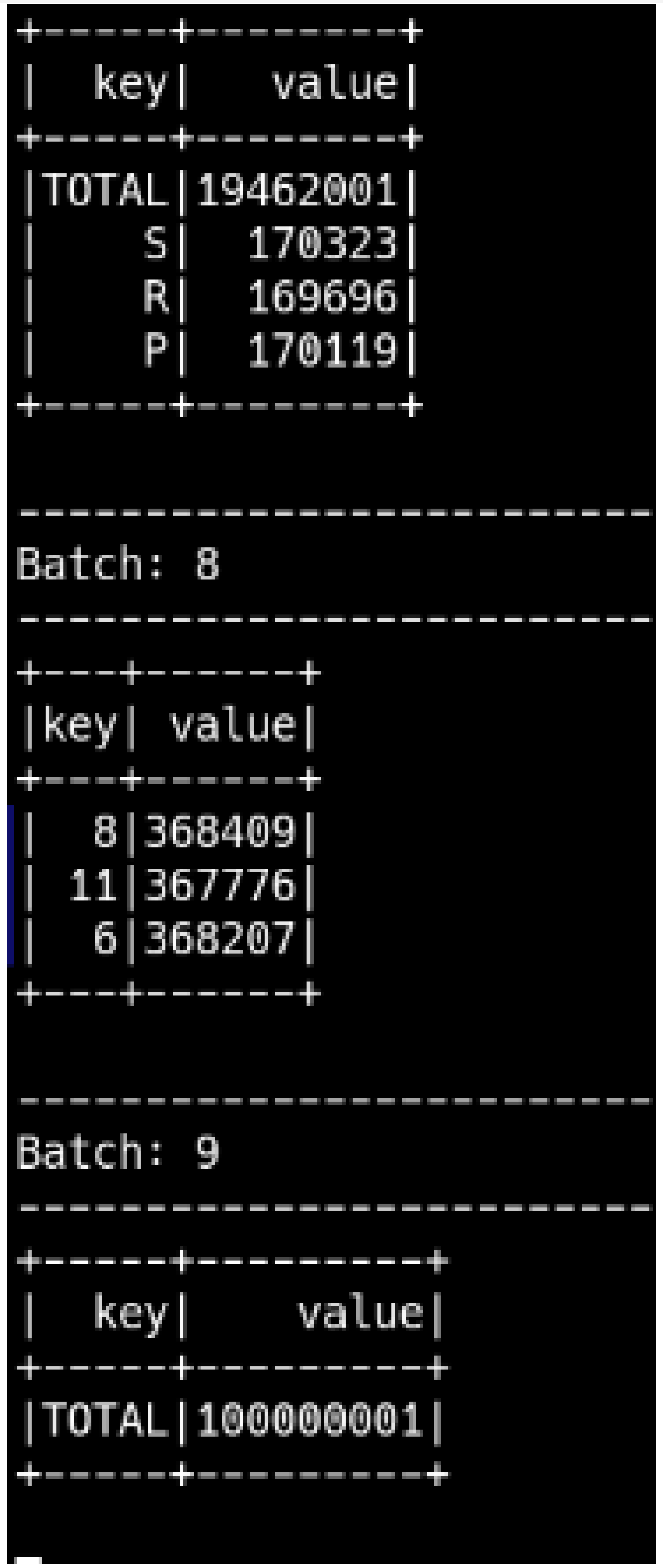


```python
from pyspark.sql import SparkSession

KAFKA_SERVER = 'localhost:9092'
STATS_TOPIC = 'statistics'

spark = SparkSession \
    .builder \
    .appName("P2 - PSPD - Stats Consumer") \
    .getOrCreate()

stats = spark \
    .readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_SERVER) \
    .option("subscribe", "statistics") \
    .load()

q = stats \
    .selectExpr("CAST(key AS STRING)", "CAST(value AS STRING)") \
    .writeStream \
    .format('console') \
    .outputMode('append') \
    .trigger(processingTime='3 seconds')\
    .start()

q.awaitTermination()
```

Solução Kafka Saida



```
+-----------+-----+
|       word|count|
+-----------+-----+
|   YTSGWVEV|    1|
|         DZ|    1|
|      VDXZW|    1|
| WNZPUDOPOF|    1|
|    FEKZUVF|    1|
|         LT|    1|
|EVVHRWKETXG|    1|
|   DAAPTIXRX|    1|
|  GFSZXJVEE|    1|
| VWEXCUXOQW|    1|
|        LEB|    1|
|       NBKP|    1|
|          K|   17|
|  ATPPYCWGX|    1|
|    PZKVVNI|    1|
|    FMXBZKE|    1|
|         MM|    1|
|KOTWSZIBIDT|    1|
|       RDJY|    1|
| BRKCICSACE|    1|
+-----------+-----+
only showing top 20 rows
```



```
+-----+--------+
|  key|   value|
+-----+--------+
|TOTAL|19462001|
|    S|  170323|
|    R|  169696|
|    P|  170119|
+-----+--------+

-------------------------------------------
Batch: 8
-------------------------------------------
+---+------+
|key| value|
+---+------+
|  8|368409|
| 11|367776|
|  6|368207|
+---+------+

-------------------------------------------
Batch: 9
-------------------------------------------
+-----+---------+
|  key|    value|
+-----+---------+
|TOTAL|100000001|
+-----+---------+
```

Obrigado pela atenção