Step 1 : Include Libraries and Initialize Spark Session¶

In [1]:

```
#import libraries
import os
os.environ['PYSPARK_SUBMIT_ARGS'] = '--packages org.apache.spark:spark-streaming-kafka-0-10_2.12:3.0.0,org.apache.spark:spark-sql-kafka-0-10_2.12:3.0.0 pyspark-shell'

#import libraries
from pyspark import SparkContext
from pyspark.streaming import StreamingContext
from pyspark.sql import Row, SparkSession

from pyspark.sql.functions import regexp_extract
import pyspark.sql.functions as F
from pyspark.sql.types import *

appName="StructuredStreamingKafka"
#initialize the spark session
spark = SparkSession.builder.appName(appName).getOrCreate()
spark.sparkContext.setLogLevel('ERROR')
#get sparkcontext from the sparksession
sc = spark.sparkContext
```

Use-Case : Tracking Server Access Log ¶

Assignment Project Exam Help

For this case, a server is going to continuously send a records of a host who is trying to access some endpoint (url) from the web server. This data will be send from a kafka producer (`KafkaProducer1.ipynb`) which is reading the data from a txt file in the dataset provided (`logs/access_log.txt`).

https://powcoder.com

Each line contains some valuable information such as:

1. Host
2. Timestamp
3. HTTP method
4. URL endpoint
5. Status code
6. Protocol
7. Content Size

Add WeChat powcoder

The goal here is to perform some real time queries from this stream of data and be able to output the results in multiple ways.

Load Kafka Stream¶

Use the `readStream` to load data from the Kafka Producer **LT1-KafkaProducer.ipynb**

In [2]:

```
# Monitor the logs data stream for new log data
topic = "w11_access_log"
df_urls = spark \
  .readStream \
  .format("kafka") \
  .option("kafka.bootstrap.servers", "127.0.0.1:9092") \
  .option("subscribe", topic) \
  .load()
```

Data Preparation ¶

We need to convert the data from the message in order to perform some queries. The steps to parse the data are:

1. Get message as a string from `value` which is binary.
2. Implement some regular expressions to capture specific fields in the message which is a line from the access log.
3. Extract the values using the regular expressions to create the dataframe.

In [9]:
```
# Get value of the kafka message
log_lines = df_urls.selectExpr("CAST(value AS STRING)")

# Parse out the common log format to a DataFrame
tsExp=r'(\d{10})\s'
statusExp = r'\s(\d{3})\s'
generalExp = r'\"(\S+)\s(\S+)\s*(\S*)\"'
hostExp = r'(\d+\.\d+\.\d+\.\d+)'

df_logs =
log_lines.select(F.from_utc_timestamp(F.from_unixtime(regexp_extract('value', tsExp,
1)),'UTC').alias('ts'),
                         regexp_extract('value', hostExp, 1).alias('host'),
                         regexp_extract('value', generalExp, 1).alias('method'),
                         regexp_extract('value', generalExp, 2).alias('endpoint'),
                         regexp_extract('value', generalExp, 3).alias('protocol'),
                         regexp_extract('value', statusExp,
1).cast('integer').alias('status'))

df_logs.printSchema()
root
 |-- ts: timestamp (nullable = true)
 |-- host: string (nullable = true)
 |-- method: string (nullable = true)
 |-- endpoint: string (nullable = true)
 |-- protocol: string (nullable = true)
 |-- status: integer (nullable = true)
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

In [10]:
```
query = df_logs \
    .writeStream \
    .outputMode("append") \
    .format("console") \
    .trigger(processingTime='5 seconds') \
    .start()
```
In [11]:
```
query.stop()
```
Aggregations on window over event-time ¶

The event-time we use here is the `ts` that we have generated in the producer.

1. Lab Task: Using the Window function, find the number of logs for each status in a window of 30 seconds. Set the window sliding interval to 10 seconds. Write the output to console sink.

In [1]:
```
#WRITE YOUR CODE HERE
```
In [13]:
```
#SEND OUTPUT TO CONSOLE SINK
```