**Exercise - Create a Kafka Producer Server**

In this workspace, you’re going to create a Kafka Producer Server. You’ll be using this code for the next few workspace exercises. You have already learned how to create a Kafka Producer Server in the previous Kafka course, so refer back to those lessons if you need to! We’re just starting you with an empty Python file for this exercise, but you can refer to the solution code if you really need to. (Try not to!)

Requirements:

* Your Kafka Server should ingest the uber.json data file in the workspace correctly. If unsure of the correct path of the file, type pwd in the console to get the absolute path of the file.
* You can name your own topic and feel free to use any free port number for this server.
* This will be your bootstrap server for your streaming application.
* You should be able to check if your server ingests data correctly by using Kafka Consumer Console.
* You can use any Kafka library (pykafka, kafka, kafka-confluent, etc). But if you wish to use a library other than kafka-confluent or kafka-python, you will have to reinstall the library each time you wake up workspace (or anytime after you've refreshed, or woken up, or reset data, or used the "Get New Content" button). The idea here is for you to generate a Python file that has a Kafka producer, and this file should act as your bootstrap server.

Please note: If you encounter this error

**Kafka Data Source**

KafkaSource is the data source for Apache Kafka in Spark Structured Streaming. It's part of kafka-0-10-sql library in Spark (source code: [**https://github.com/apache/spark/blob/4513f1c0dc450e9249d43fdad618fdcaf8d399b6/external/kafka-0-10-sql/src/main/scala/org/apache/spark/sql/kafka010/KafkaSource.scala)**](https://github.com/apache/spark/blob/4513f1c0dc450e9249d43fdad618fdcaf8d399b6/external/kafka-0-10-sql/src/main/scala/org/apache/spark/sql/kafka010/KafkaSource.scala)). A couple important functions to note in this source are:

* getOffset(), which uses the KafkaOffsetReader to get the latest available offset
* getBatch(), which returns a DataFrame from the start to end of an offset.

You should define spark-sql-kafka-0-10 module as part of the execution of your project. For Spark environments using shell script, we'll be using the following command to submit Spark jobs:

./bin/spark-shell --packages org.apache.spark:spark-sql-kafka-0-10\_2.12:2.3.4

What we’re actually doing here is including an external package to allow us to use Kafka (broker version 0.10), and the correct version of Spark we have installed (compiled with Scala 2.12 and Spark version 2.3.4). org.apache.spark:spark-sql-kafka-0-10\_<scala\_version>:<spark\_version>.

**KafkaSourceProvider Code Reference**

KafkaSourceProvider requires these options

* subscribe, subscribepattern, or assign
* kafka.bootstrap.server

Reference: [**https://github.com/apache/spark/blob/master/external/kafka-0-10-sql/src/main/scala/org/apache/spark/sql/kafka010/KafkaSourceProvider.scala**](https://github.com/apache/spark/blob/master/external/kafka-0-10-sql/src/main/scala/org/apache/spark/sql/kafka010/KafkaSourceProvider.scala)

### Exercise: Kafka Source Provider

Using the Kafka Producer Server you created in the previous exercise, we can now ingest data from the Kafka Producer Server to a Structured Streaming application. First you’ll need to set up the entry point for the stream.

Requirements:

* Set up the entry point
* Use appropriate configurations in the options to ingest the Kafka stream
* Do a df.printSchema() to explore the schema of the default Kafka ingestion
* You can use any Kafka library (pykafka, kafka, kafka-confluent, etc). But if you wish to use a library other than kafka-confluent or kafka-python, you will have to reinstall the library each time you wake up workspace (or anytime after you've refreshed, or woken up, or reset data, or used the "Get New Content" button). The idea here is for you to generate a Python file that has a Kafka producer, and this file should act as your bootstrap server.

Please note: If you encounter this error

**KafkaSourceProvider Code Reference**

KafkaSourceProvider requires these options

* subscribe, subscribepattern, or assign
* kafka.bootstrap.server

### xercise: Kafka Source Provider

Using the Kafka Producer Server you created in the previous exercise, we can now ingest data from the Kafka Producer Server to a Structured Streaming application. First you’ll need to set up the entry point for the stream.

Requirements:

* Set up the entry point
* Use appropriate configurations in the options to ingest the Kafka stream
* Do a df.printSchema() to explore the schema of the default Kafka ingestion
* You can use any Kafka library (pykafka, kafka, kafka-confluent, etc). But if you wish to use a library other than kafka-confluent or kafka-python, you will have to reinstall the library each time you wake up workspace (or anytime after you've refreshed, or woken up, or reset data, or used the "Get New Content" button). The idea here is for you to generate a Python file that has a Kafka producer, and this file should act as your bootstrap server.

Please note: If you encounter this error