Assignment 9.1

May 13, 2023

0.1 Assignment 9.1

[1]:

```
import os
import shutil
import json
from pathlib import Path

import pandas as pd
import warnings
warnings.filterwarnings('ignore')

from kafka import KafkaProducer, KafkaAdminClient
from kafka.admin.new_topic import NewTopic
from kafka.errors import TopicAlreadyExistsError

from pyspark import SparkConf
from pyspark.sql import SparkSession
from pyspark.streaming import StreamingContext
from pyspark import SparkConf
from pyspark.sql.functions import window, from_json, col
from pyspark.sql.types import StringType, TimestampType, DoubleType,␣
  ↪StructField, StructType
from pyspark.sql.functions import udf

current_dir = Path(os.getcwd()).absolute()
checkpoint_dir = current_dir.joinpath('checkpoints')
locations_checkpoint_dir = checkpoint_dir.joinpath('locations')
accelerations_checkpoint_dir = checkpoint_dir.joinpath('accelerations')

if locations_checkpoint_dir.exists():
    shutil.rmtree(locations_checkpoint_dir)

if accelerations_checkpoint_dir.exists():
    shutil.rmtree(accelerations_checkpoint_dir)

locations_checkpoint_dir.mkdir(parents=True, exist_ok=True)
accelerations_checkpoint_dir.mkdir(parents=True, exist_ok=True)
```

0.1.1 Configuration Parameters

TODO: Change the configuration prameters to the appropriate values for your setup.

[3]:

```
config = dict(
    bootstrap_servers=['kafka.kafka.svc.cluster.local:9092'],
    first_name='Jake',
    last_name='Meyer'
)

config['client_id'] = '{}{}'.format(
    config['last_name'],
    config['first_name']
)
config['topic_prefix'] = '{}{}'.format(
    config['last_name'],
    config['first_name']
)

config['locations_topic'] = '{}-locations'.format(config['topic_prefix'])
config['accelerations_topic'] = '{}-accelerations'.format(config['topic_prefix'])
config['simple_topic'] = '{}-simple'.format(config['topic_prefix'])

config
```

[3]:

```
{'bootstrap_servers': ['kafka.kafka.svc.cluster.local:9092'],
 'first_name': 'Jake',
 'last_name': 'Meyer',
 'client_id': 'MeyerJake',
 'topic_prefix': 'MeyerJake',
 'locations_topic': 'MeyerJake-locations',
 'accelerations_topic': 'MeyerJake-accelerations',
 'simple_topic': 'MeyerJake-simple'}
```

0.1.2 Create Topic Utility Function

The `create_kafka_topic` helps create a Kafka topic based on your configuration settings. For instance, if your first name is *John* and your last name is *Doe*, `create_kafka_topic('locations')` will create a topic with the name `DoeJohn-locations`. The function will not create the topic if it already exists.

[4]:

```
def create_kafka_topic(topic_name, config=config, num_partitions=1, replication_factor=1):
    bootstrap_servers = config['bootstrap_servers']
    client_id = config['client_id']
    topic_prefix = config['topic_prefix']
    name = '{}-{}'.format(topic_prefix, topic_name)
    admin_client = KafkaAdminClient(
        bootstrap_servers=bootstrap_servers,
        client_id=client_id
    )

    topic = NewTopic(
        name=name,
        num_partitions=num_partitions,
        replication_factor=replication_factor
    )

    topic_list = [topic]
    try:
        admin_client.create_topics(new_topics=topic_list)
        print('Created topic "{}"'.format(name))
    except TopicAlreadyExistsError as e:
        print('Topic "{}" already exists'.format(name))

create_kafka_topic('simple')
```

```
Topic "MeyerJake-simple" already exists
```

[6]:

```
spark = SparkSession\
    .builder\
    .appName("Assignment 9")\
    .getOrCreate()


df_locations = spark \
  .readStream \
  .format("kafka") \
  .option("kafka.bootstrap.servers", "kafka.kafka.svc.cluster.local:9092") \
  .option("subscribe", config['locations_topic']) \
  .option("startingOffsets", "earliest") \
  .load()
```

[7]:

```
spark.version
```

[7]:

```
'3.4.0'
```

[8]:

```
## Understand what df_Locations looks like.
print(df_locations)
```

```
DataFrame[key: binary, value: binary, topic: string, partition: int, offset:
bigint, timestamp: timestamp, timestampType: int]
```

[9]:

```
## Understand what the schema looks like.
print(df_locations.printSchema())
```

```
root
 |-- key: binary (nullable = true)
 |-- value: binary (nullable = true)
 |-- topic: string (nullable = true)
 |-- partition: integer (nullable = true)
 |-- offset: long (nullable = true)
 |-- timestamp: timestamp (nullable = true)
 |-- timestampType: integer (nullable = true)

None
```

TODO: Create a data frame called `df_accelerations` that reads from the accelerations topic you published to in assignment 8. In order to read data from this topic, make sure that you are running the notebook you created in assignment 8 that publishes acceleration and location data to the `LastnameFirstname-simple` topic.

[10]:

```
spark = SparkSession\
    .builder\
    .appName("Assignment 9")\
    .getOrCreate()


## Try following similar suite as df_locations setup as above, but for␣
  ↪accelerations.
df_accelerations = spark \
  .readStream \
  .format("kafka") \
  .option("kafka.bootstrap.servers", "kafka.kafka.svc.cluster.local:9092") \
  .option("subscribe", config['accelerations_topic']) \
  .option("startingOffsets", "earliest") \
  .load()
```

[11]:

```
## Understand what df_accelerations looks like.
print(df_accelerations)
```

```
DataFrame[key: binary, value: binary, topic: string, partition: int, offset:
bigint, timestamp: timestamp, timestampType: int]
```

[12]:

```
## Understand what the schema looks like.
print(df_accelerations.printSchema())
```

```
root
 |-- key: binary (nullable = true)
 |-- value: binary (nullable = true)
 |-- topic: string (nullable = true)
 |-- partition: integer (nullable = true)
 |-- offset: long (nullable = true)
 |-- timestamp: timestamp (nullable = true)
 |-- timestampType: integer (nullable = true)

None
```

TODO: Create two streaming queries, `ds_locations` and `ds_accelerations` that publish to the `LastnameFirstname-simple` topic. See http://spark.apache.org/docs/latest/structured-streaming-programming-guide.html#starting-streaming-queries and http://spark.apache.org/docs/latest/structured-streaming-kafka-integration.html for more information.

```
[13]: ## As specified in the resources, try following code examples for setting up␣
        ↪the streaming queries
      ## for ds_locations and ds_accelerations.

      ds_locations = df_locations \
        .selectExpr("CAST(value AS STRING)") \
        .writeStream \
        .format("kafka") \
        .option("kafka.bootstrap.servers", "kafka.kafka.svc.cluster.local:9092") \
        .option("subscribe", config['locations_topic']) \
        .option("checkpointLocation", locations_checkpoint_dir) \
        .start()

      ds_accelerations = df_accelerations \
        .selectExpr("CAST(value AS STRING)") \
        .writeStream \
        .format("kafka") \
        .option("kafka.bootstrap.servers", "kafka.kafka.svc.cluster.local:9092") \
        .option("topic", config['simple_topic']) \
        .option("checkpointLocation", accelerations_checkpoint_dir) \
        .start()

      try:
          ds_locations.awaitTermination()
          ds_accelerations.awaitTermination()
      except KeyboardInterrupt:
          print("STOPPING STREAMING DATA")
```

```
23/05/13 14:14:24 WARN ResolveWriteToStream: spark.sql.adaptive.enabled is not
supported in streaming DataFrames/Datasets and will be disabled.
23/05/13 14:14:24 WARN ResolveWriteToStream: spark.sql.adaptive.enabled is not
supported in streaming DataFrames/Datasets and will be disabled.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'key.deserializer'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'key.deserializer'
```

```
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'value.deserializer'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'enable.auto.commit'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'value.deserializer'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'max.poll.records'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'auto.offset.reset'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'enable.auto.commit'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'max.poll.records'
was supplied but isn't a known config.
23/05/13 14:14:25 WARN AdminClientConfig: The configuration 'auto.offset.reset'
was supplied but isn't a known config.
23/05/13 14:14:25 ERROR MicroBatchExecution: Query [id =
1f6412a2-7f18-4af9-a218-3499e16d1422, runId =
77b8b955-4e7c-4a03-afe6-5342deb06cf3] terminated with error
java.lang.NoClassDefFoundError: org/apache/kafka/clients/admin/OffsetSpec
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$2(KafkaOffsetReaderAdmin.scala:289)
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.mutable.AbstractSet.scala$collection$SetLike$$super$
map(Set.scala:50)
        at scala.collection.SetLike.map(SetLike.scala:105)
        at scala.collection.SetLike.map$(SetLike.scala:105)
        at scala.collection.mutable.AbstractSet.map(Set.scala:50)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$1(KafkaOffsetReaderAdmin.scala:289)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$partiti
onsAssignedToAdmin$1(KafkaOffsetReaderAdmin.scala:501)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.withRetries(Kafk
aOffsetReaderAdmin.scala:518)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.partitionsAssign
edToAdmin(KafkaOffsetReaderAdmin.scala:498)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.fetchEarliestOff
sets(KafkaOffsetReaderAdmin.scala:288)
```

```
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.$anonfun$getOrCre
ateInitialPartitionOffsets$1(KafkaMicroBatchStream.scala:249)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.getOrCreateInitia
lPartitionOffsets(KafkaMicroBatchStream.scala:246)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.initialOffset(Kaf
kaMicroBatchStream.scala:98)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$getStartOffset$2(MicroBatchExecution.scala:455)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.getStart
Offset(MicroBatchExecution.scala:455)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$4(MicroBatchExecution.scala:489)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$2(MicroBatchExecution.scala:488)
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.AbstractTraversable.map(Traversable.scala:108)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$1(MicroBatchExecution.scala:477)
        at
scala.runtime.java8.JFunction0$mcZ$sp.apply(JFunction0$mcZ$sp.java:23)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.withProg
ressLocked(MicroBatchExecution.scala:802)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.construc
tNextBatch(MicroBatchExecution.scala:473)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$2(MicroBatchExecution.scala:266)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
```

```
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$1(MicroBatchExecution.scala:247)
        at org.apache.spark.sql.execution.streaming.ProcessingTimeExecutor.execu
te(TriggerExecutor.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.runActiv
atedStream(MicroBatchExecution.scala:237)
        at org.apache.spark.sql.execution.streaming.StreamExecution.$anonfun$run
Stream$1(StreamExecution.scala:306)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:827)
        at org.apache.spark.sql.execution.streaming.StreamExecution.org$apache$s
park$sql$execution$streaming$StreamExecution$$runStream(StreamExecution.scala:28
4)
        at org.apache.spark.sql.execution.streaming.StreamExecution$$anon$1.run(
StreamExecution.scala:207)
Caused by: java.lang.ClassNotFoundException:
org.apache.kafka.clients.admin.OffsetSpec
        at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinCla
ssLoader.java:641)
        at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(C
lassLoaders.java:188)
        at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:520)
        … 58 more
23/05/13 14:14:25 ERROR MicroBatchExecution: Query [id =
02606d62-9c8c-4cbc-8ba9-c8b525b745f5, runId =
1278a9d7-f0b4-4ea7-b4f2-02054f697b67] terminated with error
java.lang.NoClassDefFoundError: org/apache/kafka/clients/admin/OffsetSpec
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$2(KafkaOffsetReaderAdmin.scala:289)
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.mutable.AbstractSet.scala$collection$SetLike$$super$
map(Set.scala:50)
        at scala.collection.SetLike.map(SetLike.scala:105)
        at scala.collection.SetLike.map$(SetLike.scala:105)
        at scala.collection.mutable.AbstractSet.map(Set.scala:50)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$1(KafkaOffsetReaderAdmin.scala:289)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$partiti
onsAssignedToAdmin$1(KafkaOffsetReaderAdmin.scala:501)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.withRetries(Kafk
aOffsetReaderAdmin.scala:518)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.partitionsAssign
edToAdmin(KafkaOffsetReaderAdmin.scala:498)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.fetchEarliestOff
sets(KafkaOffsetReaderAdmin.scala:288)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.$anonfun$getOrCre
ateInitialPartitionOffsets$1(KafkaMicroBatchStream.scala:249)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.getOrCreateInitia
lPartitionOffsets(KafkaMicroBatchStream.scala:246)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.initialOffset(Kaf
kaMicroBatchStream.scala:98)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$getStartOffset$2(MicroBatchExecution.scala:455)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.getStart
Offset(MicroBatchExecution.scala:455)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$4(MicroBatchExecution.scala:489)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$2(MicroBatchExecution.scala:488)
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.AbstractTraversable.map(Traversable.scala:108)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$1(MicroBatchExecution.scala:477)
        at
scala.runtime.java8.JFunction0$mcZ$sp.apply(JFunction0$mcZ$sp.java:23)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.withProg
```

```
ressLocked(MicroBatchExecution.scala:802)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.construc
tNextBatch(MicroBatchExecution.scala:473)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$2(MicroBatchExecution.scala:266)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$1(MicroBatchExecution.scala:247)
        at org.apache.spark.sql.execution.streaming.ProcessingTimeExecutor.execu
te(TriggerExecutor.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.runActiv
atedStream(MicroBatchExecution.scala:237)
        at org.apache.spark.sql.execution.streaming.StreamExecution.$anonfun$run
Stream$1(StreamExecution.scala:306)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:827)
        at org.apache.spark.sql.execution.streaming.StreamExecution.org$apache$s
park$sql$execution$streaming$StreamExecution$$runStream(StreamExecution.scala:28
4)
        at org.apache.spark.sql.execution.streaming.StreamExecution$$anon$1.run(
StreamExecution.scala:207)
Caused by: java.lang.ClassNotFoundException:
org.apache.kafka.clients.admin.OffsetSpec
        … 58 more
Exception in thread "stream execution thread for [id =
1f6412a2-7f18-4af9-a218-3499e16d1422, runId =
77b8b955-4e7c-4a03-afe6-5342deb06cf3]" java.lang.NoClassDefFoundError:
org/apache/kafka/clients/admin/OffsetSpec
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$2(KafkaOffsetReaderAdmin.scala:289)
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.mutable.AbstractSet.scala$collection$SetLike$$super$
map(Set.scala:50)
        at scala.collection.SetLike.map(SetLike.scala:105)
        at scala.collection.SetLike.map$(SetLike.scala:105)
        at scala.collection.mutable.AbstractSet.map(Set.scala:50)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$1(KafkaOffsetReaderAdmin.scala:289)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$partiti
onsAssignedToAdmin$1(KafkaOffsetReaderAdmin.scala:501)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.withRetries(Kafk
aOffsetReaderAdmin.scala:518)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.partitionsAssign
edToAdmin(KafkaOffsetReaderAdmin.scala:498)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.fetchEarliestOff
sets(KafkaOffsetReaderAdmin.scala:288)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.$anonfun$getOrCre
ateInitialPartitionOffsets$1(KafkaMicroBatchStream.scala:249)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.getOrCreateInitia
lPartitionOffsets(KafkaMicroBatchStream.scala:246)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.initialOffset(Kaf
kaMicroBatchStream.scala:98)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$getStartOffset$2(MicroBatchExecution.scala:455)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.getStart
Offset(MicroBatchExecution.scala:455)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$4(MicroBatchExecution.scala:489)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$2(MicroBatchExecution.scala:488)
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.AbstractTraversable.map(Traversable.scala:108)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$1(MicroBatchExecution.scala:477)
        at
scala.runtime.java8.JFunction0$mcZ$sp.apply(JFunction0$mcZ$sp.java:23)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.withProg
ressLocked(MicroBatchExecution.scala:802)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.construc
tNextBatch(MicroBatchExecution.scala:473)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$2(MicroBatchExecution.scala:266)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$1(MicroBatchExecution.scala:247)
        at org.apache.spark.sql.execution.streaming.ProcessingTimeExecutor.execu
te(TriggerExecutor.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.runActiv
atedStream(MicroBatchExecution.scala:237)
        at org.apache.spark.sql.execution.streaming.StreamExecution.$anonfun$run
Stream$1(StreamExecution.scala:306)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:827)
        at org.apache.spark.sql.execution.streaming.StreamExecution.org$apache$s
park$sql$execution$streaming$StreamExecution$$runStream(StreamExecution.scala:28
4)
        at org.apache.spark.sql.execution.streaming.StreamExecution$$anon$1.run(
StreamExecution.scala:207)
Caused by: java.lang.ClassNotFoundException:
org.apache.kafka.clients.admin.OffsetSpec
        at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinCla
ssLoader.java:641)
        at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(C
lassLoaders.java:188)
        at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:520)
        … 58 more
Exception in thread "stream execution thread for [id =
02606d62-9c8c-4cbc-8ba9-c8b525b745f5, runId =
1278a9d7-f0b4-4ea7-b4f2-02054f697b67]" java.lang.NoClassDefFoundError:
org/apache/kafka/clients/admin/OffsetSpec
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$2(KafkaOffsetReaderAdmin.scala:289)
```

```
        at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.mutable.AbstractSet.scala$collection$SetLike$$super$
map(Set.scala:50)
        at scala.collection.SetLike.map(SetLike.scala:105)
        at scala.collection.SetLike.map$(SetLike.scala:105)
        at scala.collection.mutable.AbstractSet.map(Set.scala:50)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$fetchEa
rliestOffsets$1(KafkaOffsetReaderAdmin.scala:289)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.$anonfun$partiti
onsAssignedToAdmin$1(KafkaOffsetReaderAdmin.scala:501)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.withRetries(Kafk
aOffsetReaderAdmin.scala:518)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.partitionsAssign
edToAdmin(KafkaOffsetReaderAdmin.scala:498)
        at org.apache.spark.sql.kafka010.KafkaOffsetReaderAdmin.fetchEarliestOff
sets(KafkaOffsetReaderAdmin.scala:288)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.$anonfun$getOrCre
ateInitialPartitionOffsets$1(KafkaMicroBatchStream.scala:249)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.getOrCreateInitia
lPartitionOffsets(KafkaMicroBatchStream.scala:246)
        at org.apache.spark.sql.kafka010.KafkaMicroBatchStream.initialOffset(Kaf
kaMicroBatchStream.scala:98)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$getStartOffset$2(MicroBatchExecution.scala:455)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.getStart
Offset(MicroBatchExecution.scala:455)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$4(MicroBatchExecution.scala:489)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$2(MicroBatchExecution.scala:488)
        at
```

```
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
        at scala.collection.Iterator.foreach(Iterator.scala:943)
        at scala.collection.Iterator.foreach$(Iterator.scala:943)
        at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
        at scala.collection.IterableLike.foreach(IterableLike.scala:74)
        at scala.collection.IterableLike.foreach$(IterableLike.scala:73)
        at scala.collection.AbstractIterable.foreach(Iterable.scala:56)
        at scala.collection.TraversableLike.map(TraversableLike.scala:286)
        at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
        at scala.collection.AbstractTraversable.map(Traversable.scala:108)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$constructNextBatch$1(MicroBatchExecution.scala:477)
        at
scala.runtime.java8.JFunction0$mcZ$sp.apply(JFunction0$mcZ$sp.java:23)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.withProg
ressLocked(MicroBatchExecution.scala:802)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.construc
tNextBatch(MicroBatchExecution.scala:473)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$2(MicroBatchExecution.scala:266)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken(ProgressReporter.scala:411)
        at org.apache.spark.sql.execution.streaming.ProgressReporter.reportTimeT
aken$(ProgressReporter.scala:409)
        at org.apache.spark.sql.execution.streaming.StreamExecution.reportTimeTa
ken(StreamExecution.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.$anonfun
$runActivatedStream$1(MicroBatchExecution.scala:247)
        at org.apache.spark.sql.execution.streaming.ProcessingTimeExecutor.execu
te(TriggerExecutor.scala:67)
        at org.apache.spark.sql.execution.streaming.MicroBatchExecution.runActiv
atedStream(MicroBatchExecution.scala:237)
        at org.apache.spark.sql.execution.streaming.StreamExecution.$anonfun$run
Stream$1(StreamExecution.scala:306)
        at
scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
        at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:827)
        at org.apache.spark.sql.execution.streaming.StreamExecution.org$apache$s
park$sql$execution$streaming$StreamExecution$$runStream(StreamExecution.scala:28
4)
        at org.apache.spark.sql.execution.streaming.StreamExecution$$anon$1.run(
StreamExecution.scala:207)
Caused by: java.lang.ClassNotFoundException:
org.apache.kafka.clients.admin.OffsetSpec
        … 58 more
```

```
---------------------------------------------------------------------------
StreamingQueryException                   Traceback (most recent call last)
Cell In[13], line 23
     13 ds_accelerations = df_accelerations \
     14   .selectExpr("CAST(value AS STRING)") \
     15   .writeStream \
   (…)
     19   .option("checkpointLocation", accelerations_checkpoint_dir) \
     20   .start()
     22 try:
---> 23     ds_locations.awaitTermination()
     24     ds_accelerations.awaitTermination()
     25 except KeyboardInterrupt:

File /opt/conda/lib/python3.10/site-packages/pyspark/sql/streaming/query.py:201 ␣
 ↪in StreamingQuery.awaitTermination(self, timeout)
    199     return self._jsq.awaitTermination(int(timeout * 1000))
    200 else:
--> 201     return self._jsq.awaitTermination()

File /opt/conda/lib/python3.10/site-packages/py4j/java_gateway.py:1322, in␣
 ↪JavaMember.__call__(self, *args)
   1316 command = proto.CALL_COMMAND_NAME +\
   1317     self.command_header +\
   1318     args_command +\
   1319     proto.END_COMMAND_PART
   1321 answer = self.gateway_client.send_command(command)
-> 1322 return_value = get_return_value(
   1323     answer, self.gateway_client, self.target_id, self.name)
   1325 for temp_arg in temp_args:
   1326     if hasattr(temp_arg, "_detach"):

File /opt/conda/lib/python3.10/site-packages/pyspark/errors/exceptions/captured
 ↪py:175, in capture_sql_exception.<locals>.deco(*a, **kw)
    171 converted = convert_exception(e.java_exception)
    172 if not isinstance(converted, UnknownException):
    173     # Hide where the exception came from that shows a non-Pythonic
    174     # JVM exception message.
--> 175     raise converted from None
    176 else:
    177     raise

StreamingQueryException: [STREAM_FAILED] Query [id =␣
 ↪1f6412a2-7f18-4af9-a218-3499e16d1422, runId =␣
 ↪77b8b955-4e7c-4a03-afe6-5342deb06cf3] terminated with exception: org/apache/
 ↪kafka/clients/admin/OffsetSpec
```

[12]:

```python
print(ds_locations)
```

```
<pyspark.sql.streaming.query.StreamingQuery object at 0x7f083304d030>
```

[13]:

```python
print(type(ds_locations))
```

```
<class 'pyspark.sql.streaming.query.StreamingQuery'>
```

[14]:

```python
print(ds_accelerations)
```

```
<pyspark.sql.streaming.query.StreamingQuery object at 0x7f083304e500>
```

[15]:

```python
print(type(ds_accelerations))
```

```
<class 'pyspark.sql.streaming.query.StreamingQuery'>
```