Spark is being integrated with the cloud data platform in the modern data world. Manipulating data with Spark became curial to any data persona like data engineers, data scientists, and data analysts.

Last time, we will cover a similar topic about using Pyspark to read and write **streaming** data using [Spark Structured Streaming](https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html) through readStream and writeStream.

In this article, we will learn:

* how to read the stream data using Pyspark
* how to sink the stream data using Pyspark
* examples on reading/writing the streaming data using Pyspark on Databricks

**Basic Concepts on Streaming data**

**Streaming data** is data that is **continuously** generated by different sources, and such data should be processed incrementally using [stream processing](https://en.wikipedia.org/wiki/Stream_processing) techniques without having access to all of the data.

[Structured Streaming](https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html) is a scalable and fault-tolerant stream processing engine built on the Spark SQL engine. You can express your streaming computation the same way you would define a batch computation on static data. The Spark SQL engine will take care of running it incrementally and continuously and updating the final result as streaming data continues to arrive. The critical idea in Structured Streaming is to treat a live data stream as a table that is being continuously appended. This leads to a new stream processing model that is very similar to a batch processing model.

**Types of streaming sources**

1. File/Directory
2. Delta Files
3. Kafka
4. Other Connectors

**Types of sinks**

1. File/Directory
2. Delta table
3. Kafka
4. Foreach
5. Other connectors

A diagram of a stream

Description automatically generated

A diagram of a stream

Description automatically generatedA diagram of a stream

Description automatically generated

A diagram of a diagram

Description automatically generated

A screenshot of a computer

Description automatically generated

***Spark streaming execution plan***

*Apache Spark Streaming execution involves a series of steps, from receiving data to processing it in batches. The streaming application starts by ingesting data from a source, such as Kafka, Flume, or custom receivers. we use read stream to read data from a landing zone or from a streaming source.*

*We then apply some transformations, do some processing, prepare the result and use right stream to write it to a table or a streaming sink, this is known as the high level spark execution plan.*

***How spark engine will execute this streaming execution plan ?***

*A spark will start a background thread to manage and coordinate the execution of this streaming execution plan.****This background thread is known as streaming query of this execution pla***

***Each streaming execution plan will have its own streaming query***

*A logo with black text

Description automatically generated*

*Once the Streaming execution plan is ready, A spark driver will start a streaming query for that execution plan, and as soon as the streaming query starts, it will prepare the checkpoint location.****Checkpoint location is nothing but a directory location where streaming query will keep some housekeeping information.****So the first thing that is streaming query will do is to initialize the checkpoint location and then it goes and looks into the streaming data source and look into the location and check if there are data files available for processing, If there are no data files available for processing, streaming Query will keep a watch on the landing zone directory and wait.*

*Once the data is available in the landing zone, what this streaming query will do, It will note down the file name and update that information in the checkpoint location and then it will trigger the Micro-batch.****Each trigger or each execution of this execution plan is known as batch.****Stream will write the result into a table or a streaming sink.*

A computer code with text

Description automatically generated