Day 16 - Spark Structured Streaming

Date – 30th June 2025

# 1. Introduction to Streaming

Data streaming is the continuous transmission of data, allowing it to be processed in near real-time. This is essential for applications that handle live feeds from sources such as IoT devices, financial systems, social media, etc. Streaming enables timely decision-making and responsiveness in data-driven systems.

# 2. What is Spark Structured Streaming?

Apache Spark Structured Streaming is a scalable and fault-tolerant stream processing engine built on the Spark SQL engine. It allows developers to express streaming computations the same way they would express batch computations on static data.

# 3. Core Concepts

## 3.1 StreamingContext

The entry point for Spark Streaming is the StreamingContext, which connects to a Spark cluster. It defines the batch interval (e.g., every 5 seconds), which is the time duration Spark uses to divide the incoming data stream.

## 3.2 DStreams (Discretized Streams)

DStream is the basic abstraction in Spark Streaming. It represents a continuous stream of data and is internally represented by a sequence of RDDs (Resilient Distributed Datasets), each corresponding to data received in a time slice.

## 3.3 Transformations on DStreams

Transformations such as map, flatMap, filter, reduce, and groupBy can be applied to DStreams. These operations are similar to those on RDDs and are applied to each RDD in the DStream.

## 3.4 Output Operations

Output operations allow the processed data to be stored to external systems. Common operations include print(), saveAsTextFiles(), and foreachRDD(), which enable writing data to files, databases, or dashboards.

## 3.5 Checkpointing

Checkpointing is the process of saving intermediate data for fault tolerance. It is crucial in stateful transformations and when data needs to be recovered after failure.

# 4. Implementation Workflow in Databricks

## 4.1 Folder Setup

The following directories were created manually in DBFS:  
- /FileStore/tables/stream\_read/ (for input CSVs)  
- /FileStore/tables/stream\_write/ (for output results)  
- /FileStore/tables/stream\_checkpoint/ (for checkpointing)

## 4.2 Reading Streaming Data

CSV files are read as streaming data using a defined schema with readStream. The data includes fields like File, Shop, and Sale\_count.

## 4.3 Transformation & Aggregation

Data is grouped by Shop and the sum of Sale\_count is calculated in real-time using groupBy and sum operations.

## 4.4 Writing Stream Output

The aggregated streaming data is written to Parquet files in the stream\_write folder. OutputMode 'append' or 'complete' is used along with checkpointing to ensure reliable processing.

## 4.5 Output Validation

The output Parquet files are read using spark.read.parquet() to verify the results.

# 5. Conclusion

Spark Structured Streaming provides a powerful abstraction for building scalable real-time data pipelines. With its integration into Databricks, it becomes even easier to handle continuous data ingestion, transformation, and output for production-grade applications.