Name : Neeraj Rijhwani Roll No. 64 Class:D15C

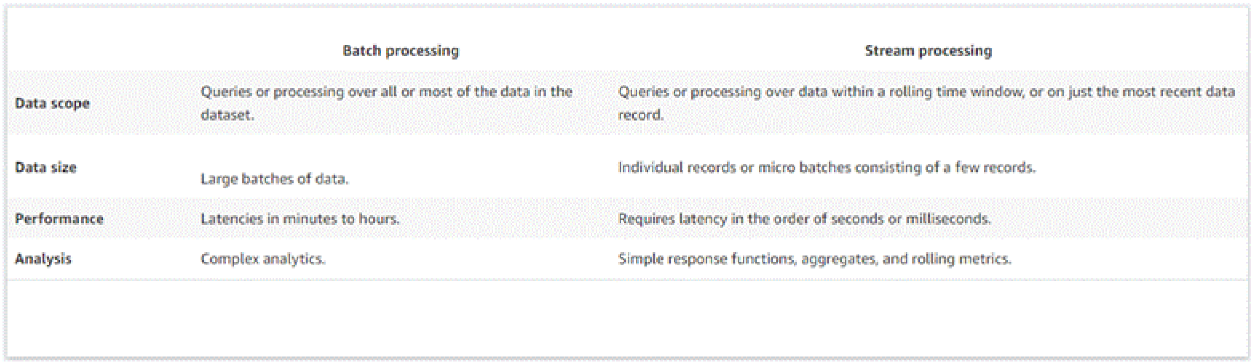
Experiment : 10

<u>Aim</u>:
To perform Batch and Streamed Data Analysis using Apache Spark.

<u>Introduction</u>:
Apache Spark is a powerful open-source data processing engine built for speed, ease of use, and sophisticated analytics. It supports both **batch processing** and **stream processing**, making it an ideal choice for handling large-scale data workloads.

- **Batch processing** deals with processing large volumes of data that are collected over a period. It's used when data doesn't need to be processed in real time.

- **Stream processing** involves real-time data processing, allowing organizations to analyze and react to data as it arrives.



	Batch processing	Stream processing
Data scope	Queries or processing over all or most of the data in the dataset.	Queries or processing over data within a rolling time window, or on just the most recent data record.
Data size	Large batches of data.	Individual records or micro batches consisting of a few records.
Performance	Latencies in minutes to hours.	Requires latency in the order of seconds or milliseconds.
Analysis	Complex analytics.	Simple response functions, aggregates, and rolling metrics.

<u>Batch Processing Using Apache Spark</u>

In batch processing, data is collected, entered, and processed in groups or batches. Apache Spark processes this data by loading it into memory, applying transformations, and then performing actions to generate outputs. It is commonly used for:

- Word counts from large text datasets
- Aggregation operations (e.g., sum, average)
- Log analysis
- ETL (Extract, Transform, Load) processes

<u>Streamed Data Analysis Using Apache Spark</u>

Name : Neeraj Rijhwani Roll No. 64 Class:D15C

Big data streaming is the process of analyzing data in real time as it flows into the system. Apache Spark's **Structured Streaming** API enables scalable and fault-tolerant stream processing of live data.

Key Benefits of Streaming Data:

- **Real-time insights**: Enables instant decision-making based on current data.
- **Applicability**: Suitable for industries like finance, media, energy, e-commerce, and gaming.
- **Scalability**: Can handle high-throughput and low-latency data streams.
- **Advanced analytics**: Can incorporate machine learning models, time-windowed aggregations, and trend analysis.

Real-world Examples:

- **Financial sector**: Monitoring stock market trends and managing portfolios in real time.
- **Real estate apps**: Delivering property suggestions based on a user's location.
- **Solar energy companies**: Monitoring and maintaining power panels live to avoid performance penalties.
- **Media companies**: Analyzing user clickstreams to optimize content delivery.
- **Online gaming**: Reacting to player actions instantly to enhance user experience.

Challenges:

- Managing data security and privacy
- Handling high-frequency data without system lag

- Designing efficient pipelines for real-time operations

<u>Approach</u>:

Approach to count the words using Spark:

1. Let's create an RDD by using the following command
   data = sc.textFile("file_name.txt")

2. Here, pass any filename that contains the data. Now, we can read the generated result by using the following command.
   data.collect

3. Here, we split the existing data in the form of individual words by using the following command.
   splitdata= book.flatMap(lambda x: x.split()).countByValue()

4. Now, we can read the generated result by using the following command.
   splitdata.collect

5. Now, perform the map operation.
   ```
   for i, (word, count) in enumerate(word_counts.items()):
       if i == 100: break
       print(word, count)
   ```

   Here, we are assigning a value 1 to each word. Now, we can read the generated result by running the for loop.

6. Now, perform the reduce operation if needed.
   reducedata = mapdata.reduceByKey(lambda a,b : a+b)

   Here, we are summarizing the generated data.

Name : Neeraj Rijhwani Roll No. 64 Class:D15C

Conclusion:
Thus, we have learnt what batch and stream processing is and also learnt how to implement it using Apache Spark.