NAME: SHIVAM MAKHIJA ROLL NO:32 D15C

Experiment No: 10

Aim: To perform Batch and Streamed Data Analysis using Apache Spark.

Theory:

1. What is Streaming? Explain Batch and Stream Data.

Streaming is a method of continuously transmitting data, often used in scenarios where real-time or near-real-time processing is needed. It allows data to be processed as soon as it is generated, without waiting for the entire dataset to be collected. This is especially useful in applications like online gaming, social media feeds, fraud detection systems, and IoT sensor monitoring, where immediate insights and actions are required.

Batch data processing involves collecting and storing large amounts of data over a period of time and then processing it all at once. It is suitable for operations that do not require instant results, such as generating end-of-day sales reports, monthly billing statements, or performing data analysis on historical records. Batch processing is often simpler to implement but has higher latency because it waits for the complete data before starting the computation.

On the other hand, stream data processing handles data in real-time, processing each piece as it arrives. This allows for immediate reactions to data changes, making it ideal for time-sensitive applications like financial trading, recommendation engines, and live analytics dashboards. While more complex to set up, stream processing enables businesses to act quickly on fresh data and gain competitive advantages through instant decision-making.

2. How Data Streaming Takes Place Using Apache Spark.

Apache Spark enables data streaming through its component called Spark Structured Streaming, which allows real-time data processing using the same APIs as batch jobs. It works by continuously ingesting data from sources like Kafka, Flume, or sockets, treating the incoming stream as an unbounded table that is constantly being updated. The data is processed in micro-batches (small batches created at short intervals), allowing developers to use SQL-like queries, aggregations, and transformations in real time.

Spark Structured Streaming abstracts the complexity of real-time systems by managing fault tolerance, scalability, and stateful processing under the hood. Once the streaming computations are defined, Spark continuously monitors the data sources, applies the transformations, and writes the output to destinations like databases, file systems, or dashboards. This makes it powerful for applications like fraud detection, live data analytics, and monitoring systems, all with minimal latency and high throughput.

Conclusion:

Apache Spark efficiently handles both batch and stream processing through its unified engine. While batch processing is suitable for historical data analysis, stream processing enables real-time analytics on continuously incoming data. Spark's Structured Streaming model simplifies real-time data handling using familiar APIs and ensures fault-tolerant and scalable performance. Through this experiment, the practical understanding of how Apache Spark processes both static and dynamic data was achieved, highlighting its significance in real-world big data applications.