**IN CLASS ACTIVITY-1**

****

**GROUP MEMBERS**

1. **MIHIR MATHUR- (C0806779)**
2. **JAPAN PATEL-(C0805745)**
3. **DHAIRYA SHAH-(C0800579)**
4. **RAVI PATEL-(C0791604)**

**DATA STREAMING**

**Introduction**

The term streaming is employed to explain continuous, unending data streams with no starting or end that gives a relentless feed that may be utilized/acted upon with no need to be downloaded first. Similarly, data streams are generated by all sorts of supplies in various formats and sizes. For examples like applications, networking devices, and server log files, website activity, banking transactions, and site data, they will all be collective to seamlessly gather time period information and analytics from one source of truth.

**Easy Data Scalability in Data Streaming**

In data streaming, growing data volumes will break an execution system, requiring you to provision additional resources or modify the architecture. Modern stream processing infrastructure is hyper-scalable, ready to handle Gigabytes of information per second with one stream processor. This enables you to simply deal with growing data volumes while not infrastructure changes. When system failures happen, log data returning from every device may increase from being sent a rate of kilobits per second to megabits per second and collective to be gigabits per second. Adding additional capacity, resources and servers as applications scale happen instantly, exponentially increasing the number of information generated. Coming up with applications to scale is crucial in operating with streaming data.

**Batch Processing vs Real-Time Streams**

The batch data processing method needs data to be downloaded as batches before it is processed, stored, or analyzed, whereas streaming data flows in continuously, permitting that data to be processed simultaneously, in a period of time the second it's generated. Today, data arrives naturally as endless streams of events. The data arrives in all volumes, formats, from various locations and cloud, on-premises, or hybrid cloud.

With the complexness of today's modern requirements, traditional data processing methods have become obsolete for many use cases because it solely processes data as groups of transactions collected over time. Organizations have to act on up-to-the-millisecond data before the data becomes stale. This continuous data offers various advantages that are remodelling the way a business run.

Use cases for batch processing include Payroll, Billing and Order from Customers. While, Stream processing includes Fraud detection, Social media analysis and Log monitoring.

**Real-Time Query in streaming data Warehouse**

Traditional streaming analytics is too shallow, and traditional data warehouses take too long to load data and query it after the fact. With a streaming data warehouse, we can analyze as fast as we can stream, with up-to-the-second results and event detection with near-zero latency. Deployment of machine learning models for real-time analysis needs infrastructure which will calculate options with streaming information in real-time, and geospatial analytics and visualisation are reducing accuracy. A streaming data warehouse does machine learning at a large scale and displays up-to-date made geospatial data at interactive speed.

**Conclusion**

Nowadays, several applications need to handle streaming data, which is huge and arriving rapidly, like the sensor network, traffic monitoring and Internet. Different from traditional database management systems (DBMS), the goal of streaming data management is to plan one-pass, space- and time-efficient solutions to implement specific querying and analyzing tasks. Additionally, data uncertainty also widely exists in such applications. Dealing with an uncertain data stream is more challenging than dealing with a deterministic data stream due to the exponential growth of the possible world instances to the data volume.

**References:**

<https://www.kinetica.com/products/why-streaming-data-warehouse/>

<https://www.precisely.com/blog/big-data/big-data-101-batch-stream-processing/>

<https://www.upsolver.com/blog/streaming-data-architecture-key-components/>

<https://www.confluent.io/learn/data-streaming/>