Cloud Computing Discussion 1

Name - Ravi Rajpurohit **UTA ID - 1002079916**

Ques 1 - When would you use a batch system vs a streaming system?
Ans - Batch system is a type of data processing system technique for processing large amount of data consistently. It allows to process data with less user interaction whenever resources are available. This method is most commonly used when the data is homogenous and in large quantities.
Benefits of batch system are -

- Simplicity and Efficiency
- Better Data Quality
- Fast Business Intelligence

Stream processing is the data processing system in which action is taken as the data is being generated. It is also known as real time processing system. As the name clarifies, data is processed as frequently as required for any use case. For example - processing sensor data for a real time traffic monitoring system to detect high traffic.
Benefits of stream system are -

- Time taken for processing is minimal
- The information is real time and is ready to use
- Fewer resources required
- Can identify problems immediately

Ques 2 - What about the Lambda architecture?
Ans - Lambda architecture is one of the well known data-processing architectures specifically designed to handle huge amount of data by taking advantage of batch and stream processing methods.
By providing accurate view of batch data using the batch processing, this architecture provides benefits like balanced latency, better throughput and fault tolerance. Simultaneously, it provides view of online data by using real time stream processing.

Ques 3 - What is the purpose of the windowing model in Google Cloud Dataflow?
Ans - Dividing unbounded collections into windows i.e. logical components is called windowing model. Sharing data through automated manner, like every midnight, sometimes the data is large and problems like losing connection might happen while sharing the data. To tackle this problem, we can use windowing model. The idea is to divide the data into time based windows. Each data item must have a timestamp so that it is possible to know which window to put the data item in.

Ques 4 - Why were different types of windowing patterns required for different applications?
Ans - There are mainly 3 types of windowing -

- **Fixed window** - represents a disjoint, consistent time interval in the data stream
  For example, if the window size is 10 seconds, the elements with timestamp values [00-10) are in the first window, the elements with timestamp values [00-20) are in the second window.
- **Sliding window** - represents windows of fixed interval duration that are uniform across all the keys. Also, there is an overlap between two or more windows. A part of previous window is common with the next windows.
  For example, moving average of data.
- **Session window** - represents windows of intervals that are set dynamically. It can vary across keys. Also, there is no overlap between two windows.
  For example, real time gaming data analysis, user session data.

Ques 5 - What is the purpose of triggering in Dataflow? How is it related to/different from the windowing models?
Ans - Triggers are used to emit aggregated results as the data is available. The default point for emitting results is when the watermark passes the window's end. For example, dataflow triggers can be used to start a job after a file is written to the filesystem by the data pipeline.

Triggers are in action after the data has been windowed. While processing, the windowing models are used to make logical components, i.e. windows, of data in the required manner. The manner of windowing can be a fixed window, sliding window or session window as suited per the use case. Once the data is sectioned, the watermark (threshold that indicates when in a window, all the data has arrived) passes the end of the window and events are triggered.

Hence, triggers are followed by windowing of data to perform the required action. For different windowing models, triggering depends on the windowing condition and the watermark. Following are a few types -

- Early Firing: when trigger is fired before the end of window and there is still time to receive the late data. The output is available with a warning of it being early results and possibility of change
- On-Time Firing: when trigger is fired upon the closure of watermark and the computation is triggered
- Late Firing: when trigger is fired after observing late data for late data comutation

Ques 6 - Is event-time windowing that important?
Ans - Yes, event-time windowing is important. It provides the most accurate results across the other options. A timestamp is assigned at the moment of the event or when the message is generated at the source. Across all time related operations, this timestamp will be followed. Event time is supported and handled by Watermarks in Apache Flink.
For example, a window operator building hourly windows is required to be to be alerted when event time has passed beyond the limit so that the window in progress can be closed.