Project Milestone-- Data Ingestion Software-- Kafka Clusters

Kaushik Ramani
100651855

14/2/22

What is EDA? What are its advantages and disadvantages?
EDA stands for Event Driven Architecture. Event-driven architecture enables loose coupling, which makes it a good option for distributed application architectures. This differs from a traditional request-driven model. Event Driven Architecture uses the pub-sub model (publish/subscribe). There are alot of benefits to this approach:

- **Fault tolerance**. Since services and resources are distributed and decoupled in EDA. Due to this, the faults are also isolated to the individual services. This means that communications are queued in a broker until the faults are resolved.
- **Loose coupling**. This means that services do not need to be aware of each other. This allows for asynchronous communication as there would be an event broker that will retain all messages.

Now, with this also comes caveats and disadvantages when using EDA. Some of the things to consider are:

- Increased complexity. In EDA, the API backend does all the work. This means that the producers take on the responsibility to maintain state and keep track of all services and state changes. Events also have to be published with high availability and reliability.
- In a pub-sub model, there is no clear workflow order. This is due to the fact that each step in EDA is only triggered when a broker is alerted to a particular event. This means that if a service or component is not working correctly or responding to the event, it might create workflow issues and more errors down the line.

In Kafka, what's meant by cluster, broker, topic, replica, partition, zookeeper, controller, leader, consumer, producer, and consumer group?
Cluster: This is just a group of brokers in Kafka
Broker: Brokers allow consumers to fetch messages by either topics, partitions, and/or offset
Topic: Topics are basically titles that are used to organize data within Kafka. Topics are stored in **Brokers**. A topic can be partitioned and stored in multiple brokers if need be
Replica: Replicas are copies of topic partitions
Partition: Partitions are pieces or shards of topics. This is for data parity and scalability.
Zookeeper: used for service synchronization and controller elections
Controller: Controllers are just fancy brokers that manage brokers. They are used for managing partitions, and broker elections
Leader: The leader broker is where the consumers read from and where the producers write to. Due to this, the leader will always have the most recent data on all topics and partitions in it.
Consumer: Responsible for reading data that is stored in Kafka
Producer: Applications that write to the kafka lead broker
Consumer group: Are a group of consumers that are subscribed to the same topic