What is the Apache Kafka?

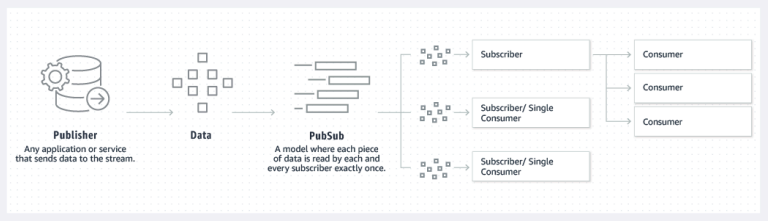
Kafka is an open source software which provides a framework for storing, reading and analysing streaming data.

Kafka is designed to be run in a “distributed” environment, which means that rather than sitting on one user’s computer,it runs across several (or many) servers, leveraging the additional processing power and storage capacity that this brings.

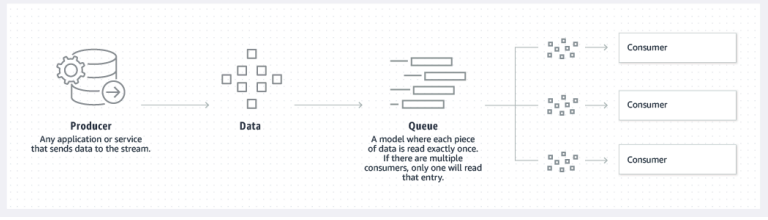
1. How Does Apache Kafka work?

Kafka combines two messaging models, queuing and publish-subscribe, to provide the key benefits of each to consumers.

Pub-Sub Modelling



Queue Modelling



Source: https://aws.amazon.com/tr/msk/what-is-kafka/

1.1. Where was the Apache Kafka first used?

Kafka was originally designed to track the behaviour of visitors to large, busy websites (such as LinkedIn).

By analysing the clickstream data (how the user navigates the site and what functionality they use) of every session, a greater understanding of user behaviour is achievable.

1.2. What is the Kafka Basic Design Principles?

- Persistent messaging
- High throughput
- Distributed
- Real time

1.3. What are the features of Apache Kafka?

- Scalability
- Fault Tolerance
- Reliability
- Durability
- Performance
- High-Volume
- Zero Downtime

2. Main Components of Apache Kafka

What is the Brokers?

A Kafka server or Kafka Node is known as a Kafka broker. We can setup a Kafka cluster with one or many brokers. In real life broker is a third person stand in between a buyer and a seller. In the case of Kafka modelling, broker plays the same role but in between Producer and Consumer. A Kafka broker receives messages from producers and stores them on disk keyed by unique offset.

Kafka broker hosts topics with events in it. Topics can have one or more partitions.

A Kafka broker allows consumers to fetch messages by topic, partition and offset.

Kafka brokers can create a Kafka cluster by sharing information between each other directly or indirectly using Zookeeper.

What is the Zookeeper?

Application that provides a coordinator service for distributed structures.

- What are their duties?
  - Controller Election= The controller is the broker responsible for maintaining the leader/follower relationship for all partitions. If ever a node shuts down, ZooKeeper ensures that other replicas take up the role of partition leaders replacing the partition leaders in the node that is shutting down.
  - Cluster Membership= ZooKeeper keeps a list of all functioning brokers in the cluster.
  - Topic Configuration= ZooKeeper maintains the configuration of all topics, including the list of existing topics, number of partitions for each topic, location of the replicas, configuration overrides for topics, preferred leader node, among other details.
  - Access Control Lists (ACLs)= ZooKeeper also maintains the ACLs for all topics. This includes who or what is allowed to read/write to each topic, list of consumer groups, members of the groups, and the most recent offset each consumer group received from each partition.
  - Quotas= ZooKeeper accesses how much data each client is allowed to read/write.

What is the Producer?

Producers publish data to appropriate topics. They have the responsibility to choose topics and partition topics. Producer sends data as records and each record contains key and value pair so it converts data to byte array with the help of Key Serializer and Value Serializer. By default, partitioner chooses partition number by hash key or it can be done in a round-robin fashion. It has various approaches to send data to the server.

What is the Consumer?

Consumers read and process the data from appropriate topics within the Kafka cluster. Consumers are labeled with consumer group names. Those which have the same consumer group name for multiple consumers are called consumer groups. Kafka cluster delivers each record from the topics to single consumer

instant of the consumer group. If each consumer instant has a different group name, then records are delivered to all consumer instants. Each consumer instant can run on a different process or different machine.

What is the Topic in Apache Kafka?

Kafka topics are the categories used to organize messages. Each topic has a name that is unique across the entire Kafka cluster.

Messages are sent to and read from specific topics. In other words, producers write data to topics, and consumers read data from topics.

Kafka topics are multi-subscriber. This means that a topic can have zero, one, or multiple consumers subscribing to that topic and the data written to it.

In Kafka, topics are partitioned and replicated across brokers throughout the implementation. Brokers refer to each of the nodes in a Kafka cluster. The partitions are important because they enable parallelization of topics, enabling high message throughput.

What is the Partition in Apache Kafka?

Kafka topics are further split in to partitions. These partitions are replicated across the brokers in a Kafka cluster. Data is written to partitions randomly unless a key is added to it. No limitations to partition the topic. Messages are stored in sequenced fashion in one partition. Each message in a partition is assigned an incremental id, also called offset. We can define replication factor to increase the availability. Partitions can have copies to increase durability and availability and enable Kafka to failover to a broker with a replica of the partition if the broker with the leader partition fails.

What is the Replication in Apache Kafka?

Replication is the process of having multiple copies of the data available across different servers for purpose of availability in case one of the brokers goes down. In Kafka, replication happens at the partition level i.e. copies of the partition are maintained at multiple broker instances.

When we say a topic has a replication factor of 3, this means we will be having three copies of each of its partitions. Kafka considers that a record is committed

when all replicas in the In-Sync Replica set (ISR) have confirmed that they have taking the record into account.

While creating a Kafka topic, we can define the number of copies we want to have for the data. We define this using the replication-factor config setting.

What is the KafDrop?

Kafdrop is a web UI for viewing Kafka topics and browsing consumer groups. The tool displays information such as brokers, topics, partitions, consumers, and lets you view messages.