KAFKA

What is Kafka?

Kafka is open-source **distributed Message System** platform that uses **Publish** and **Subscribe** mechanism to stream records.

Originally developed by LinkdIn and later denoted to Apache foundation.

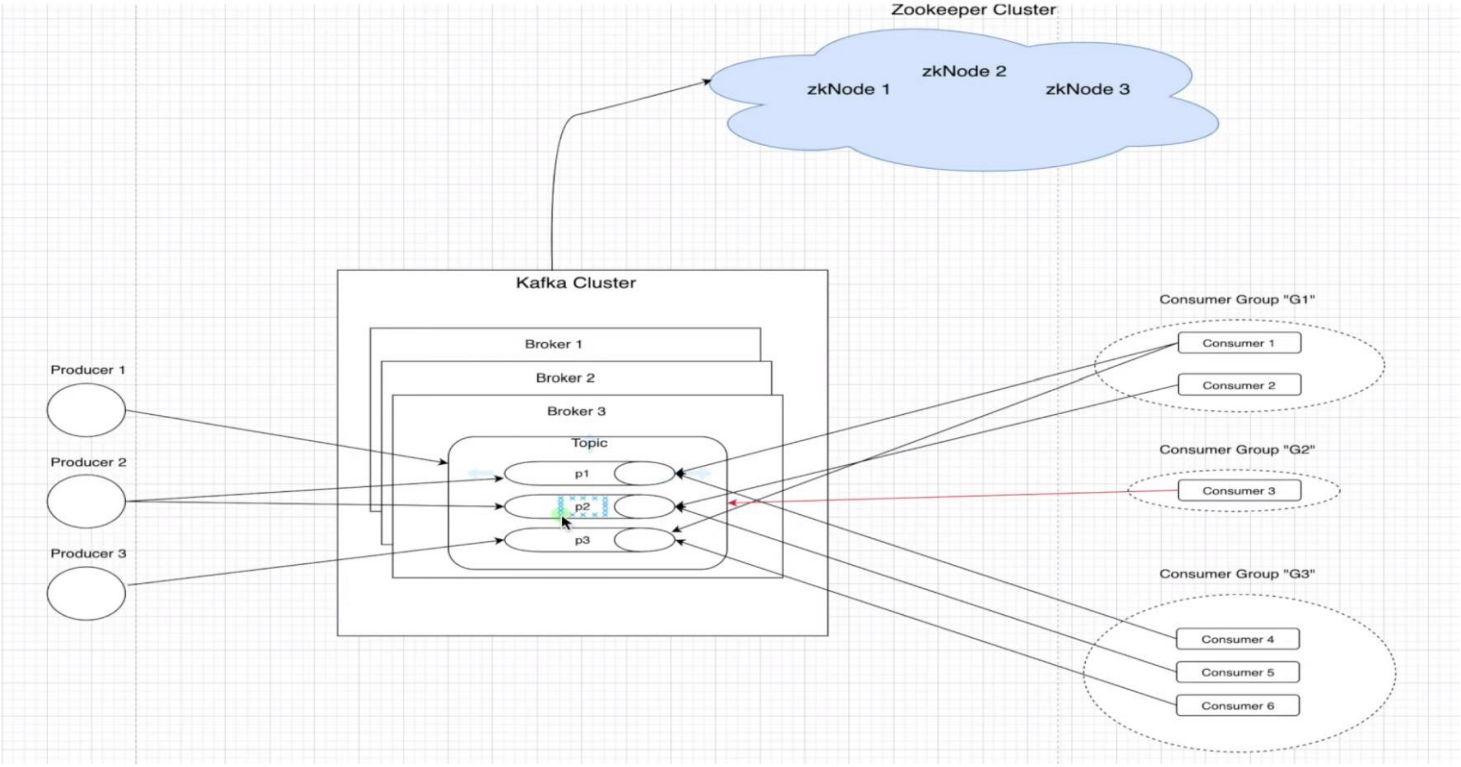
Messaging System:

A Messaging System is responsible for transferring data from one application to another so the application can focus on data without getting bogged down on data transmission and sharing.

There are two type of Messaging system:

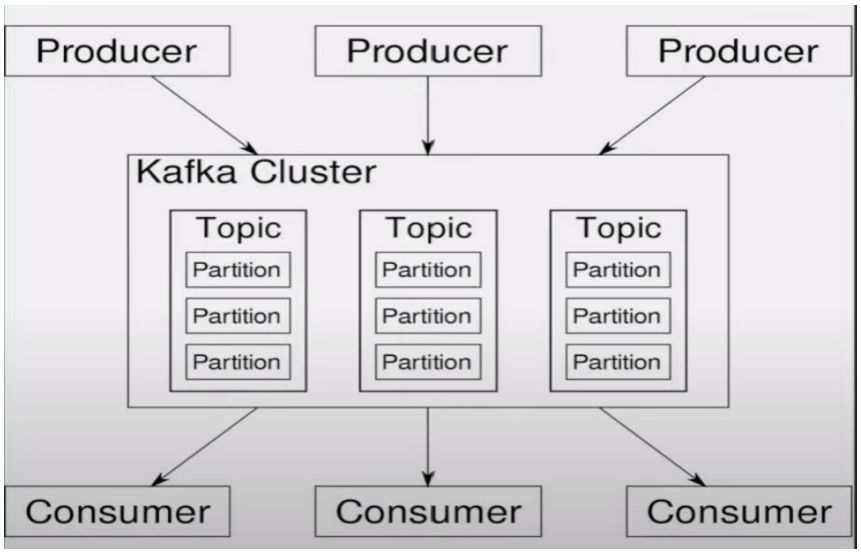
1. Point to Point messaging system
   - Message are persisted in a queue
   - A particular message can be consumed by a maximum of one receiver only
   - There is no time dependency laid for the receiver to receive the message
   - When the receiver receives the message, it will send the acknowledgement back to the sender
2. Publish-Subscribe Messaging System
   - Message are persisted in a **Topic**
   - A particular message can be consumed by any number of consumers
   - There is time dependency laid for the consumer to consumer the message
   - When the Subscriber receives the message, it does not send an acknowledgement back to the publisher

Kafka Architecture:



Topics:

A stream of message belonging to a particular category is called a topic. It is logical feed name to which record are published. Similar to a table in database (record are considered as message here).
Unique identifier of Topic is its **NAME**.
We can create as many Topics as we want.

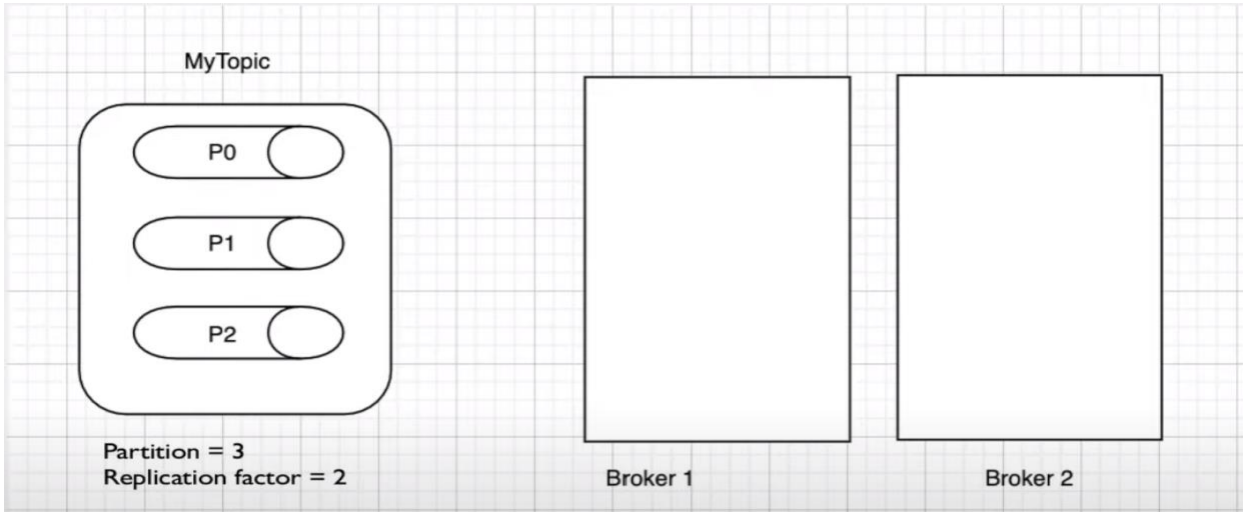


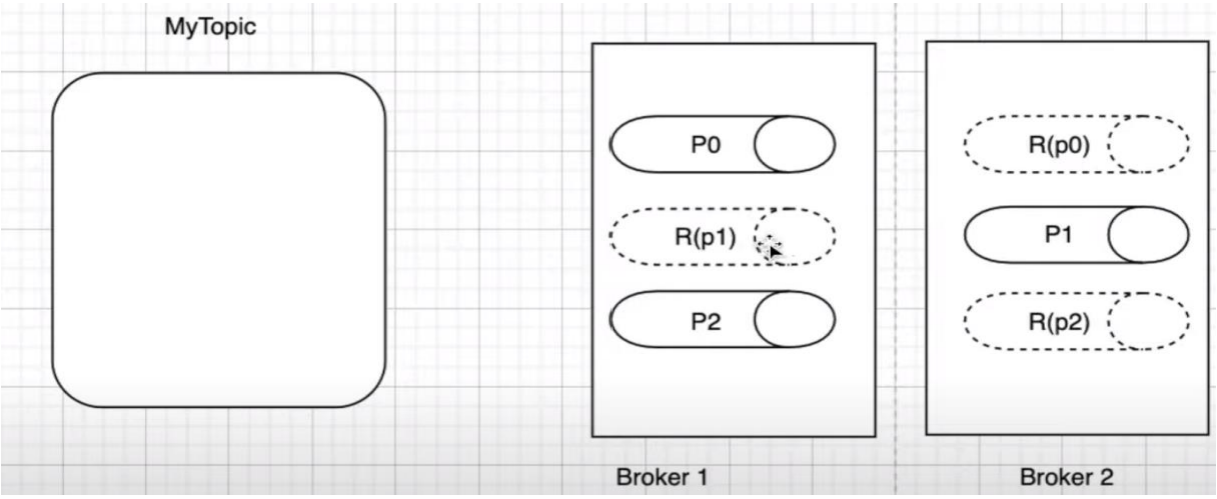
Partitions:

- Topics are split into partitions.
- All the message within Partitions is ordered and immutable
- Each message within a partition has a unique Id associated known as Offset

Replicas and Replication:

- Replicas are backup of partition
- Replicas are never read or write data
- They are used to prevent data loss (Fault-Tolerance)





Producers:

Producers are applications which write/push data/message to the Topics with in a cluster using producing APIs.

Producers can write data either on the Topic level (All the Partition of that Topic) or specific partition of that Topic.

Consumers:

Consumers are applications which read/consume data from the topics within a cluster using the Consuming APIs.

Consumers can read data either on the topic level (All the partition of that topic) or specific partitions of the topic.

Consumers are always associated with exactly one Consumer Group.

A Consumer Group is a group of related consumers that perform a task.

Brokers:

Brokers are simple software processes who maintain and manage the published messages

Also known as **Kafka servers.**

Brokers also manage the consumer-offsets and are responsible for the delivery of message to the right consumers.

A Set of brokers who are communicating with each other to perform the management and maintenance task are collectively known as Kafka Cluster.

We can add more brokers in a already running Kafka cluster without any downtime.

Zookeeper:

Zookeeper is used to monitor Kafka Cluster and co-ordinate with each broker.

Keeps all the metadata information related to Kafka cluster in the form of a key-value pair.

Metadata includes:

1. Configuration information
2. Health status of each broker

It is used for the controller election within Kafka cluster.

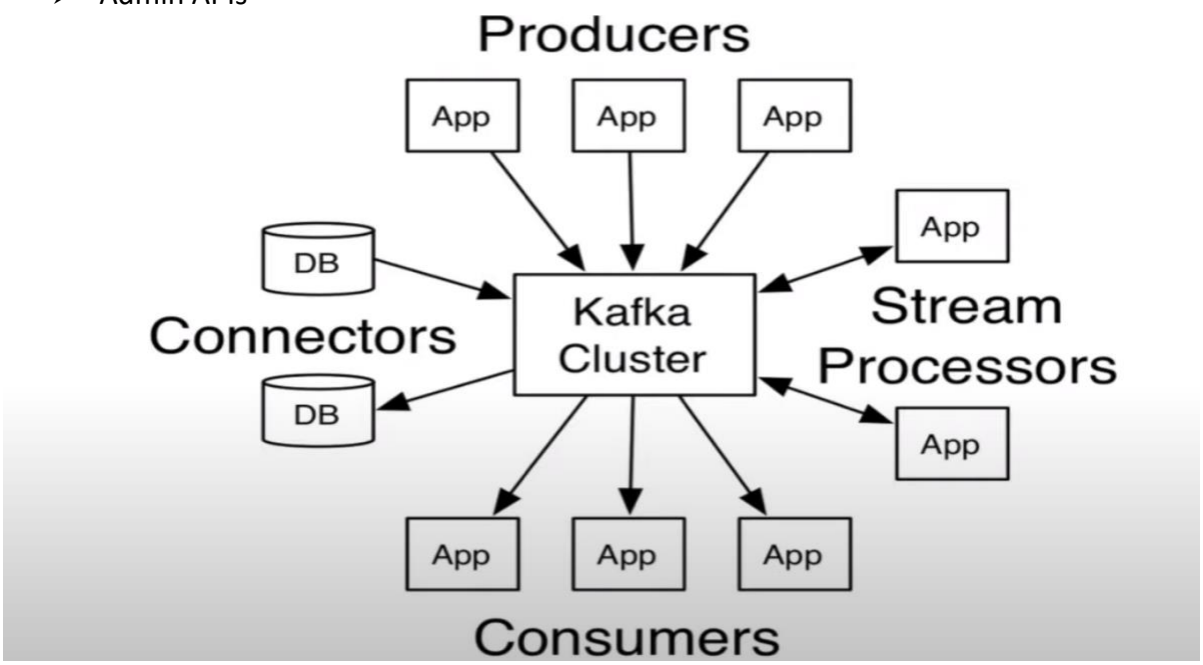
A set of Zookeepers nodes working together to manage other distributed system is known as **Zookeeper cluster** or **"Zookeeper Ensemble"**

Features of Kafka over other Messaging System:

1. **Scalable:**
   Horizontal Scaling is done by adding new brokers to the existing clusters.
2. **Fault Tolerance:**
   Kafka clusters can handle failures because of its distributed nature.
3. **Durable:**
   Kafka uses "Distributed commit log" which means messages persists on disk as fast as possible.
4. **Performance:**
   Kafka has high throughput for both publishing and subscribing messages.
5. **No Data loss:**
   It ensures no data loss if we configure it properly.
6. **Zero downtime:**
   It ensures zero downtime when required number of brokers are present in the cluster.
7. **Reliability:**
   Kafka is reliable because it provides above features.

Kafka Provides:

- Producer APIs
- Consumer APIs
- Stream APIs
- Connector APIs
- Admin APIs



https://www.youtube.com/watch?v=BkL5blN057M&list=PLxv3SnR5bZE82Cv4wozg2uZvaOlDEbO67&index=3