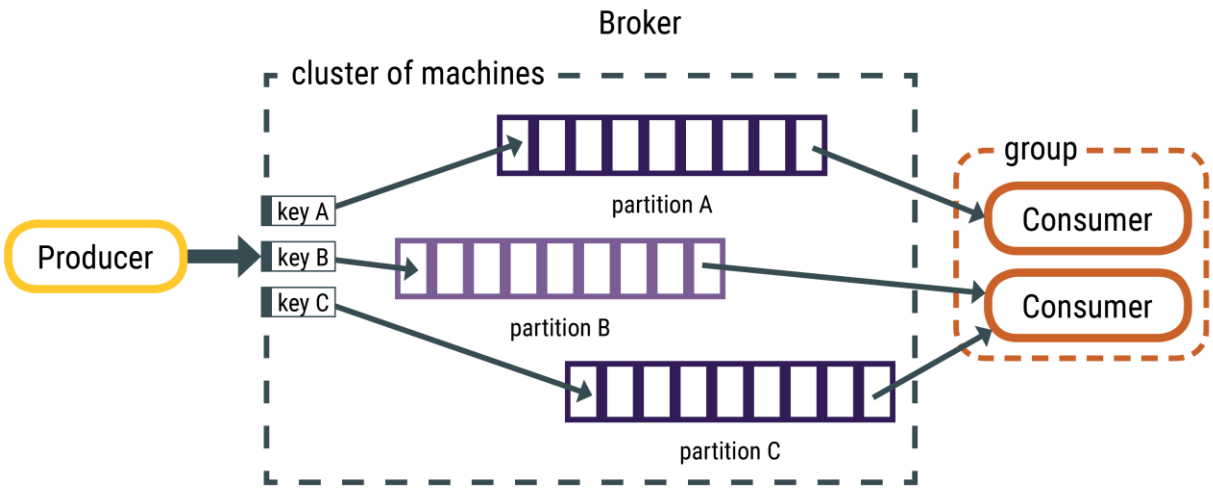
Kafka Partitions

Apache Kafka is an event streaming platform, or we can say that Kafka can handle the transportation of messages across multiple systems/microservices.

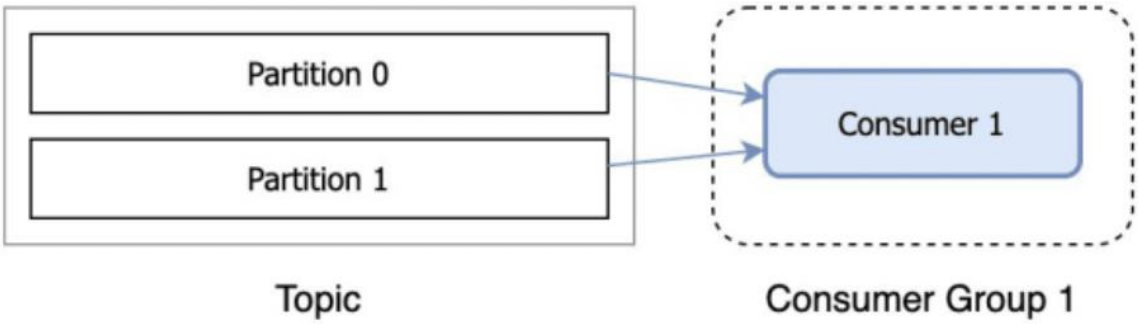
Apache Kafka is distributed as it relies on multiple servers and is also scalable, where we can start small and add servers per requirement.

Partitions and Consumer Groups in Apache Kafka

The topics on Apache Kafka Cluster are stored on servers called **brokers**. The topics are split into multiple pieces and stored across multiple machines. These chunks are called **partitions**. A Consumer Group is a group of consumers sharing the same *group_i* . When consumers consume a topic within the same group(having the same group_id), every message will be delivered to only one consumer.

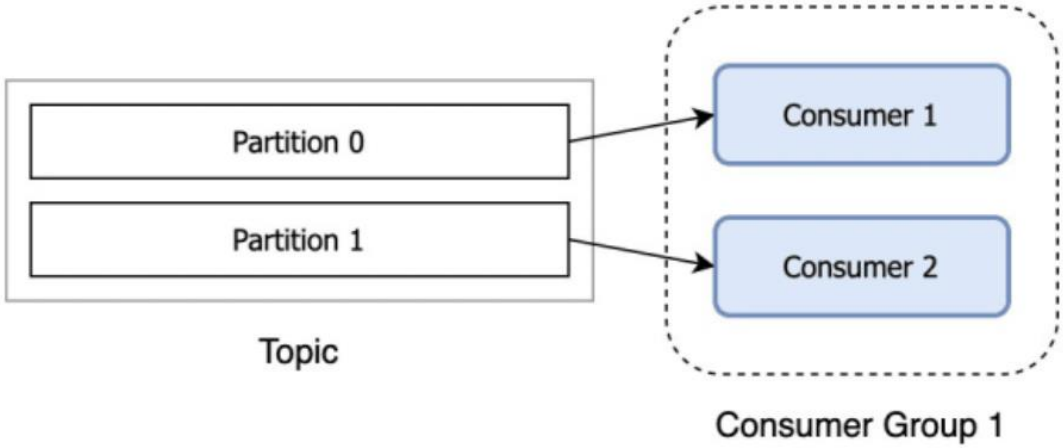


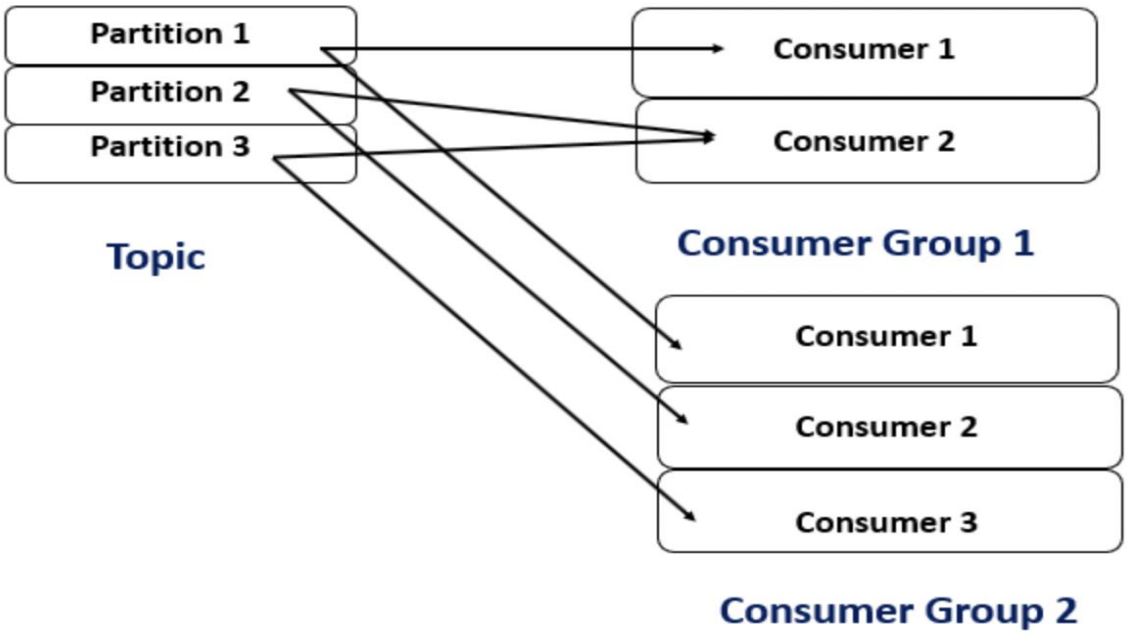
Kafka assigns the partitions of a topic to the consumers in a group so that each partition is assigned to one consumer in the group. This ensures that records are processed in parallel and nobody steps on other consumers' toes.



If we have two partitions and only one consumer, the consumer reads from both. On the other hand, if there are two partitions and two consumers, each consumer is assigned one partition.

If more consumers are in a group than the number of partitions, extra consumers will sit idle. When a consumer fails, the partitions assigned to it will be reassigned to another consumer in the same group.





Limits on partitions:

There are no hard limits on the number of partitions in Kafka clusters.

But here are a few general rules:

- maximum 4000 partitions per broker (in total; distributed over many topics)
- maximum 200,000 partitions per Kafka cluster (in total; distributed over many topics)
- resulting in a maximum of 50 brokers per Kafka cluster

Rules to follow:

- **Offsets only have a meaning for a specific partition**. That means offset number 3 in Partition 0 does not represent the same data or the same message as offset number 3 in partition 1.
- **Order is going to be guaranteed** only from within a partition.
- But across partitions, we have no ordering guarantee. So this is a very important certainty of Kafka is that you're going to have **ordered at the partition level only**.

- **Data in Kafka by default is kept only for a limited amount of time** and the default is one week. That means that after one week the data is going to be erased from a partition and this allows Kafka to keep on renewing its disk and to make sure it does not run out of disk space.
- Kafka is **immutable**. That means once the data is written into a partition, it cannot be changed. So if you write the message number 3 in partition 0 you cannot overwrite. So as such, you want to be careful about the kind of data you send to a Kafka topic and your recovery mechanism instead of in case you send bad data.
- Also if you don't provide a key to your message, then when you send a message to a Kafka topic the data is going to be assigned to a random partition.
- Finally, a topic can have as many partitions as you want but it is not it is common to have topics with say 10, 20, 30, or 1000 partitions unless you have a truly high throughput topic.