ORACLE Certified Professional

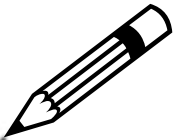
Oracle Certified Professional Java Programmer

Microsoft Certified Professional

sohailimran@yahoo.com

APACHE kafka™

A distributed streaming platform



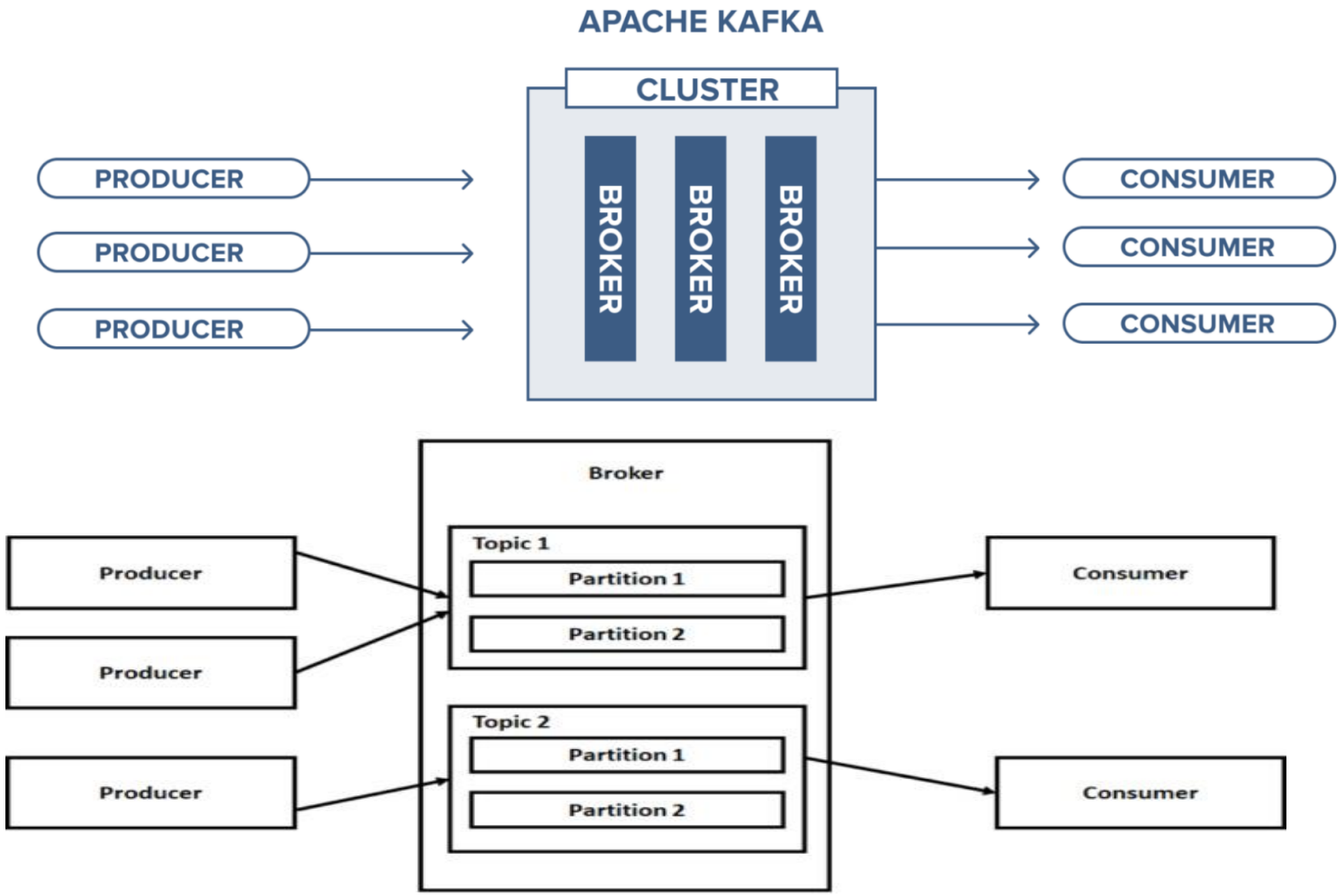
Introduction

Ref: Web

intro

- Kafka is a messaging system that is designed to be fast, scalable, and durable.
- A **producer** is an entity/application that publishes data to a Kafka cluster, which is made up of **brokers.**
- A **Broker** is responsible for receiving and storing the data when a producer publishes.
- A **consumer** then consumes data from a broker at a specified offset, i.e. position.
- A **Topic** is a category/feed name to which records are stored and published. Topics have partitions and order guaranteed per partitions
- All Kafka records are organized into topics. Producer applications write data to topics and consumer applications read from topics.

producer-broker-consumer

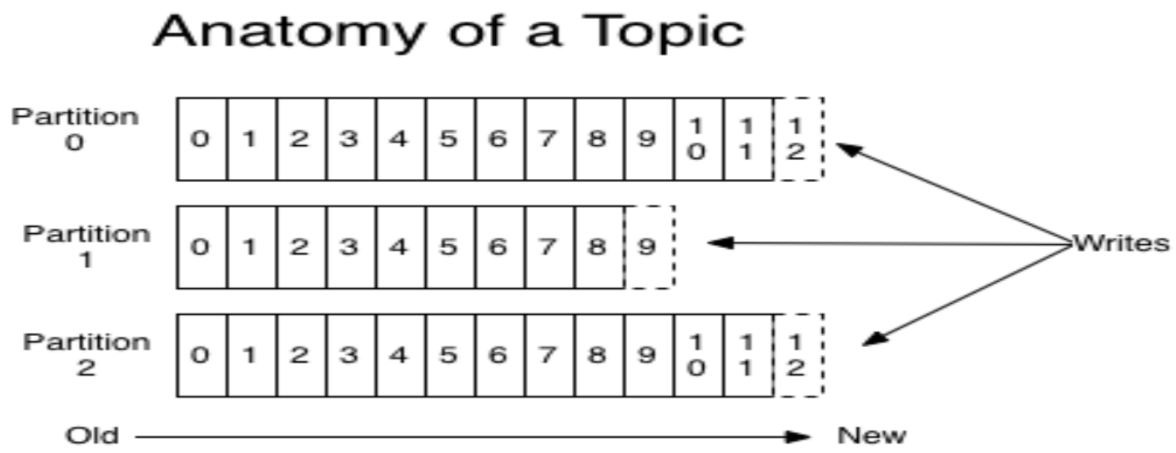


Key Concepts

- **Topic** is divided in partitions.
- The **message** order is only guarantee inside a partition
- **Consumer** offsets are persisted by Kafka with a commit/auto-commit mechanism.
- Consumers subscribes to topics
- Consumers with different group-id receives all messages of the topics they subscribe. They consume the messages at their own speed.
- Consumers sharing the same group-id will be assigned to one (or several) partition of the topics they subscribe. They only receive messages from their partitions. So a constraint appears here: the number of partitions in a topic gives the maximum number of parallel consumers.
- The assignment of partitions to consumer can be automatic and performed by Kafka (through **Zookeeper**). If a consumer stops polling or is too slow, a process call “re-balancing” is performed and the partitions are re-assigned to other consumers.

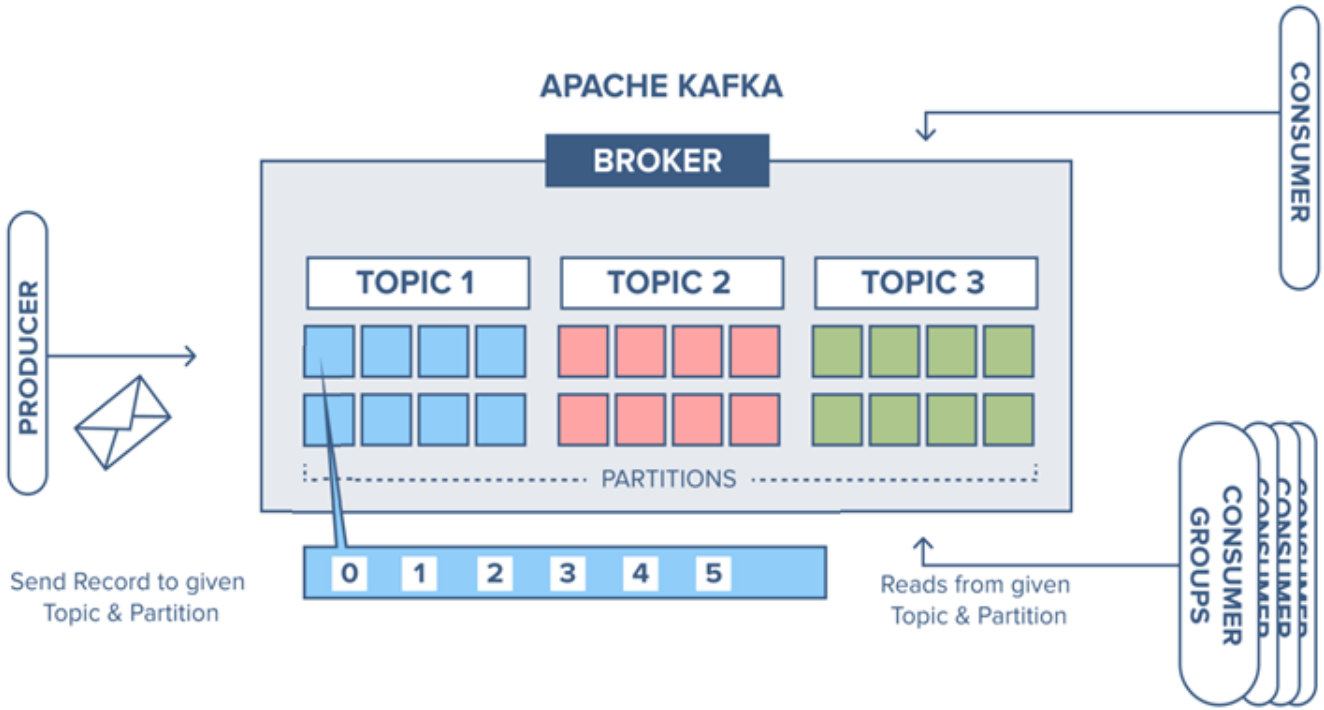
Key Concepts (contd..)

- Kafka normally divides topic in multiply **partitions**.
- Each partition is an ordered, immutable sequence of messages that is continually appended to.
- A message in a partition is identified by a sequence number called offset.
- The **FIFO** is only guarantee inside a partition.
- When a topic is created, the number of partitions should be given
- The producer can choose which partition will get the message or let Kafka decides for him based on a hash of the message key (recommended). So the message key is important and will be the used to ensure the message order.
- Moreover, as the consumer will be assigned to one or several partition, the key will also “group” messages to a same consumer.



Record flow

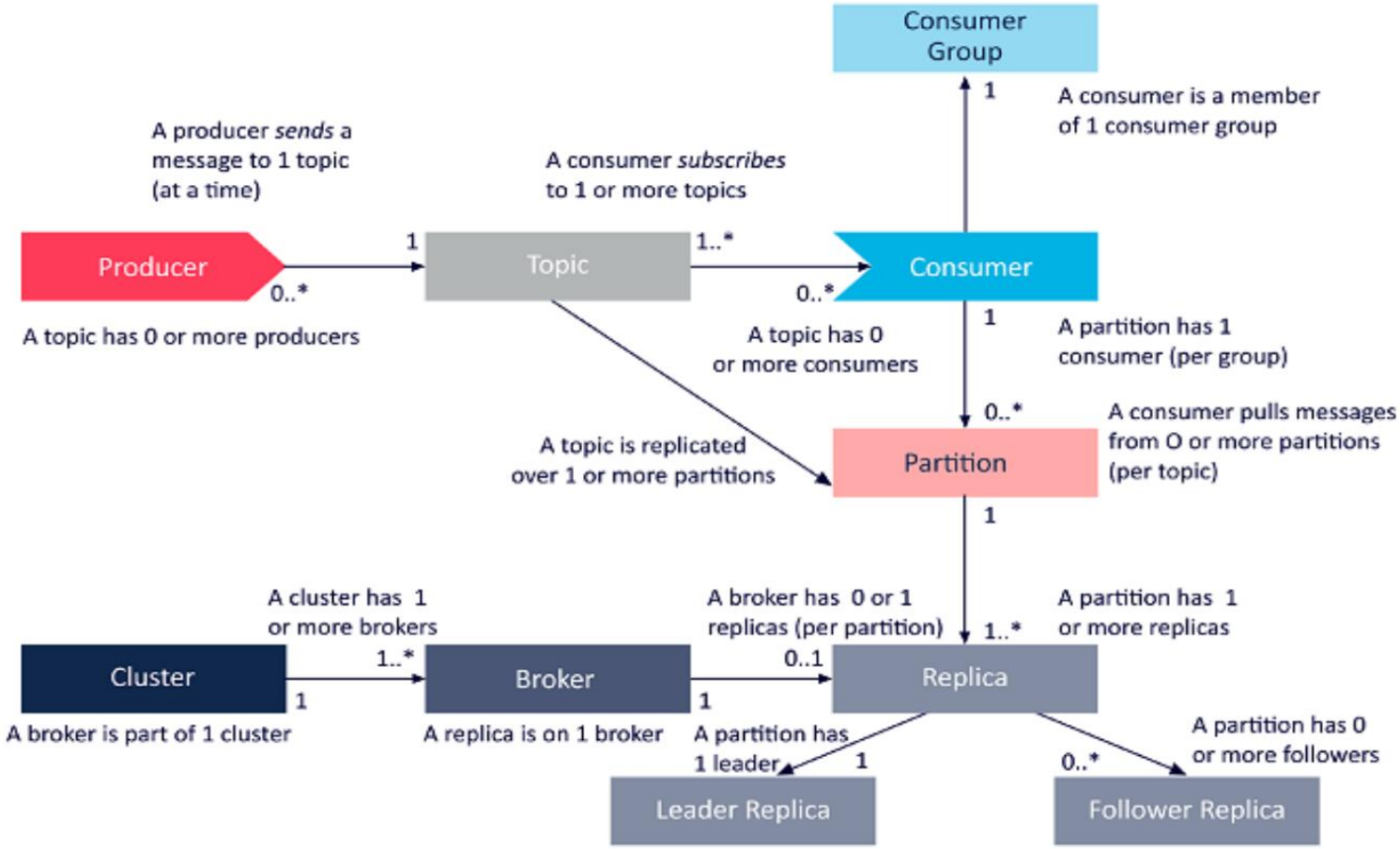
- We have a broker with three topics, where each topic has 8 partitions.
- The producer sends a record to partition 1 in topic 1 and since the partition is empty the record ends up at offset 0.



Next record is added to partition 1 will and up at offset 1, and the next record at offset 2 and so on.

This is a commit log, each record is appended to the log and there is no way to change the existing records in the log(immutable). This is also the same offset that the consumer uses to specify where to start reading.

Architecture



Partitions and Brokers

- Each broker holds a number of partitions and each of these partitions can be either a leader or a replica for a topic.
- All writes and reads to a topic go through the leader and the leader coordinates updating replicas with new data. If a leader fails, a replica takes over as the new leader.