Kenneth R N Wilson

Kafka

Introduction

THIS DOCUMENT COVERS

10,000 Feet View

WHAT IS KAFKA?

- High throughput distributed messaging system.

KEY FEATURES?

- High throughput distributed message system
- Horizontally scalable to 100s of brokers and millions of messages per second
- Latency of less that 10ms

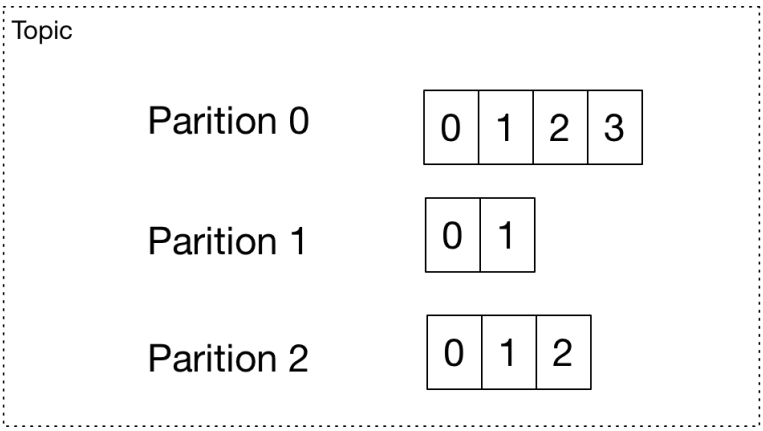
USES?

- Messaging
- Activity Tracking
- Stream Processing
- Decoupling Systems

Overview

Topic

A topic is a stream of data that is analogous to a table in a relational database. Topics are split into partitions

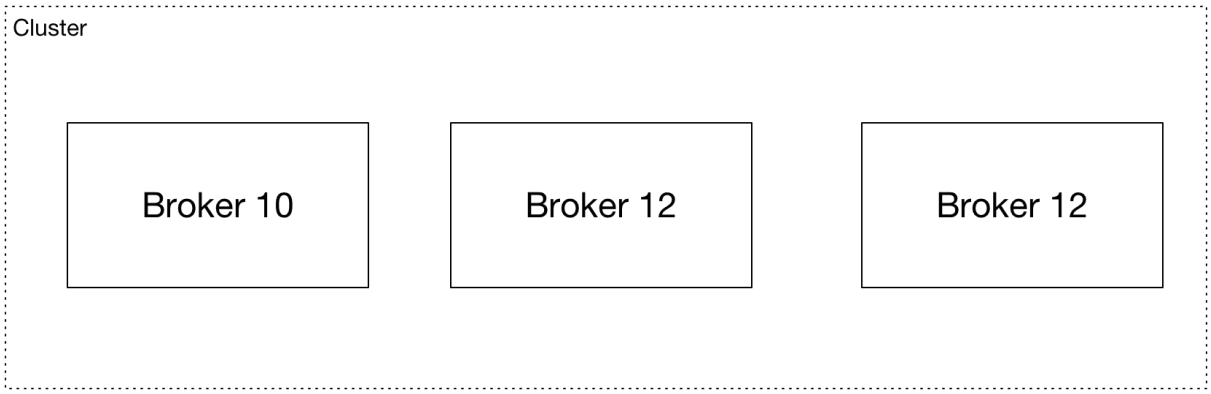


Within each partition messages are allocated unique incremental integer identifiers known as offsets. The message at offset 1 of partition 0 will be different from the messages at offset 1 of partition 1 and the message at partition 2. Ordering is only guaranteed within partitions and not across partitions.

- Topics are identified by topic name
- Each topic is split into one or more partitions.
- The order is only guaranteed within each partition and not across the topic as a whole.
- Data written to a partition is immutable. It can never be changed.

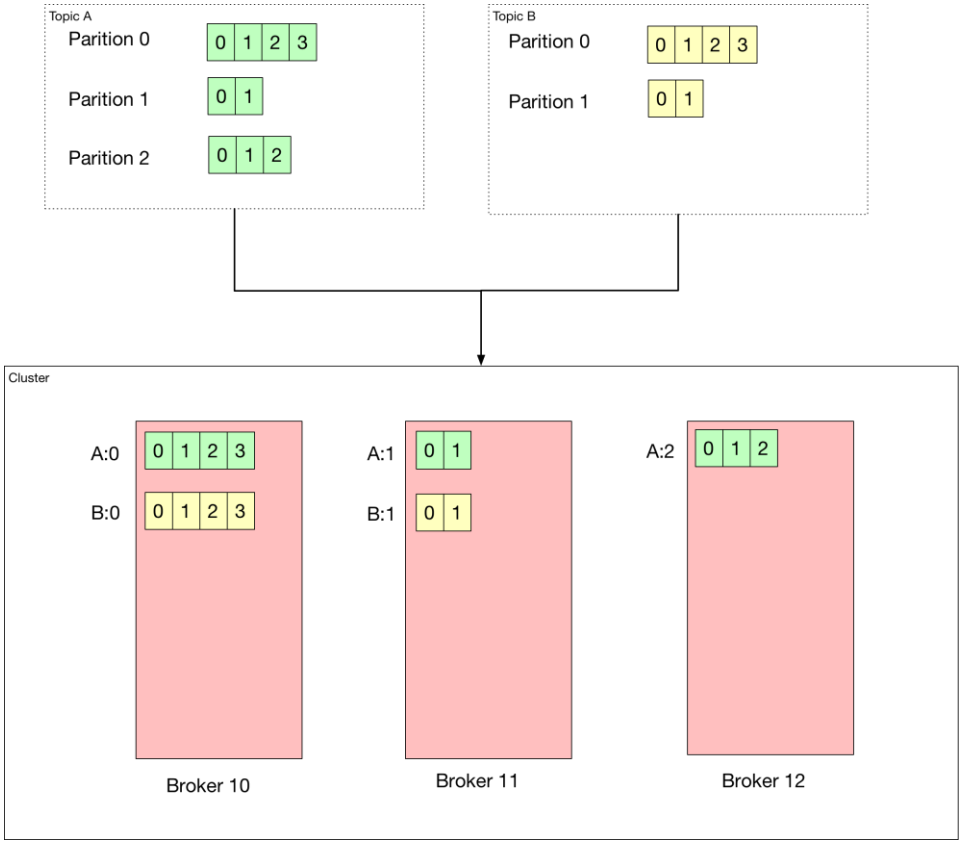
Brokers and Clusters

A broker is a physical server process. Brokers are grouped together into a cluster to provide scalability and reliability. Each broker has unique integer identifier



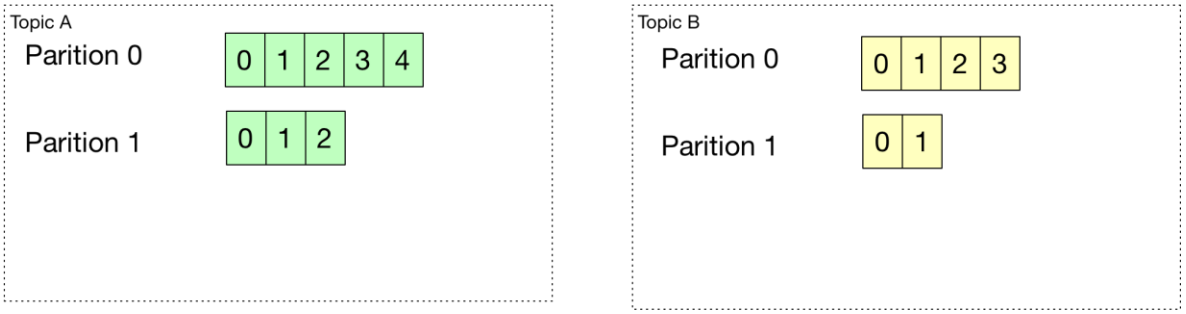
- A broker is identified by a unique integer ID.
- Each broker will contain a subset of the cluster's partitions.
- Connecting to any broker gives access to whole cluster
- The broker we connect to is called the bootstrap broker

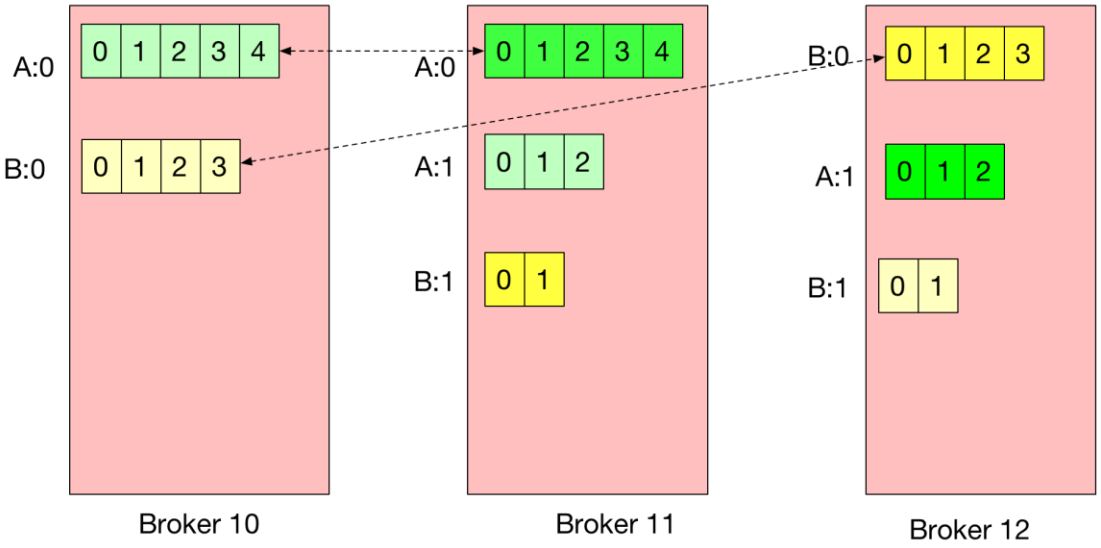
Imagine we have two topics: A and B, each with two partitions. The partitions are mapped onto the brokers within the cluster



Replication

When we create a topic, we specify a replications factor between 2 and 3. 3 is considered the safest replication factor. Imagine we have a replication factor of 2.

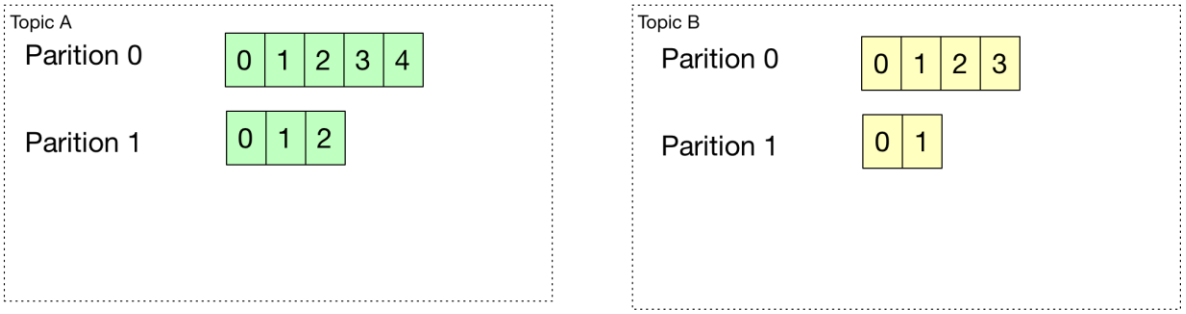


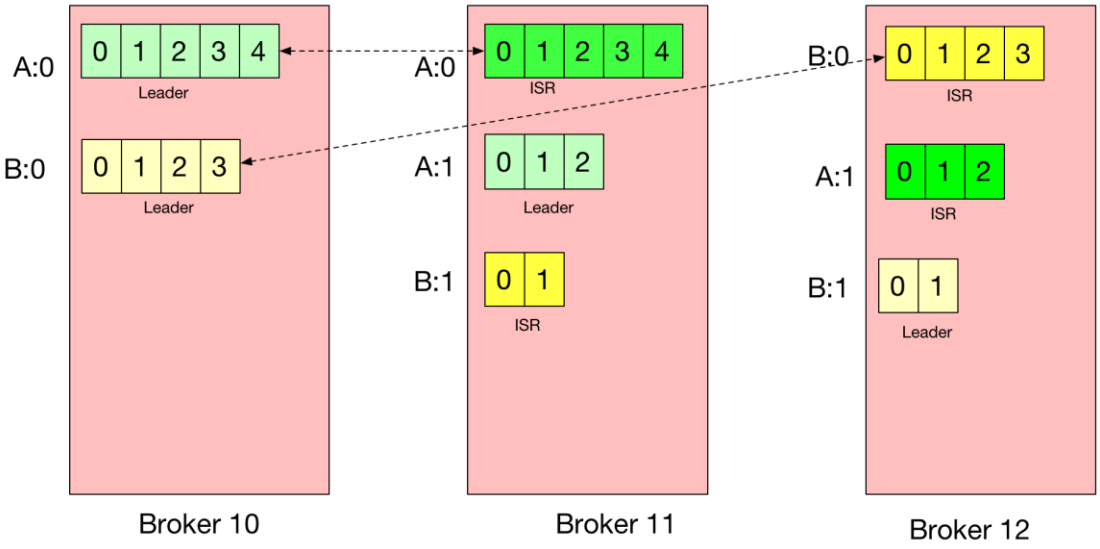


Replication

Leader and ISR

If we have a replication factor greater than one, then one broker is considered as the leader for the partition. A partition can have only one leader at any time. Only the leader provides and receives data for the partition with the other brokers carrying out synchronization. Any brokers other than the leader for a given partition are that partitions ISR (in-sync replica)





The leaders and ISR are determined by ZooKeeper

Producing

A producer writes to a topic. The client knows which partition to write to an automatically handles failover so that the application does not need to deal with it. There are different strategies to handle the acknowledgement of successful writes.

- Acks0: Producer does to care about ack and can have possible data loss.
- Acks1: Wait for leader to acknowledge – data loss is limited.
- Acks2: wait for leader and all replicates to acknowledge – no data loss

Keys

If a producer provides a key all messages with the same key go to the same partition. Otherwise, messages are sent round-robin to the partitions. Keys facilitate ordering for a specific piece of information.

Consuming

A consumer reads form a topic. The consumer client knows which broker to read from and handles failover. Within a partition data is read in order.

A consumer group enables horizontal scaling of reading. Each partition is read by only one consumer from a consumer group. Some consumers may be inactive where we have more consumers than partitions.