Apache Kafka

Agenda

- Introduction
- Core Concepts of Kafka
- Features of Kafka
- Kafka Streams
- Kafka Connect
- ksqlDB

INTRODUCTION

Apache Kafka is an open source publish-subscribe messaging system based on the concept of a distributed commit log.

Messages (aka records) in Kafka are distributed, stored durably and in order, and can be read deterministically.

INTRODUCTION

Apache Kafka is an open source "umbrella" project with the following modules:

- Kafka Core
  - Broker
  - Producer and Consumer APIs
- Kafka Streams
- Kafka Connect

From a messaging queue to a distributed streaming platform

Architecture of kafka

Apache Kafka is designed for distributed, fault-tolerant, and scalable handling of streaming data.

On high level overview, There is a cluster containing multiple brokers, where 1 is leader and rest are followers there by maintaining replica of data. Leader broker hosts a leader partition where the producer writes and consumer reads. Rest brokers hosts follower partitions and hence will replicate the same data to different disk location for fault tolerance.

Producer: applications that publish messages to Kafka topics.

public ProducerRecord(String topic, Integer partition, K key, V value, Iterable<Header> headers)

CORE CONCEPTS OF KAFKA (CORE)

- Records
- Topics
- Partitions
- Replicas and In-Sync Replicas
- Offsets
- Brokers
- Producers
- Consumers
- Retention

Records

- Record (aka message or event) is the unit of data in Kafka
  - Array of bytes (in no particular format)
- Apache Avro as data serialization framework
- Record has a key and a value
  - Both could be null
- Records are categorized into topics

TOPICS

- Records are categorized into topics
  - Think a table or a directory
- Producers publish messages to topics while consumers consume them
- Topics are partitioned
  - Namespaces of one or many partitions
- kafka-topics shell script manages Kafka topics

PARTITIONS

- Topics are partitioned into one or more partitions
- Partitions hold zero, one or many records
- Ordered (by offsets) immutable sequence of records
- A partition is a single ordered log
- Stored durably on disk
- Records are added to partitions in append-only fashion
- Partitions are replicated among brokers as replicas
- In-sync replicas (ISRs)

REPLICAS AND IN-SYNC REPLICAS

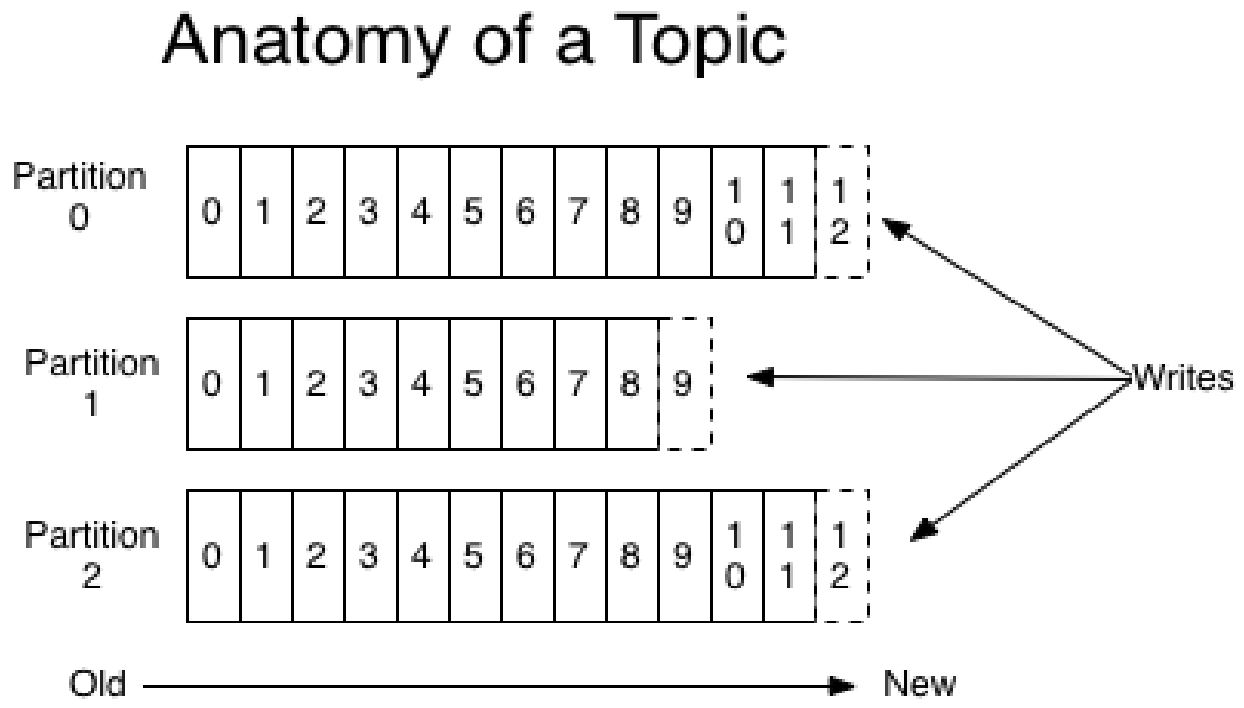
- Replica is a copy of a partition
- Replication factor is the number of replicas of a topic
- There can be one or many replicas
- Allows for automatic failover when a broker fails
- One replica is the leader while others are followers
- Leader handles writes from producers, and the followers merely copy the leader's log
- In-Sync Replica is a replica that has enough records to be considered in partition leader election
- Use kafka-topics --describe to list the details of a topic (incl. replicas and in-sync replicas)

OFFSETS

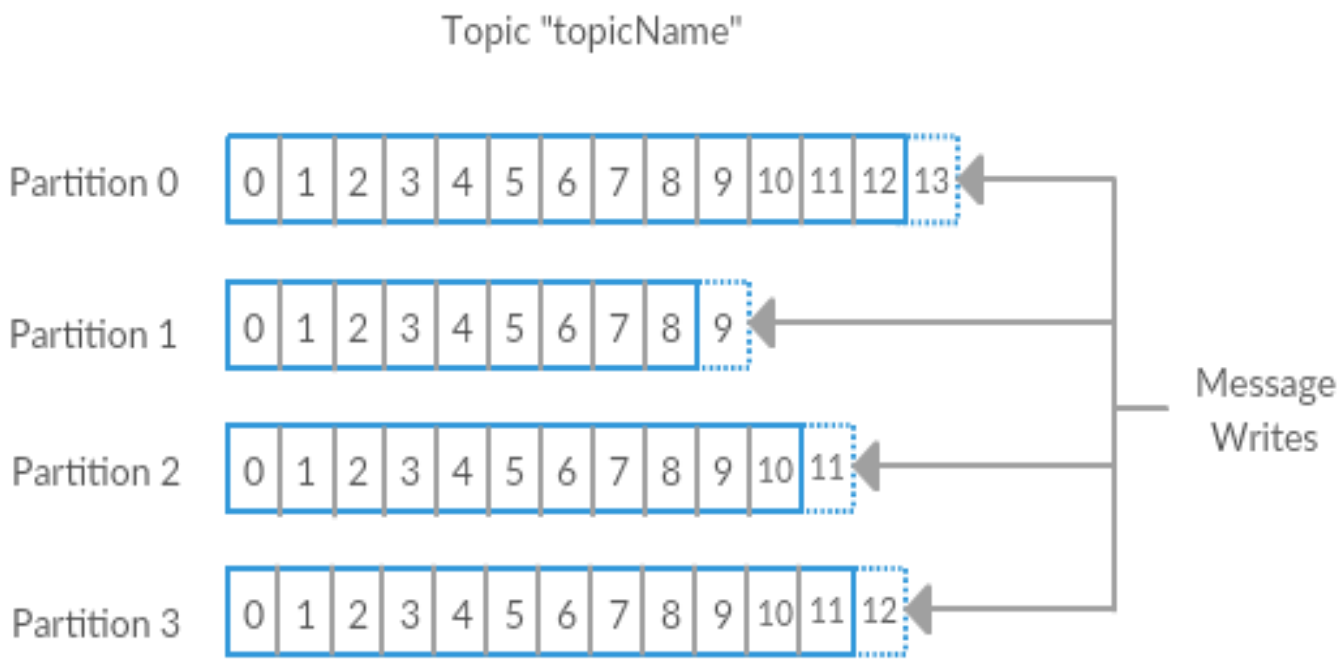
- Offset is a unique sequential numerical position of a record (in a partition of a topic)
    - A message in a partition has a unique offset
- Offsets start from 0
- Offsets are unique per partition only
    - Not across partitions

KAFKA TOPICS AND PARTITIONS

(DISTRIBUTED COMMIT LOG)



KAFKA TOPICS AND PARTITIONS (CNTD)



BROKERS

- Kafka Broker is a Kafka server that manages records
- Receives messages, assigns offsets, and commits messages to storage on disk
- Kafka Cluster consists of one or more brokers
- Uses Zookeeper as the source of truth

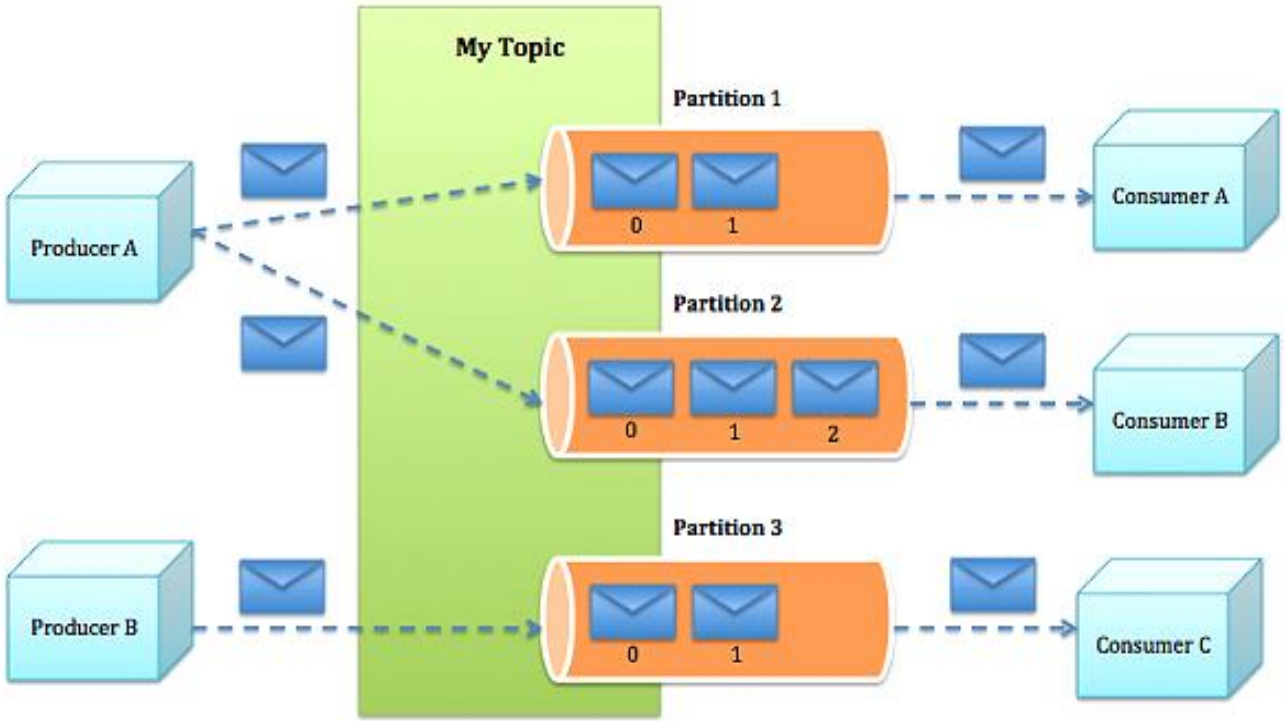
PRODUCERS

- Kafka clients that publish records to a Kafka cluster
- Send messages to topics
- Can optionally specify partitions
- KafkaProducer API for Java

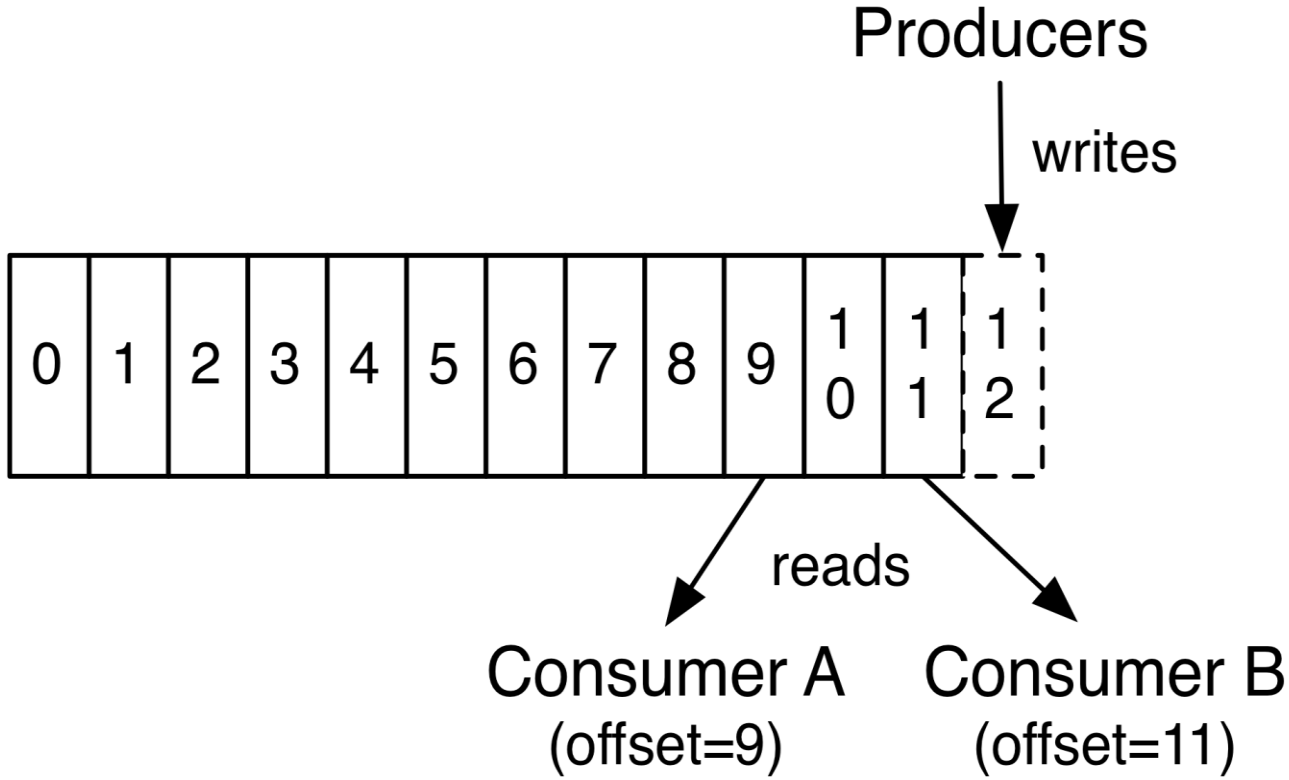
CONSUMERS

- Kafka clients that consumes records from a Kafka cluster
- Subscribe to receive messages from topics
- Read messages in the order they were produced
  - Per partition only
- KafkaConsumer API for Java

KAFKA PRODUCERS AND CONSUMERS



KAFKA PRODUCERS AND CONSUMERS



RETENTION

- Retention of messages in topics is how long messages are stored in topics
  - Durable message retention
  - For some period of time, e.g. 7 days
  - Until a topic reaches a certain size in bytes, e.g. 1 gigabyte
- Once these limits are reached, messages are expired and deleted
- Can be selected on a per-topic basis

FEATURES OF KAFKA

- Thousands of Producers
- Thousand of Consumers
- Client Independence
- High Throughput
- Message Persistence
- Disk-based Retention
- Scalability
- High Performance

KAFKA STREAMS

- Kafka Streams is a client library for stream processing on Kafka
- Stream and Table Abstractions (over Kafka topics)
- Elastic, highly scalable, fault-tolerant
- Home Page

KAFKA CONNECT

- Kafka Connect is a framework for a scalable and reliable data streaming between Apache Kafka and other systems
- A framework for Kafka connectors
- Distributed and standalone (single process) modes
- REST interface for connector deployment and management
- Home Page

KSQLDB

- ksqlDB is an open source streaming SQL engine for stream processing on Kafka
- Interactive SQL interface
- "KSQL: Query Your Streams Without Writing Code"
- No need to write code in a programming language like Java or Python
- SQL layer atop Kafka Streams
- Executing SQL on tables and streams
- Home Page

Thank you

Mohammad Rafiq

mohammadr5@hexaware.com

Project Resources:

https://github.com/morfiq/ApacheKafka/