Chapter 5: Apache Kafka

→ What is Kafka?

- A distributed streaming platform
  - ↳ set of machines working together to be able to handle and process real-time infinite data
- Developed first at LinKedin
- Based on publish-subscribe model
  - ↳ producers send messages
  - ↳ consumers receive messages

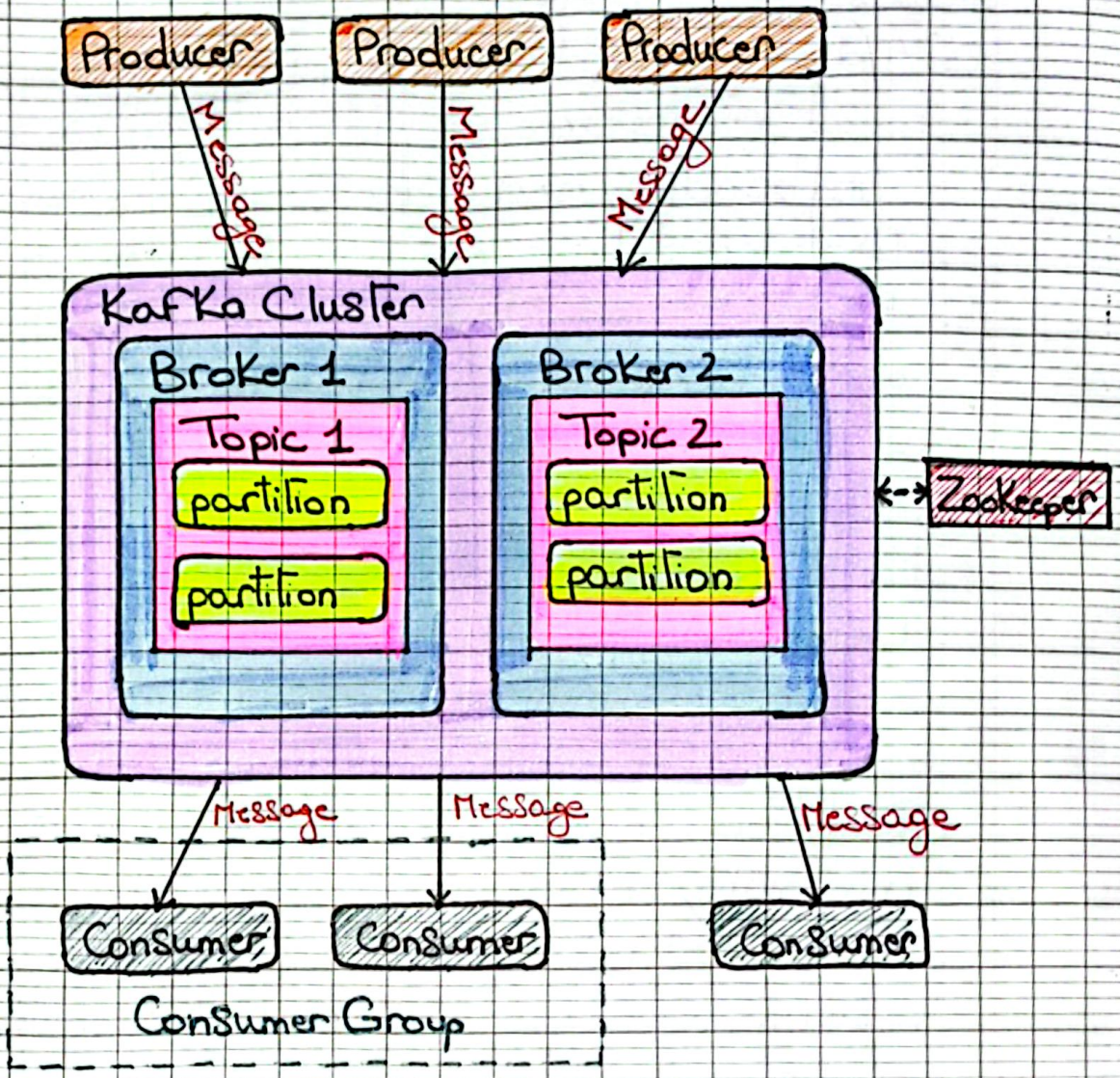
→ Kafka Characteristics

- **High Throughput**
  - ↳ can process millions of messages per second
  - ↳ works well on commodity hardware (low-cost machine)
- **Real-time**: message are available to consumers immediatly after being produced
- **Persistent Messaging**
  - ↳ stores messages on disk with constant-time performance (O(1)), Keep performance high with large datasets
- **Fault Tolerant**
  - ↳ message are replicated across mulliple servers (cluster) to avoid data loss
- **Scalability**: scales horizontally (add more machines) without downtime
- **Multi-Client**: Support Java, Python, .Net, Php, Ruby...

→ Why use Kafka

- Support both real-time and batch (offline) processing
- Integrates with Apache Spark/Storm and Hadoop
- Decouple data pipelines (independent producers/consumer

→ Kafka Architecture



1) Kafka Cluster:
   - Distributed system composed of multiple brokers
   - Each broker has an integer id
   - Producer / consumer can connect to any broker to access the full cluster

2) Brokers
   - Servers (physical or virtual) that form the Kafka cluster
   - Stores partitions of topics

- Each broker is responsible for receiving, storing, and serving data
- Handle read/write operations from consumers and producers
- Manage replication of data

3) Topics and Partitions
- Data in Kafka is organized as topics
- Each topic is divided into partitions (basic unit of parallelism in Kafka)
- Topic represent a category / Feed name to which records are published
- Once data is published, it can't be changed or updated

4) Producers
- Client application that publish (write) data to Kafka
- Send records to the appropriate topic and partition based on partitioning strategy

5) Consumers
- Client application that subscribe (read) data from Kafka and process it
- Maintains TCP connections to brockers to fetch data

c. Consumer Group
- Set of consumers sharing a common group ID
- Each consumer within the group reads from a unique partition
- Together, the group consumes the full topic

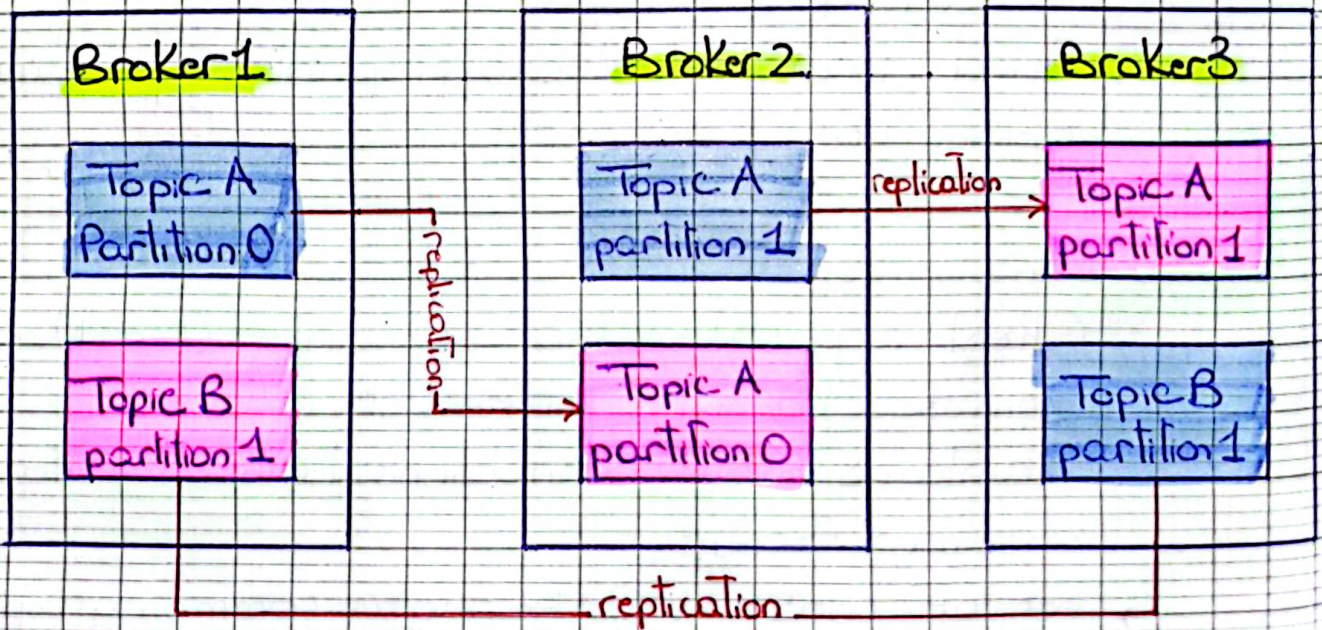
6) ~~ZooKeeper~~

- Manages brokers and cluster membership
- Maintains topic configurations (e.g. nb. of partitions, leaders)
- Store info about cluster status and consumer offsets

→ Partition Replication

- Partitions are replicated across brokers for fault tolerance
- A broker can hold partitions from same/different topics
- A partition can be:
  - ↳ Leader: primary replica that handles all reads/writes for a partition and syncs with replicas
  - ↳ Follower: Secondary replica that syncs from the leader

If a leader fails, a Follower becomes the new leader



Leader replica