ADALTAS

A distributed streaming platform

What is data streaming?

ADALTAS

It is the practice of:

- **capturing data** in real-time
- **storing** these event streams durably for later retrieval
- **manipulating, processing, and reacting** in real-time and retrospectively
- **routing** the event streams to different destination technologies as needed

Data streams examples

ADALTAS

- Financial transactions (stock exchanges, banks, insurances)
- Logistics and the automotive industry (track and monitor cars, trucks, fleets, and shipments)
- Website events (order, click on ads)
- Health care (monitor patients in hospital)
- IoT devices (capture and analyze sensor data)
- Sport tracking (game score boards)
- User interactions (chats, likes, reacts, ...)

Apache Kafka

ADALTAS

- distributed streaming platform
- open source and free
- Created by LinkedIn in 2011

Real-world examples with Kafka

ADALTAS

- **Netflix** uses Kafka to apply recommendations in real-time while you're watching TV shows.
- **Uber** uses Kafka to gather user, taxi, and trip data in real-time to compute and forecast demand, and compute surge pricing in real-time.
- **Linkedin** uses Kafka to prevent spam, collect user interactions to make better connections recommendations in real-time.

Kafka functionalities

ADALTAS

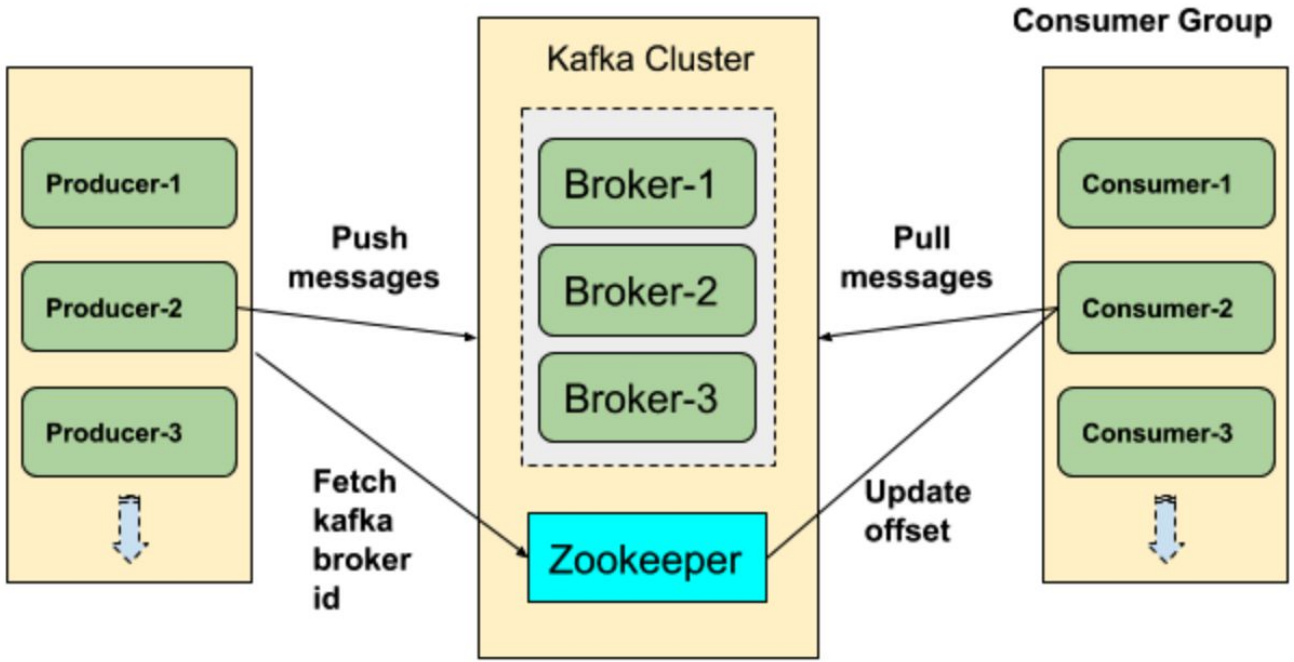
- **Publish and subscribe to streams of records**, similar to a message queue or enterprise messaging system
- **Store streams of records** in a fault-tolerant durable way
- **Process streams of records** as they occur (new ones)

Read more on [Kafka web site]

Kafka features

- **distributed**
- **horizontally-scalable** (because of built-in partitioning)
- **fault-tolerant** (because of replications)
- **low latency**

Kafka architecture



Messaging system

ADALTAS

Records are published to **topics**.

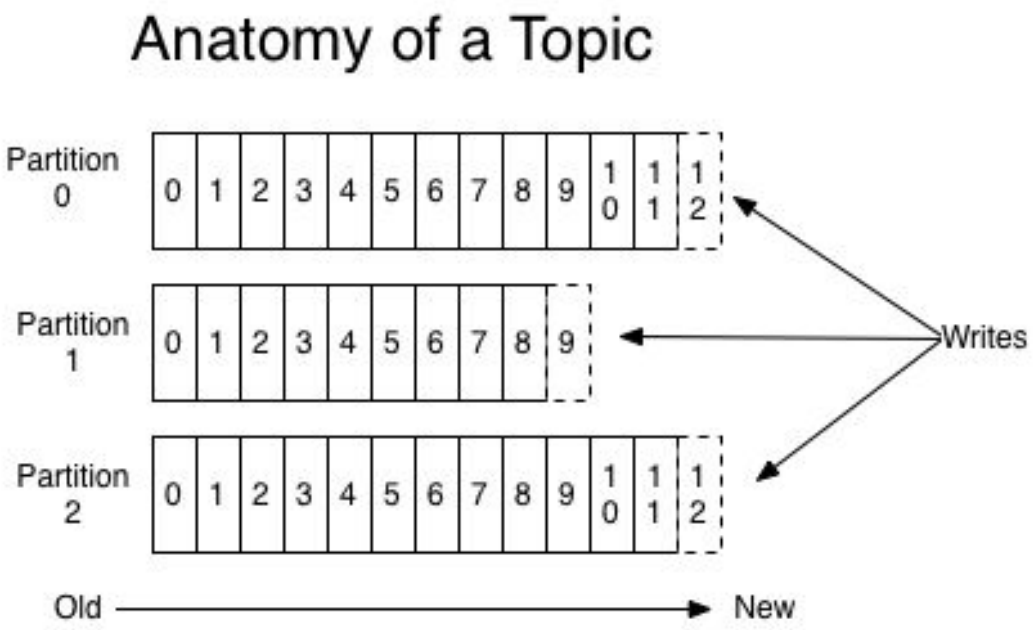
- 1 record = (key, value, timestamp)
- 1 record belongs to 1 topic

Topics

ADALTAS

- 1 topic is split into **1 to N partitions**
- Each topic is **replicated** 1 to M times
- Records **order is guaranteed** within a partition
- Records are persisted given a **retention period**

Topics


Anatomy of a Topic
Partition 0
0 1 2 3 4 5 6 7 8 9 10 11 12
Partition 1
0 1 2 3 4 5 6 7 8 9
Partition 2
0 1 2 3 4 5 6 7 8 9 10 11 12
Writes
Old
New

Producers

ADALTAS

Kafka is **dumb** (no routing policy) = **producers** are responsible for choosing **which topic and which partition** to write to

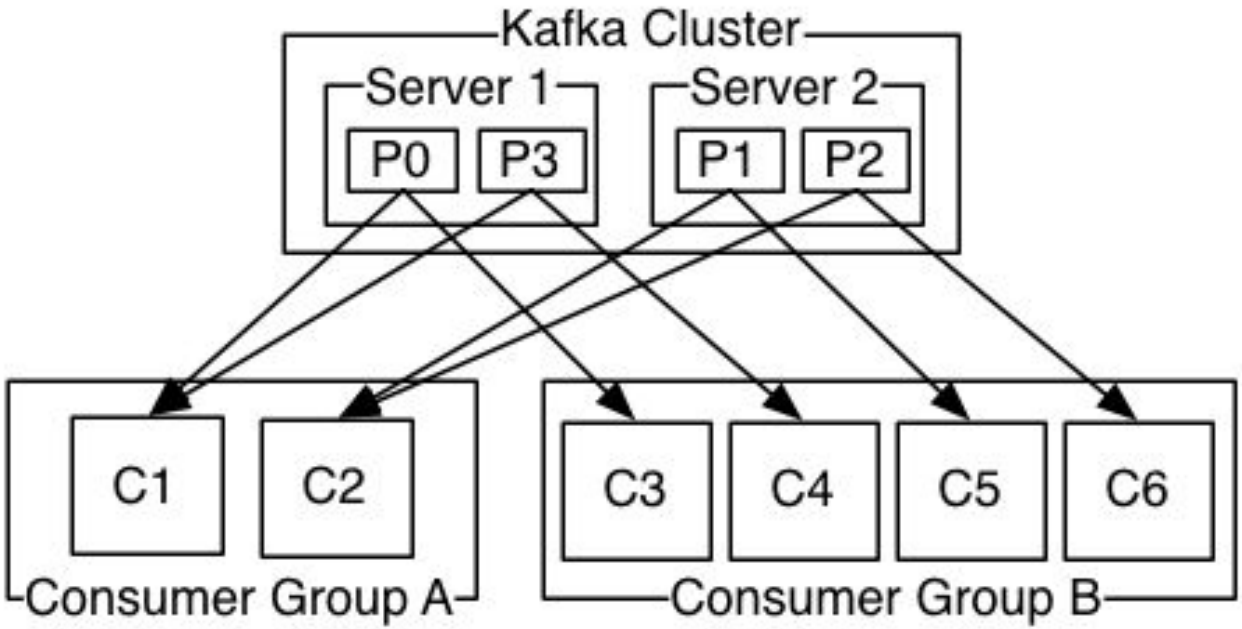
Methods to choose: round-robin, based on key, etc.

Consumers

ADALTAS

- **Choose the offset** to start reading
- Each record delivered to **1 consumer of each consumer group** (1 partition to 1 consumer)
- **Fair distribution** of records between consumers of a group → scalability + fault tolerance

Consumers


Kafka Cluster
Server 1
Server 2
P0
P3
P1
P2
C1
C2
C3
C4
C5
C6
Consumer Group A
Consumer Group B

Data distribution

ADALTAS

Server = Kafka **broker**

For each partition:

- 1 “leader” → read + write requests
- 0 to N-1 “followers” → replication

Performance

ADALTAS

- The performance is **not impacted by the volume** stored
- Allow **scale of processing** → increase consumer instances
- Keep **records order** → 1 consumer receives records from 1 partition
- **Multiple independent "customers"** → 1 offset per consumer

Other use cases

- **Storage system** (especially for logs):
  - Data written to disk + replicated (CP)
  - No performance impact from ↗️ volume
- **Stream processing:** New API **Kafka Streams**