Kafka: Introduction

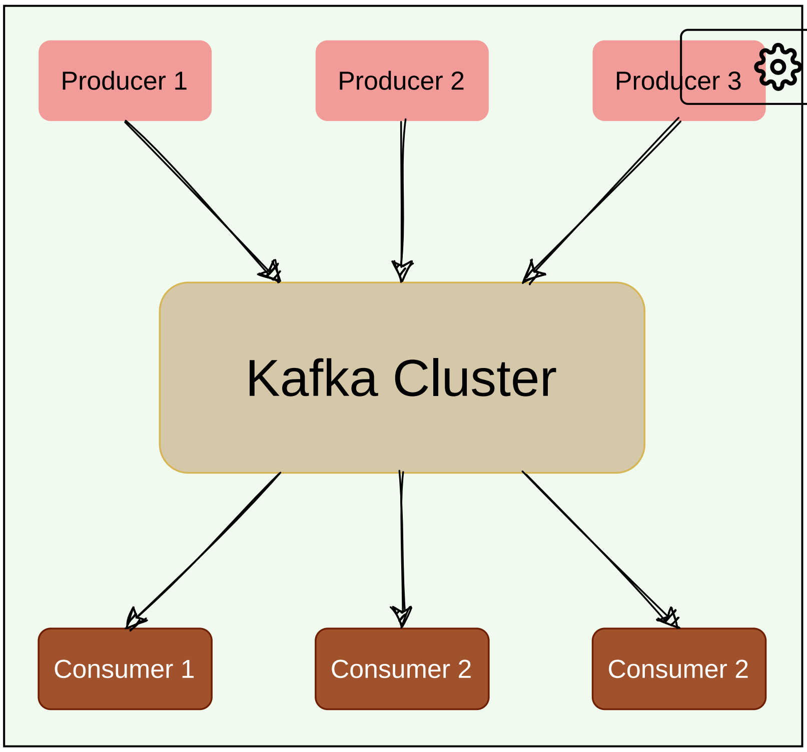
This lesson presents a brief introduction and common use cases of Kafka.

We'll cover the following

- What is Kafka?
- Background
- Kafka use cases

What is Kafka?#

Apache Kafka is an open-source **publish-subscribe**-based messaging system (*Kafka can work as a message queue too, more on this later*). It is **distributed**, **durable**, **fault-tolerant**, and **highly scalable** by design. Fundamentally, it is a system that takes streams of messages from applications known as producers, stores them reliably on a central cluster (containing a set of brokers), and allows those messages to be received by applications (known as consumers) that process the messages.



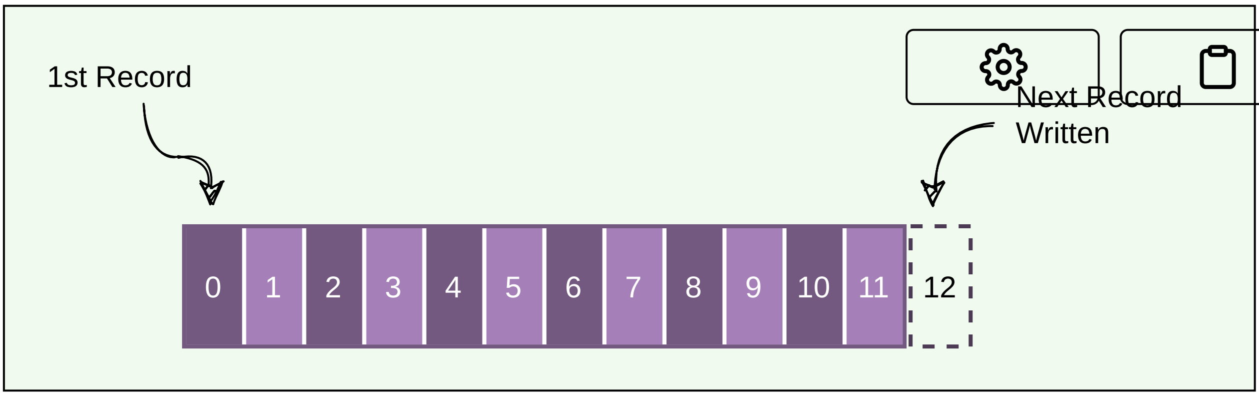
A high-level view of Kafka

Background#

Kafka was created at LinkedIn around 2010 to track various events, such as page views, messages from the messaging system, and logs from various services. Later, it was made open-source and developed into a comprehensive system which is used for:

1. Reliably storing a huge amount of data.
2. Enabling high throughput of message transfer between different entities.
3. Streaming real-time data.

At a high level, we can call Kafka a distributed **Commit Log**. A Commit Log (*also known as a Write-Ahead log or a Transactions log*) is an **append-only** data structure that can persistently store a sequence of records. Records are always appended to the end of the log, and once added, records cannot be deleted or modified. Reading from a commit log always happens from left to right (*or old to new*).



Kafka as a write-ahead log

Kafka stores all of its messages on disk. Since all reads and writes happen in sequence, Kafka takes advantage of sequential disk reads (*more on this later*).

Kafka use cases#

Kafka can be used for collecting big data and real-time analysis. Here are some of its top use cases:

1. **Metrics**: Kafka can be used to collect and aggregate monitoring data. Distributed services can push different operational metrics to Kafka servers. These metrics can then be pulled from Kafka to produce aggregated statistics.
2. **Log Aggregation**: Kafka can be used to collect logs from multiple sources and make them available in a standard format to multiple consumers.
3. **Stream processing**: Kafka is quite useful for use cases where the collected data undergoes processing at multiple stages. For example, the raw data consumed from a topic is transformed, enriched, or aggregated and pushed to a new topic for further consumption. This way of data processing is known as stream processing.
4. **Commit Log**: Kafka can be used as an external commit log for any distributed system. Distributed services can log their transactions to Kafka to keep track of what is happening. This transaction data can

be used for replication between nodes and also becomes very useful for disaster recovery, for example, to help failed nodes to recover their states.

5. **Website activity tracking**: One of Kafka's original use cases was to build a user activity tracking pipeline. User activities like page clicks, searches, etc., are published to Kafka into separate topics. These topics are available for subscription for a range of use cases, including real-time processing, real-time monitoring, or loading into Hadoop (https://hadoop.apache.org/) or data warehousing systems for offline processing and reporting.
6. **Product suggestions**: Imagine an online shopping site like amazon.com (http://amazon.com), which offers a feature of 'similar products' to suggest lookalike products that a customer could be interested in buying. To make this work, we can track every consumer action, like search queries, product clicks, time spent on any product, etc., and record these activities in Kafka. Then, a consumer application can read these messages to find correlated products that can be shown to the customer in real-time. Alternatively, since all data is persistent in Kafka, a batch job can run overnight on the 'similar product' information gathered by the system, generating an email for the customer with product suggestions.

← Back
Messaging Systems: Introduction

Next →
High-level Architecture

☑ Mark as Completed

Report an Issue