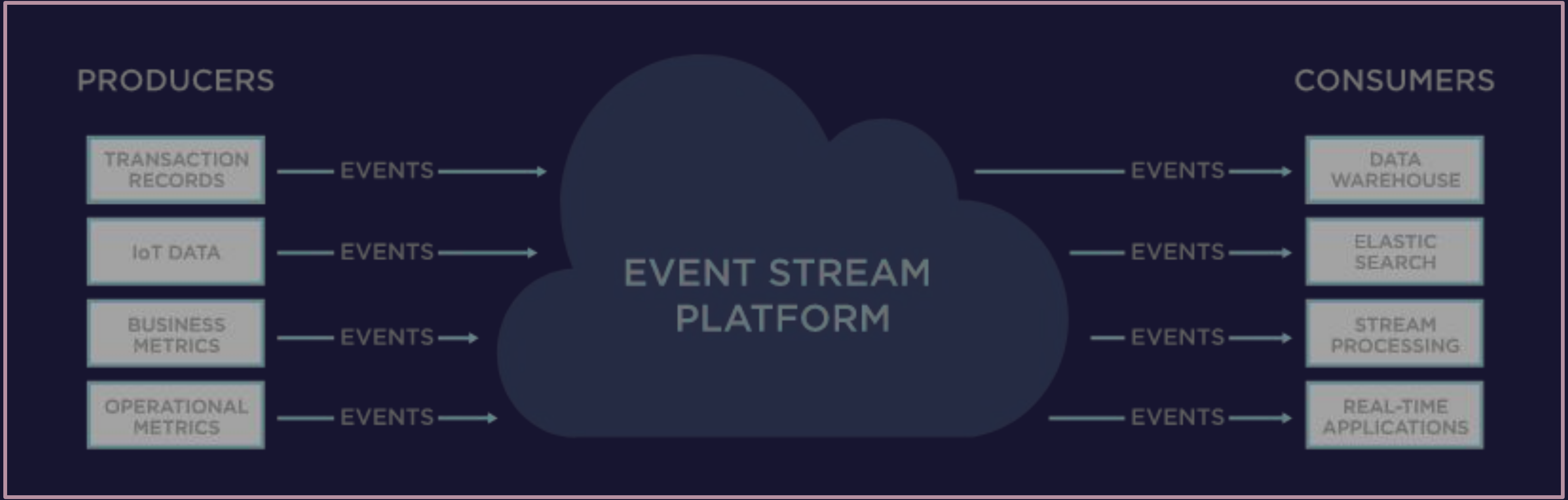
What the Kafka?

●●●

eidos.ai

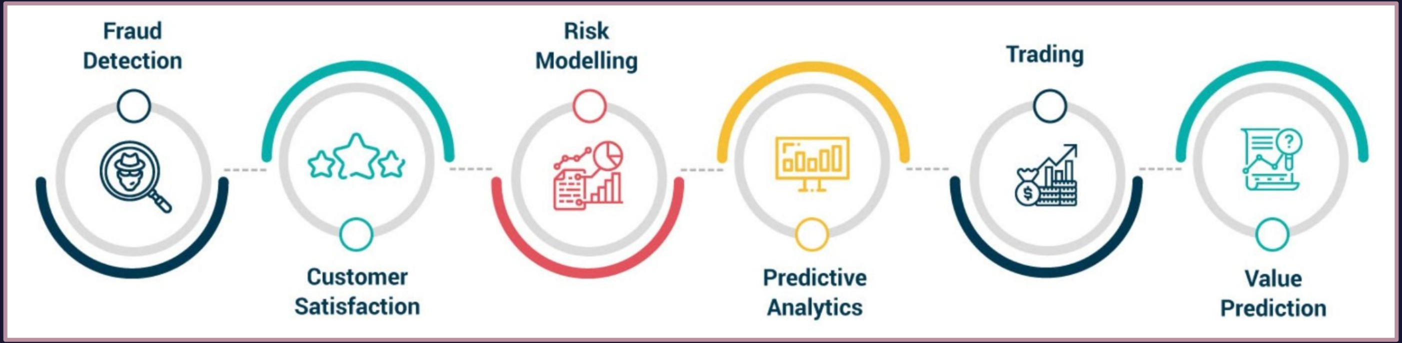
What is event streaming?



Use cases

Event streaming is applied to a wide variety of use cases across a large number of industries and organizations.

- As a messaging system.
- Activity tracking.
- To gather metrics data.
- For stream processing.
- To decouple a system.
- To integrate with other big data technologies such as Hadoop.



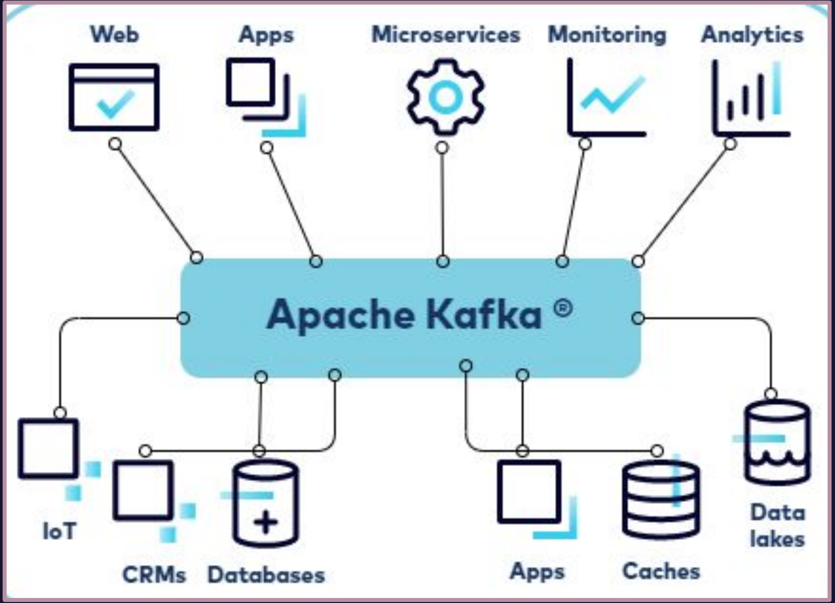
Kafka as event stream platform

Is an opensource distributed event streaming platform.

Is optimized for ingesting and processing streaming data in real-time.

Kafka provides three main functions to its users:

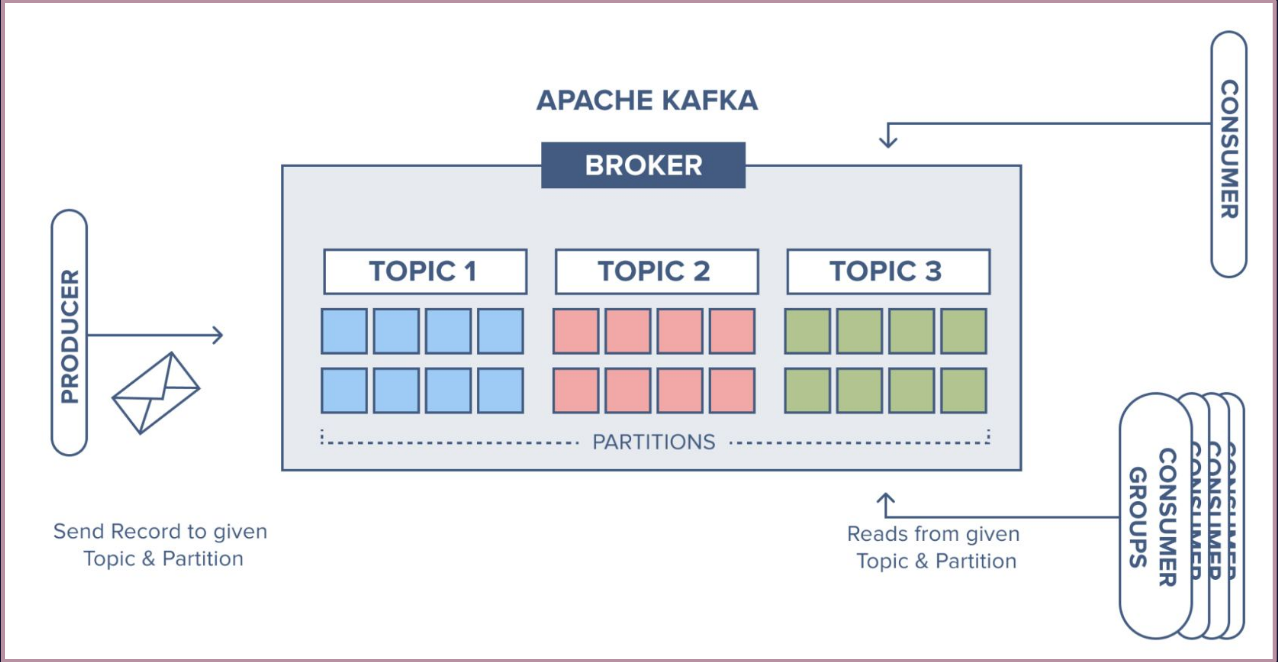
- To publish (write) and subscribe to (read) streams of events, including continuous import/export of your data from other systems.
- To store streams of events durably and reliably for as long as you want.
- To process streams of events as they occur or retrospectively.



Popularity

Who Uses Kafka?
airbnb
NETFLIX
Goldman Sachs
LinkedIn
Microsoft
The New York Times
intuit
TARGET

Kafka architecture



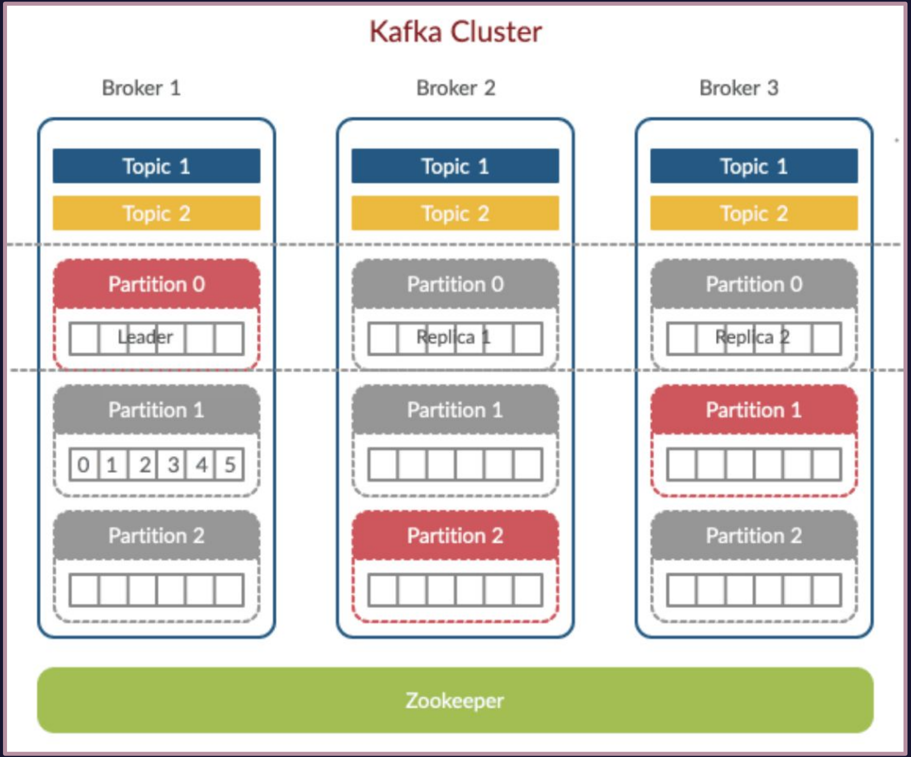
eidos.ai

Cluster, brokers & topics

A **broker** refers to a server in the Kafka storage layer that stores event streams from one or more sources.

A **Kafka cluster** is typically comprised of several brokers.

The Kafka cluster organizes and durably stores streams of events in categories called **topics**, which are Kafka's most fundamental unit of organization.
A topic is a log of events, similar to a folder in a filesystem, where events are the files in that folder.

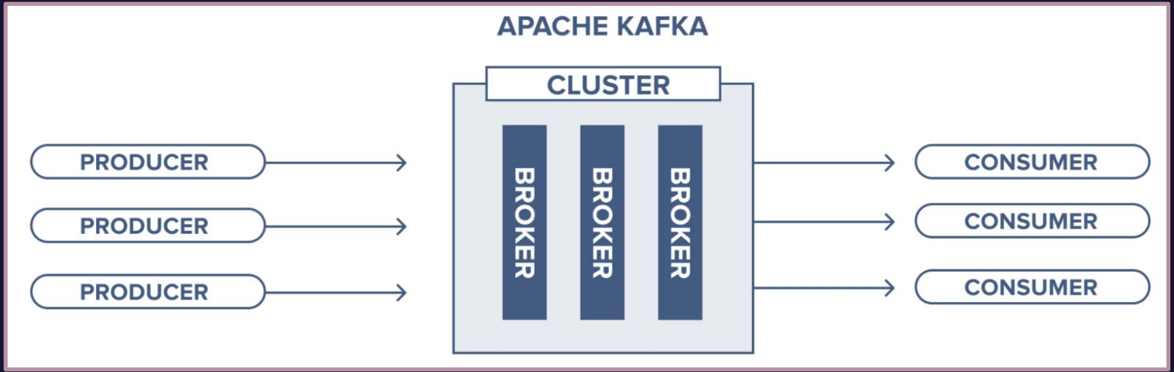


eidos.ai

Producers & Consumers

Producers are clients that write events to Kafka. The producer specifies the topics they will write to and the producer controls how events are assigned to partitions within a topic.

Consumers are clients that read events from Kafka.



Message persistance, scalability and fault tolerance

Message persistence refers to the capability of Kafka to durably store messages that are published to topics. When a producer publishes a message to a Kafka topic, the message is written to disk on the Kafka brokers. This ensures that the message is not lost in case of failures or system restarts.

Scalability in Kafka refers to its ability to handle large-scale data streams and increasing workloads. Kafka achieves scalability through a distributed architecture. It allows you to add more Kafka brokers to the cluster to handle higher data volumes and increased throughput.

Fault tolerance is a critical aspect of Kafka's design. It ensures that the system continues to operate reliably even in the face of failures. Kafka achieves fault tolerance through data replication

eidos.ai

Kafka APIs

- The Admin API
- The Producer API
- The Consumer API
- The Kafka Streams API
- The Kafka Connect API

eidos.ai