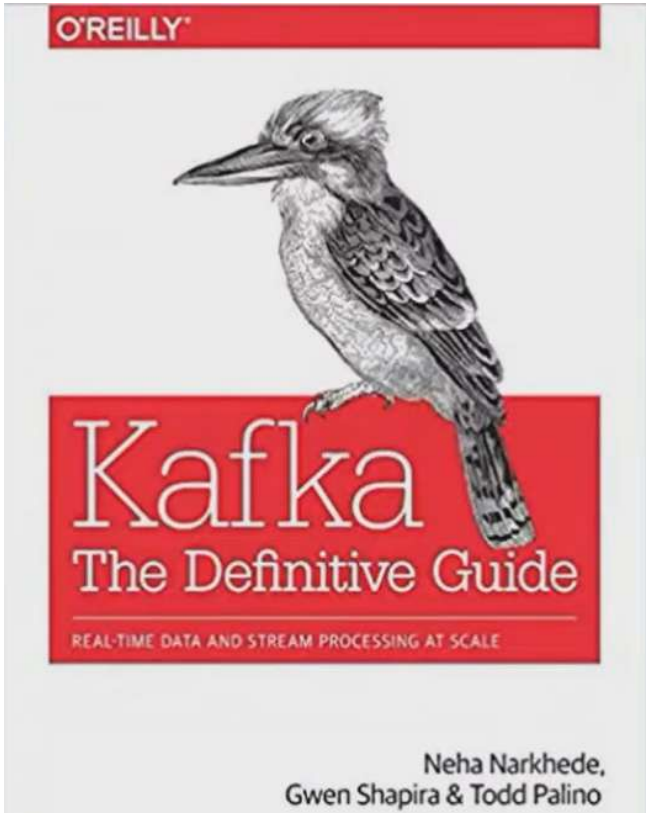
Módulo Profesional: Big Data Aplicado

Kafka

ÍNDICE



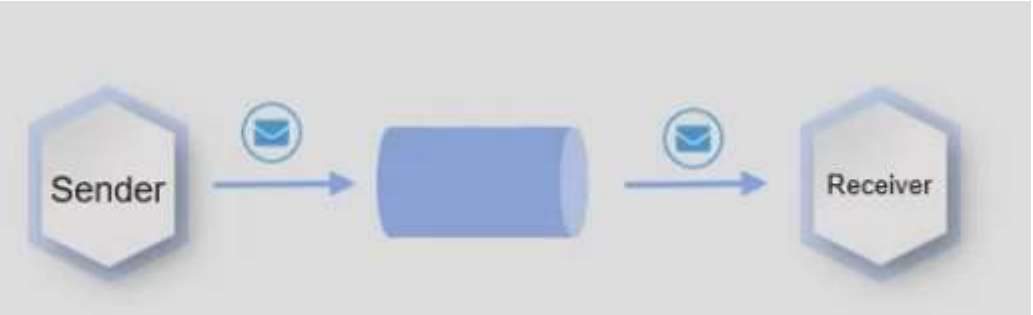
https://www.confluent.io/resources/ebook/kafka-the-definitive-guide/

ÍNDICE

Introducción

Sistema de mensajería punto a punto

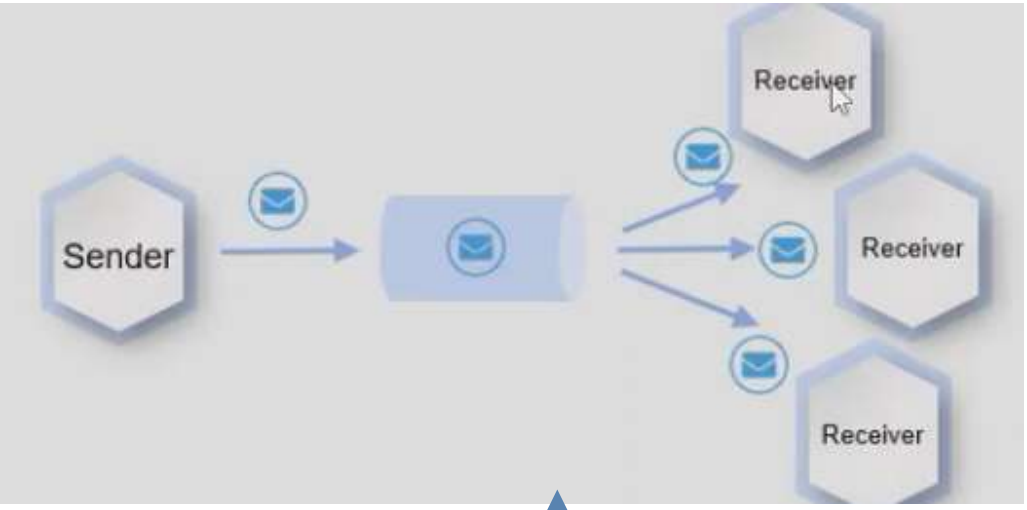
En un sistema punto a punto, los mensajes se almacenan en una cola. Uno o más consumidores pueden consumir los mensajes en la cola, pero un mensaje en particular solo puede ser consumido por un máximo de un consumidor.



Introducción

Sistema de mensajería Publish-Subscribe

En el sistema de publicación-suscripción, los mensajes se almacenan en un **topic**. A diferencia del sistema punto a punto, los consumidores pueden suscribirse a uno o más topics y consumir todos los mensajes de ese topic. En el sistema de publicación-suscripción, los productores de mensajes se llaman **publicadores** y los consumidores de mensajes se llaman **suscriptores**.



Kafka es de este tipo

Apache Kafka

Kafka como un sistema de mensajería Publicación/Suscripción

Kafka como un sistema de almacenamiento

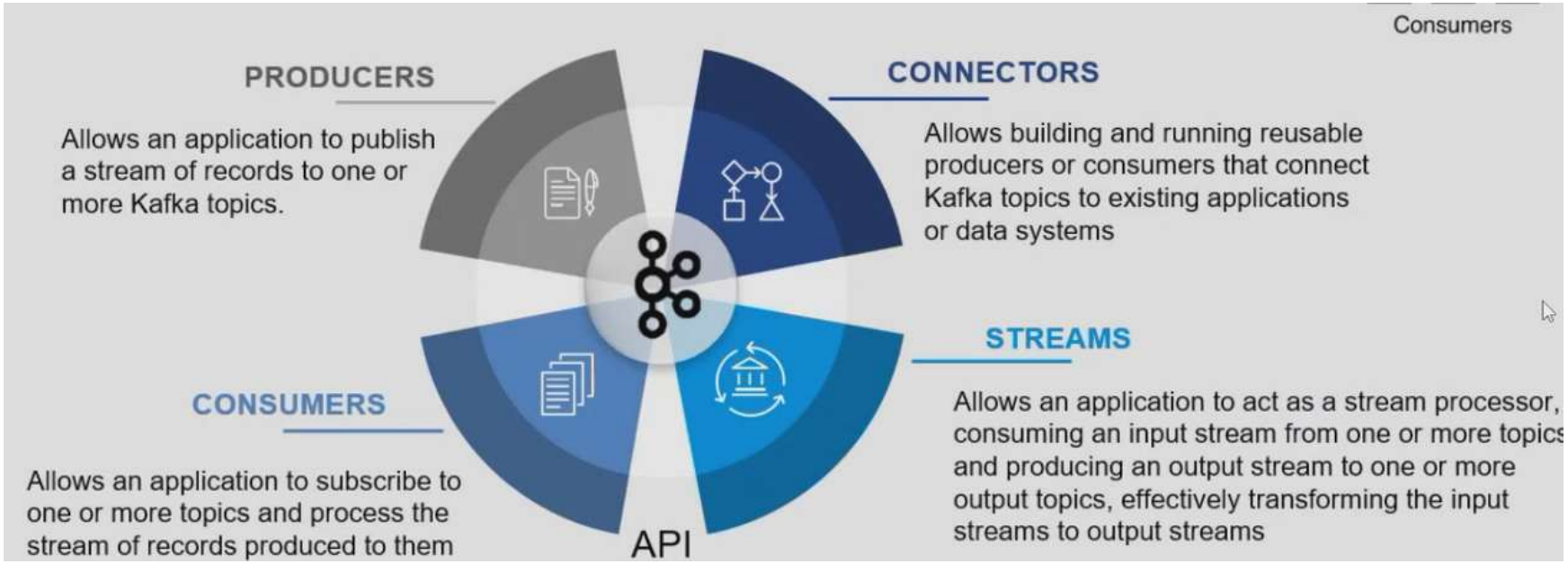
Sistema de archivos distribuido para el registro de confirmaciones (commit log)

Características

- Rápido, Baja latencia
- Escalable
- Duradero
- Tolerante a fallos
- Confiable
- Replicación

Apache Kafka

Apache Kafka es una cola de mensajes pub/sub en tiempo real, tolerante a fallos y altamente escalable, diseñada como un registro de transacciones distribuido.

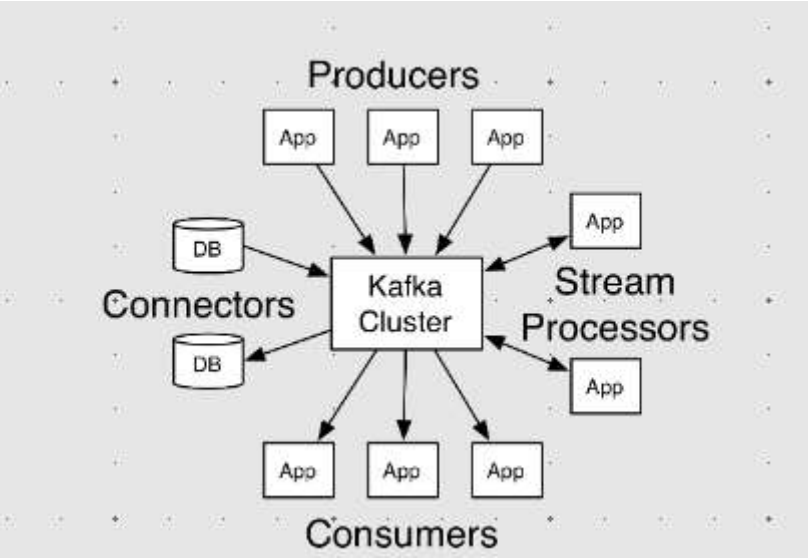


Kafka. Connectors

Los conectores de Kafka son componentes que permiten la ingesta y exportación de datos entre Kafka y otros sistemas. Existen dos tipos principales de conectores:

Source Connectors (Conectores de origen): se utilizan para extraer datos de fuentes externas y publicarlos en topics de Kafka. *Ejemplo*: Un conector que lee datos de una base de datos MySQL y los envía a Kafka.

Sink Connectors (Conectores de destino): se utilizan para leer datos de Kafka y escribirlos en un sistema externo. Ejemplo: Un conector que toma datos de un topic de Kafka y los almacena en Amazon S3.



Kafka. Connectors

Conectores más conocidos:

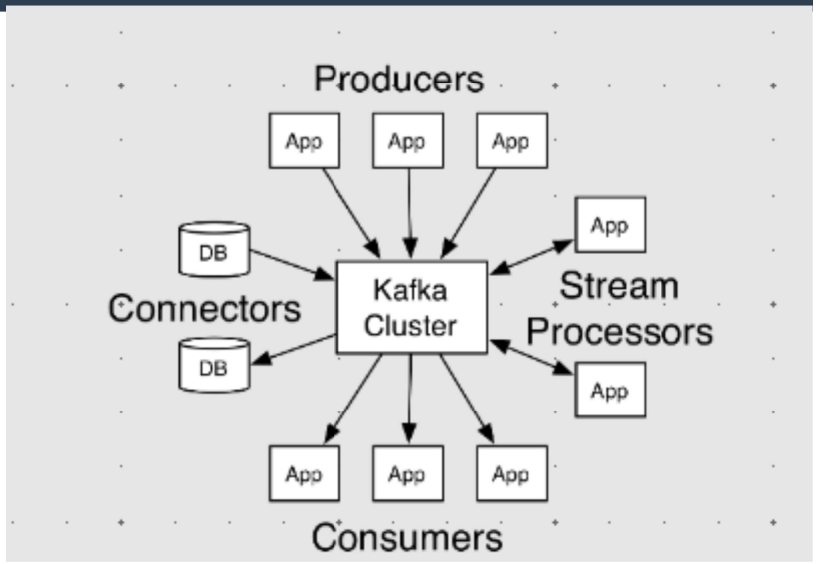
- JDBC Source Connector → Extrae datos de bases de datos como MySQL, PostgreSQL, Oracle.

- Elasticsearch Sink Connector → Envía datos de Kafka a Elasticsearch para búsquedas avanzadas.

- S3 Sink Connector → Guarda datos en Amazon S3.

- MongoDB Connector → Integra Kafka con MongoDB.

- Debezium Connector → Permite Change Data Capture (CDC) desde bases de datos como MySQL, PostgreSQL y MongoDB.



Confluent Kafka

Confluent es la empresa detrás de Apache Kafka, fundada por los creadores de Kafka.
https://www.confluent.io/

Apache Kafka

Licencia Apache v2.0
Kafka Brokers
Producer y Consumer API
Streams API
Connect API

Confluent Open

Clientes (Java, C, C++, Python, .)
Conectores (JDBC, Elasticsearch, HDFS, etc.)
Schema Registry
REST Proxy
KSQL

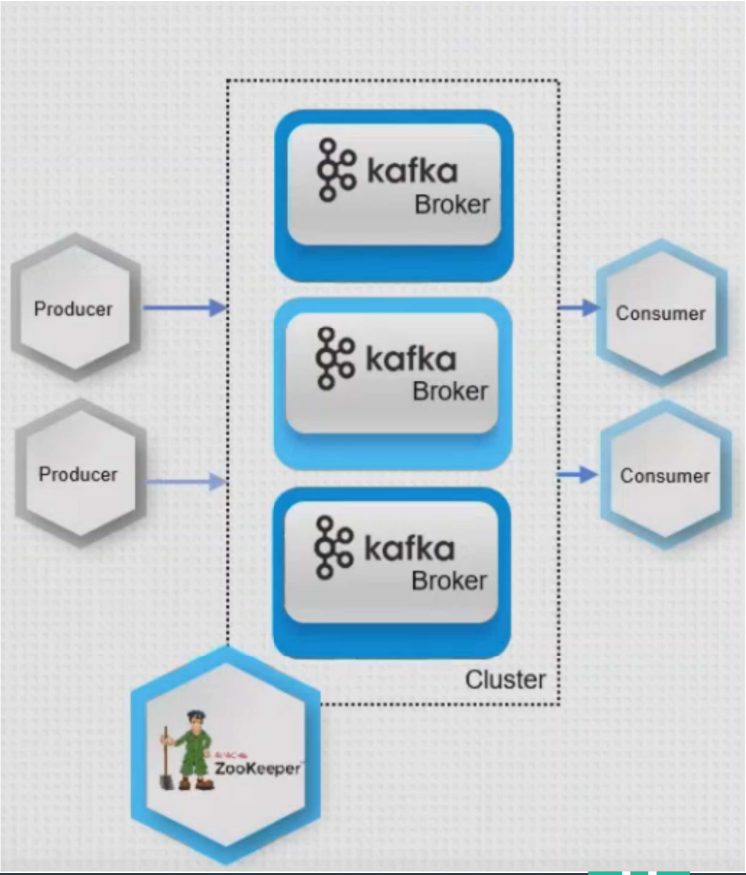
Confluent Enterprise

Control Center para monitoreo
Replicación en múltiples centros de datos
Confluent Cloud:
- AWS
- GCP

Trainings & certificaciones
Soporte 24/7

Kafka. Brokers

- Operan como parte de un clúster.
- Almacenan particiones y réplicas de topics: Los datos escritos en Kafka se guardan en disco y se replican para tolerancia a fallos.
- Reciben mensajes de los productores:
  - Les asignan offsets.
  - Confirman los mensajes y los almacenan en disco.
- Sirven a los consumidores:
  - Responden a solicitudes de recuperación de particiones.
  - Responden con los mensajes: Kafka utiliza un método de copia cero (zero-copy) para enviar los mensajes directamente desde el archivo al canal de red, sin buffers intermedios, lo que mejora el rendimiento.



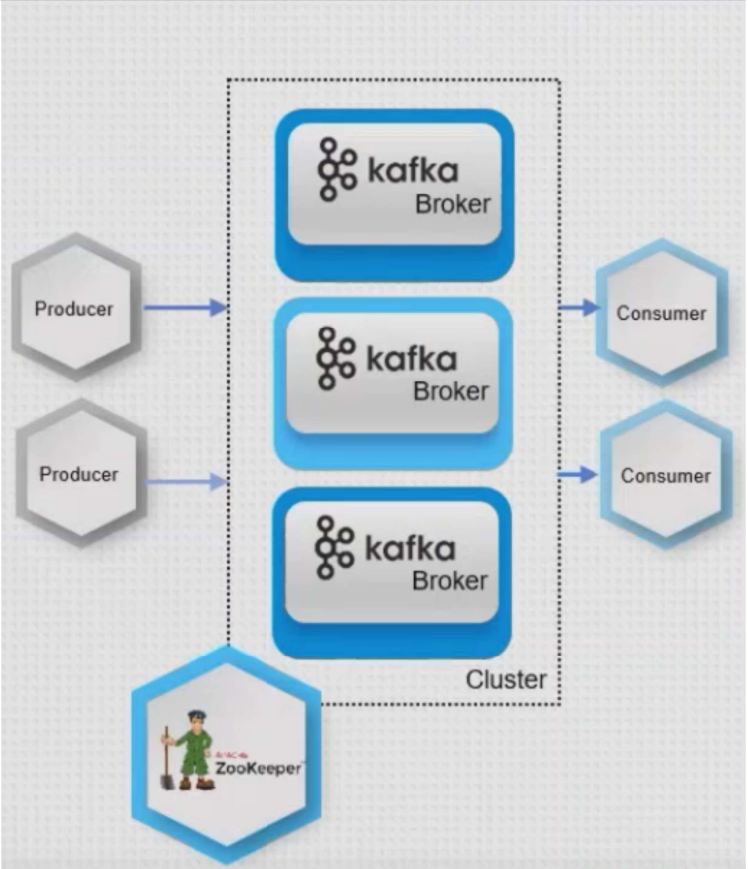
Kafka. Brokers

TOPIC equivalente a una tabla de Base de Datos (no es una tabla es un equivalente).

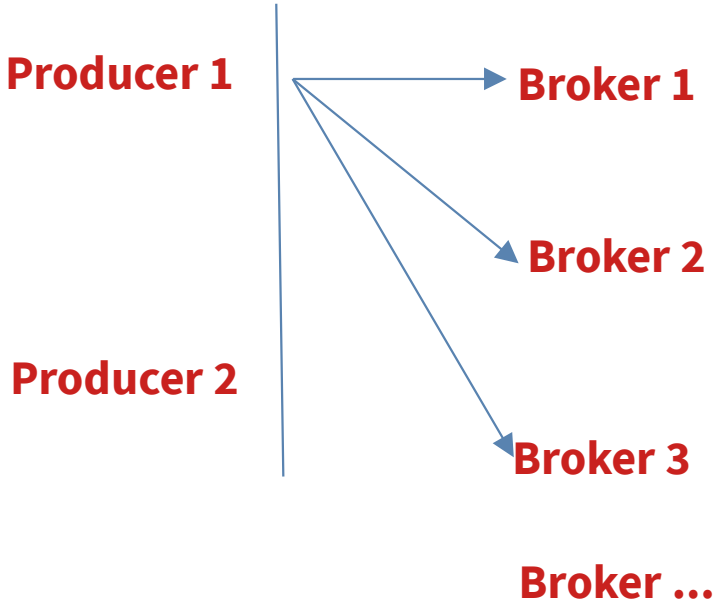
Ejemplo: Queremos guardar un topic que nos hemos creado que se llama “Topic Ventas”.

Pregunta: *Imaginad que el productor quiere enviar un mensaje que sea una Venta donde envía ¿Qué es lo que he vendido?; ¿A qué precio lo he vendido?; ¿A quién y qué día se lo he vendido? Lo metemos en un JSON.*
Si el productor lo envía : **¿se guarda en el primer, segundo y tercer Broker o sólo en alguno de ellos?**

¿Qué sería lo más lógico y lo mejor?



Kafka. Brokers

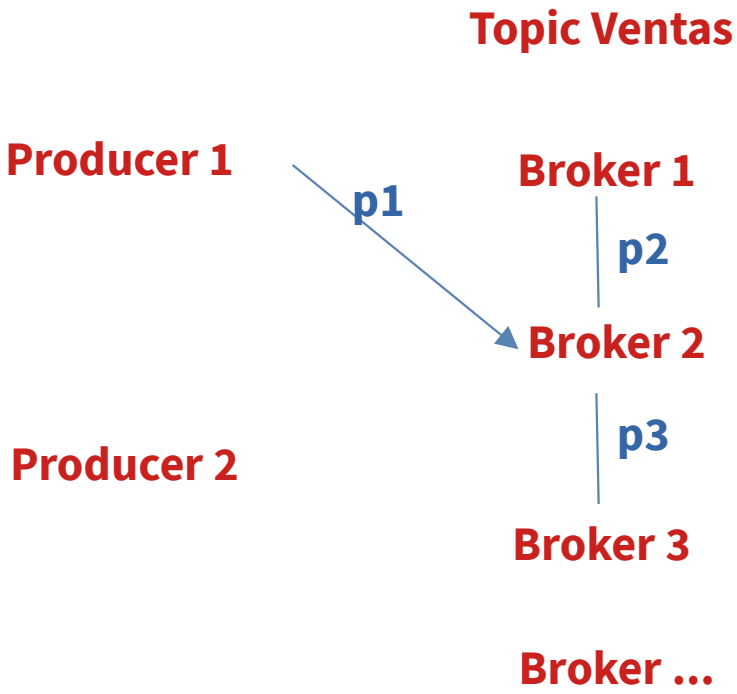


Broker trabaja en memoria y disco

¿Tiene sentido guardar en todos los brokers?

Backup

Kafka. Brokers



Pasos:
p1. Envío mensaje y lo guarda en el servidor Broker 2 (zooKeeper se lo dice). Obtiene respuesta Ok.
P2 Guarda un Backup en el Broker 1
P3 Guarda también un Backup en el Broker 3.
¿Hay algún problema en este comportamiento?

Brokers se conocen entre sí de forma limitada cada uno tiene una IP que se la asigna ZooKeeper y quién asigna quién debe ser Backup.

Kafka. Zookeeper

Lleva el registro del estado de los nodos del clúster de Kafka.

- Membresía del clúster se refiere al mecanismo mediante el cual los brokers y consumidores de un grupo coordinan su participación en el sistema distribuido.
- Elección del controlador.
- Configuración de topics.
- Listas de control de acceso.