For Dzone

Created with draw.io

@simplymanas

APACHE KAFKA

A distributed streaming platform

Kafka® is used for building real-time data pipelines and streaming apps. It is horizontally scalable, fault-tolerant, wicked fast, and runs in production in thousands of companies.

STREAMING PLATFORM

Publish and subscribe to streams of records, similar to a message queue or enterprise messaging system.

STORAGE SYSTEM

Any message queue that allows publishing messages decoupled from consuming them is effectively acting as a storage system for the in-flight messages. What is different about Kafka is that it is a very good storage system.

MESSAGING SYSTEM

As with publish-subscribe, Kafka allows you to broadcast messages to multiple consumer groups. The advantage of Kafka's model is that every topic has both these properties—it can scale processing and is also multi-subscriber—there is no need to choose one or the other.

KAFKA INTERNALS

- A CLUSTER has one or more brokers
- A BROKER has zero or one replica per partition
- A PRODUCER sends messages to a topic
- A CONSUMER subscribes to a topic
- A topic is replicated to one or more PARTITIONS
- A consumer is a member of a CONSUMER GROUP
- A TOPIC has zero or more conumsers
- A PARTITION has one consumer per group
- A partition has one or more REPLICAS
- An OFFSET is the number assgined to a record in a partition
- Each RECORD consists of a KEY, a VALUE and a TIMESTAMP
- The kafka cluster maintains a PARTITIONED LOG

KAFKA CORE APIS

PRODUCER API	CONSUMER API	STREAMS API	CONNECT API
The PRODUCER API allows applications to send streams of data to topics in the Kafka cluster.	The CONSUMER API allows applications to read streams of data from topics in the Kafka cluster.	The STREAMS API allows transforming streams of data from input topics to output topics.	The Connect API allows implementing connectors that continually pull from some source data system into Kafka or push from Kafka into some sink data system.

KAFKA GUARANTEES

- **Message delivery guarantees provided by Kafka are**
  - ***At most once***—Messages may be lost but are never redelivered.
  - ***At least once***—Messages are never lost but may be redelivered.
  - ***Exactly once***—Each message is delivered once and only once.
- **For Kafka node to be alive (**According to distributed systems**)**
  - A node must be able to **maintain its session** with ZooKeeper (via ZooKeeper's heartbeat mechanism)
  - If it is a follower it **must replicate the writes** happening on the leader and not fall "too far" behind

KAFKA STREAMS

1. **Simple & Light weight client library**

   KAFKA Streams can be easily embedded in any Java application and integrated with any existing packaging, deployment and operational tools that users have for their streaming applications.

2. **Exactly Once Processing semantic gurantee**

   In Kafka streams each record will be processed once and only once even when there is a failure on either Streams clients or Kafka brokers in the middle of processing.

3. **Stream Processor is a node in processor topology**

   Two special processor are there by the name 1. Source Processor (This processor doesn't have any up-stream processors , it consume records from 1 or multiple kafka topics and forward them to down-stream processors.) 2. Sink Processor (This proessor doesn't have any down-processors, it sends the received records from up-stream processors to a specified kafka topic)

Check my blog

Check my GitHub