Apache Kafka

PUC Minas Virtual

O que é o Kafka

- Plataforma de streaming distribuída
- Permite publish/subscribe de streams semelhante a uma fila de mensagens
- Trabalha em cluster com um ou mais nós
- Suporta milhões de mensagens por segundo

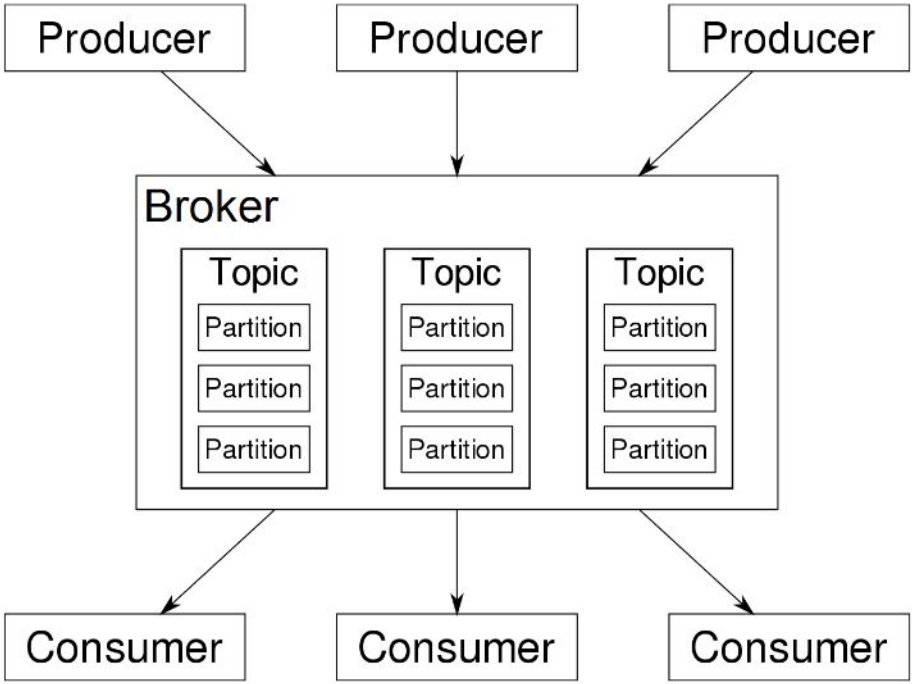
O que é o Kafka

- Entrega de mensagens de baixa latência
- Suporta múltiplos clientes como Java, .NET, PHP, Ruby, e Python
- Escrita e leitura em memória
- Persiste todos os dados no disco
- Integração nativa com Nifi, Spark Streaming e outras

História

- Nome em homenagem a Franz Kafka
- Nasceu para resolver problema de pipeline de dados do Linkedin
- ActiveMQ não escalava
- LinkedIn aumentou para mais de um trilhão de mensagens produzida (a partir de agosto de 2015) e PB de Dados

Arquitetura



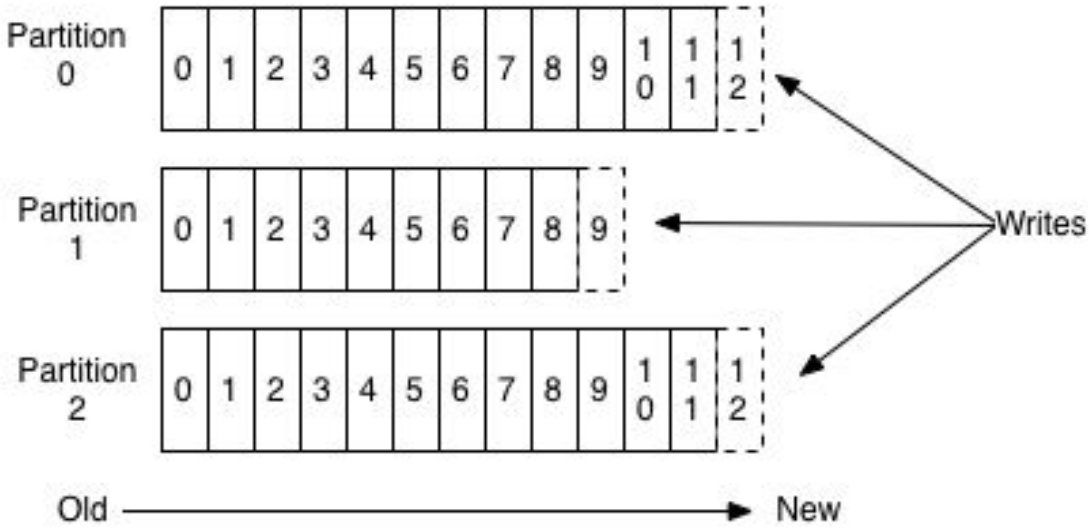
Tópico

- Onde os dados são armazenados
- São armazena em logs
- São multi-assinantes, podendo ter zero, um ou muitos consumidores
- São divididos em partições
- Os registros publicados no tópico são retidos por um período configurável de tempo

Partição

- Cada partição é uma sequência ordenada e imutável de registros que é anexado continuamente a um log de confirmação estruturado
- Os registros nas partições recebem cada um um número de identificação sequencial chamado offset que identifica de forma exclusiva cada registro dentro da partição
- São distribuídas pelos servidores no cluster Kafka
- É replicada em um número configurável de servidores para tolerância a falhas

Tópico com 3 Partições



Broker

- Kafka Cluster possui vários Broker (Instância de Kafka), se houve mais de um temos um Cluster Kafka
- Cada Broker pode ter zero ou mais partições por tópico
- Mantém o equilíbrio da carga
- São stateless, usam o ZooKeeper para manter seu estado no cluster
- Um Broker pode lidar com centenas de milhares de leituras e gravações por segundo e TB de mensagens sem perder desempenho

Producer

- Permite que um aplicativo publique registros nos tópicos
- Envia os dados para os Brokers
- Escolhe em qual tópico o dados vai ser escrito
- Quando um novo Broker é iniciado, todos os producers enviam uma mensagem para o ele
- Quanto mais partições, mais escalável é o envio de dados no producer

Consumer

- Lêem os dados do Broker
- Assinam um ou mais tópicos e consomem as mensagens publicadas
- Tem o controle das mensagens que foram consumidas usando o offset da partição
- Emite uma solicitação de offset assíncrona ao Broker para ter um buffer de bytes pronto para consumir

Zookeeper

- Parte importante de um Cluster Kafka
- Coordenação entre os Brokers e os Consumers
- Cluster Kafka compartilha informações através de um cluster Zookeeper
- Kafka armazena metadados básicos no Zookeeper
- Utilizado para fazer eleição de liderança dos Brokers e partições
- Fornece uma visão sincronizada da configuração do Kafka Cluster

Retenção

- Armazena os dados em disco
- Por padrão são armazenados 7 dias
- Após expiração as mensagens são excluídas automaticamente
- Configuração por tópico
- Cleanup também por espaço

Instalação

- Pode ser instalado em qualquer servidor (linux ou windows).
- Download:
  https://www.apache.org/dyn/closer.cgi?path=/kafka/3.1.1/kafka_2.12-3.1.1.tgz
- Após o download, descompactar o arquivo:
  tar -xzf kafka_2.12-3.1.1.tgz
- Acessar o diretório:
  cd kafka_2.12-3.1.1

Iniciando o Kafka

- Caso ainda não exista um Zookeeper instalado, iniciar pelo comando abaixo:
  nohup zookeeper-server-start.sh config/zookeeper.properties &

- Iniciar o Kafka:
  nohup bin/kafka-server-start.sh config/server.properties &

- Analisando o start:
  more nohup.out

- Verificando o processo kafka:
  ps -ef | grep kafka

Kafka no Docker

- Iniciando o Docker:
  docker-compose up -d kafka
- Acesso ao host kafka:
  docker exec -it kafka bash
- Diretório da Kafka:
  /opt/kafka/bin

Criando Tópico

- Para criação de tópico é utilizado o script kafka-topics no diretório bin
  ./kafka-topics.sh --create --zookeeper zookeeper:2181 --replication-factor 1 --partitions 1 --topic msg
  Onde:
  --zookeeper é o conjunto de zookeeper
  --replication-factor é o número de réplica para o tópico
  --partitions é o número de partições desejado para aquele tópico
  --topic é o nome do tópico a ser criado

- Ao criar um tópico é exibido a mensagem:
  Created topic "msg".

Verificando Tópico

- Para listar os tópicos existentes:
  ./kafka-topics.sh --list --zookeeper zookeeper:2181
- Para verificar as configurações do tópico:
  ./kafka-topics.sh --describe --zookeeper zookeeper:2181 --topic msg

Criando Producer

- Por padrão, a porta utilizada pelo Kafka Broker é a 9092, podendo ser alterada no arquivo de configuração.
  ./kafka-console-producer.sh --broker-list kafka:9092 --topic msg
- Após a abertura do producer, basta digitar as mensagens.
- Várias aplicações possuem clients para o Kafka, como por exemplo Java, Python e C#.

Criando Consumer

- Para iniciar um consumer é utilizado o script kafka-console-consumer.sh
  ./kafka-console-consumer.sh --bootstrap-server kafka:9092 --topic msg
- Ao iniciar o consumer, todas as mensagens a partir daquele momento serão recebidas no console.
- Para receber as mensagens desde o inicio do tópico, o parâmetro --from-beginning é utilizado.
  ./kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic msg --from-beginning

PUC Minas
Virtual