Kafka

Kafka란?

- Apache Kafka는 LinkedIn에서 시작되었으며 2011년에 오픈 소스 Apache 프로젝트가 된 후 2012 년에는 Apache 프로젝트가 됨
- Kafka는 Scala와 Java로 개발됨
- Apache Kafka는 게시 - 가입 기반 내결함성 메시징 시스템고 확장성이 뛰어남

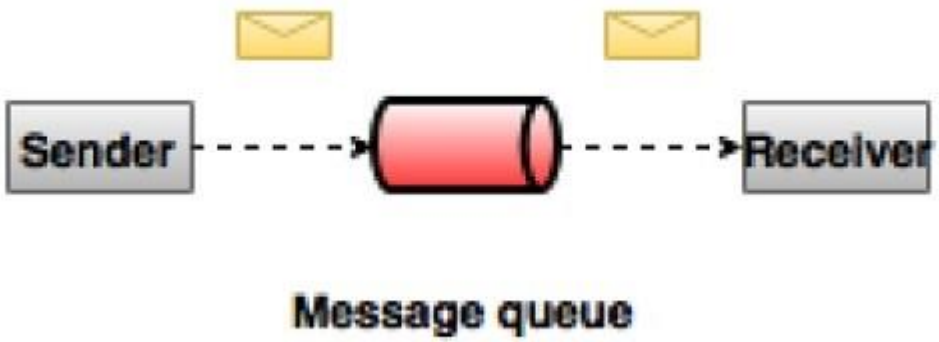
- 빅 데이터에서는 엄청난 양의 데이터가 사용됨. 데이터에 관해서 우리에게는 두 가지 주요한 어려움이 있음.
- 첫 번째는 대량의 데이터를 수집하는 방법이고 두 번째는 수집된 데이터를 분석하는 것임. 이러한 문제를 해결하려면 메시징 시스템이 필요함
- Kafka는 분산형 큰 처리량을 위해서서 만들어짐.
- 카프카(Kafka)는 전통적인 메시지 브로커를 대체함.
- 다른 메시징 시스템과 비교하여 Kafka는 처리량, 파티셔닝, 복제 및 내결함성이 뛰어나 대규모 메시지 처리 응용 프로그램에 적합함

메시징 시스템(Messaging System)이란 무엇입니까?

- 메시징 시스템은 한 응용 프로그램에서 다른 응용 프로그램으로 데이터를 전송함. 그러므로써, 응용 프로그램이 데이터에만 집중하고 공유 방법에 대해서는 걱정할 필요가 없음.
- 분산 메시징은 안정적인 메시지 대기열(queue) 개념에 기반함.
- 메시지는 클라이언트 응용 프로그램과 메시징 시스템 간에 비동기적으로 처리됨.
- 두 가지 유형의 메시징 패턴이 있음. 하나는 지점 간(point to point)이고 다른 하나는 게시 - 가입 (pub-sub) 메시징 시스템입니다. 대부분의 메시징 패턴은 pub-sub를 사용됨.

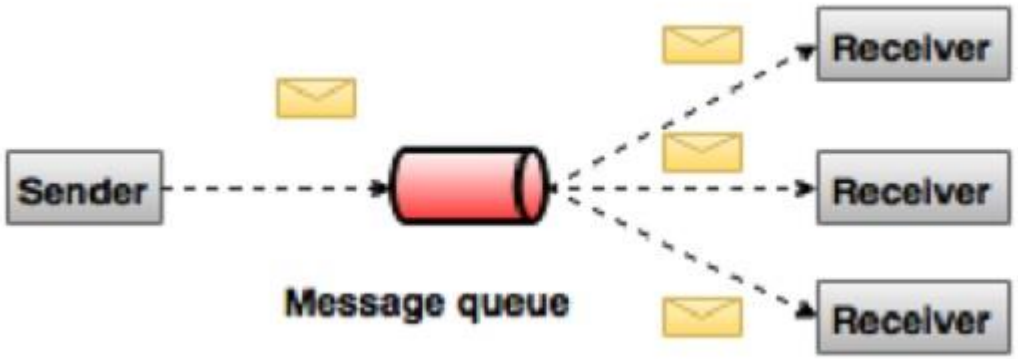
지점간 메세징 시스템(Point to Point Messaging System)

- 지점 간 시스템에서 메시지는 대기열에 유지됨.
- 하나 이상의 사용자가 대기열의 메시지를 사용할 수 있지만 특정 메시지는 최대 하나의 소비자 만 사용할 수 있음.
- 소비자가 대기열에서 메시지를 읽으면 해당 대기열에서 메시지가 사라짐.
- 예는 Order Processing System으로, 각 주문은 하나의 Order Processor에 의해 처리되지만, Multiple Order Processor는 동시에 작동할 수 있음.



게시-구독 메시징 시스템(Publish-Subscribe Messaging System)

- publish-subscribe 시스템에서 메시지는 Topic에 유지됨
- 지점 간 (point-to-point) 시스템과 달리 소비자는 하나 이상의 Topic를 구독하고 해당 주제의 모든 메시지를 사용할 수 있음.
- Publish-Subscribe 시스템에서 메시지 생성자는 게시자이고 메시지 소비자는 구독자임.
- 예는 스포츠, 영화, 음악 등과 같은 다양한 채널을 게시하는 TV. 누구나 자신의 채널 세트를 구독하고 구독 채널을 사용할 수있을 때마다 얻을 수 있음



Kafka란 무엇인가?

- Apache Kafka는 분산 발행 - 가입 메시징 시스템이며 대용량 데이터를 처리 할 수 있는 견고한 대기열(robust queue)이며 한 끝점에서 다른 끝점으로 메시지를 전달할 수 있게 함.
- Kafka는 오프라인 및 온라인 메시지 소비 모두에 적함.
- Kafka 메시지는 데이터 손실을 방지하기 위해 디스크에 유지되고 클러스터 내에 복제됨.
- Kafka는 ZooKeeper 동기화 서비스 위에 구축이 가능함.
- 실시간 스트리밍 데이터 분석을 위해 Apache Storm 및 Spark와 매우 잘 통합됨.

Kafka의 장점

- 안정성(**Reliability)** - Kafka는 분산, 파티션 분할, 복제 및 내결함성을 제공함.
- 확장성(**Scalability)** - Kafka 메시징 시스템은 다운 타임없이 쉽게 확장됨.
- 내구성(**Durability)** - Kafka는 분산 된 커밋 로그 를 사용하여 메시지가 가능한 한 빨리 디스크에 저장되므로 내구성이 우수함.
- 성능(**Performance)** - Kafka는 게시 및 구독 메시지 모두에 대해 높은 처리량을 제공함. 많은 TB의 메시지가 저장되는 경우에도 안정된 성능을 유지함.

Kafka 사용 사례

- 지표 - Kafka는 운영 모니터링 데이터로 자주 사용됩니다. 여기에는 분산 응용 프로그램의 통계를 집계하여 운영 데이터의 중앙 집중식 피드를 생성하는 작업이 포함됩니다.

- 로그 집계 솔루션 - Kafka는 조직 전체에서 여러 서비스의 로그를 수집하여 여러 사용자에게 표준 형식으로 제공 할 수 있습니다.

- 스트림 처리 - Storm 및 Spark Streaming과 같은 인기있는 프레임 워크는 주제에서 데이터를 읽고 처리하며 처리 된 데이터를 새로운 주제(Topic)로 작성하여 사용자 및 응용 프로그램에서 사용할 수 있게 합니다. Kafka의 강한 내구성은 또한 스트림 처리와 관련하여 매우 유용합니다.

Kafka의 필요성

- Kafka는 모든 실시간 데이터 피드를 처리하기위한 통합 플랫폼임. Kafka는 지연 시간이 짧은 메시지 전달을 지원하고 시스템 오류가 있는 경우 내결함성을 보장함.
- 다양한 소비자를 대처할 수 있는 능력이 있음.
- Kafka는 매우 빠르며 2 백만 건의 쓰기 / 초를 수행함.
- 카프카(Kafka)는 모든 데이터를 디스크에 보관하므로 기본적으로 모든 쓰기가 OS (RAM)의 페이지 캐시로 이동 처리함. 따라서 페이지 캐시에서 네트워크 소켓으로 데이터를 전송하는 것이 매우 효율적임

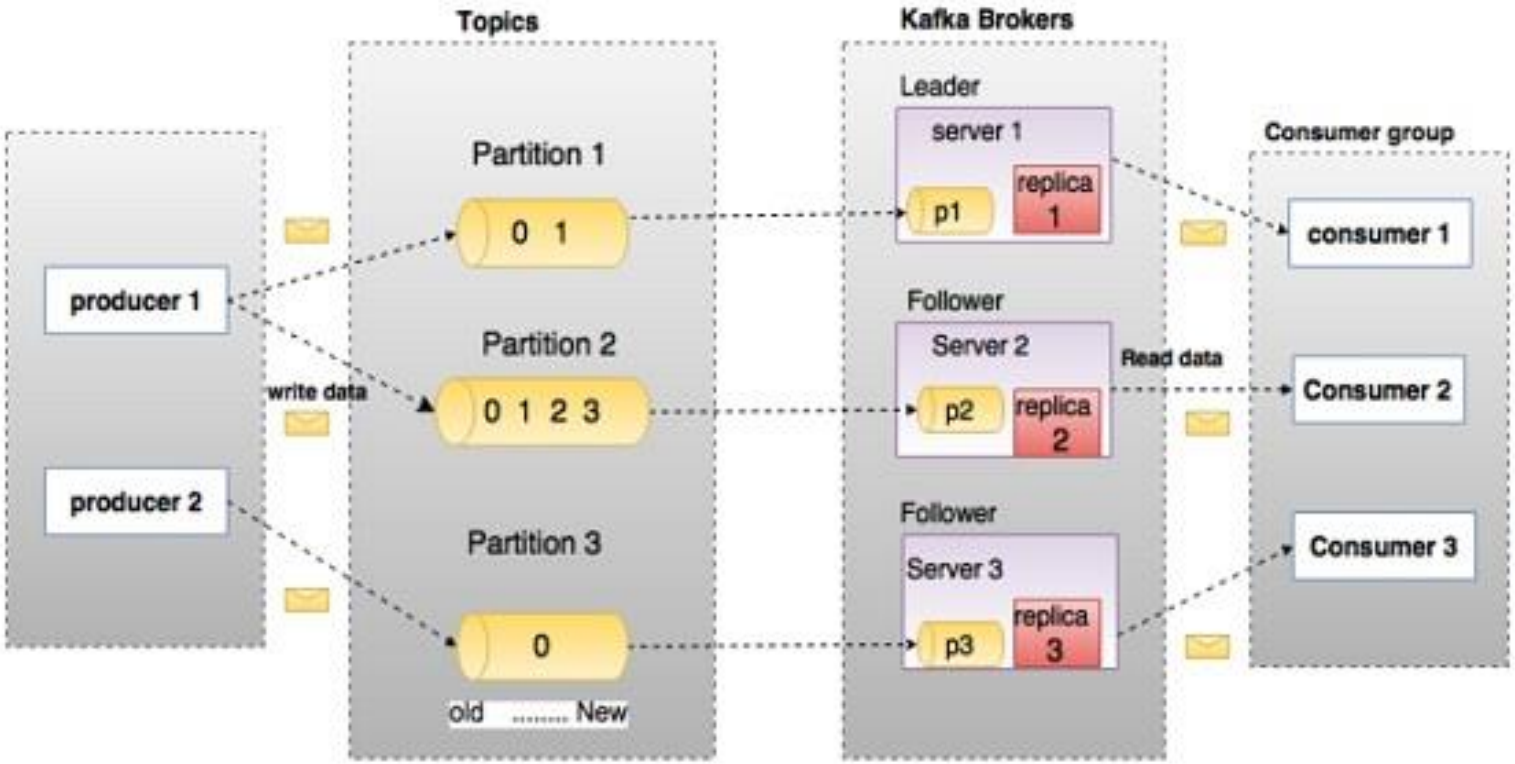
Kafka 용어 및 구조

- **Topics**
  - 특정 범주에 속하는 메시지 스트림을 Topic라고 합니다. 데이터는 Topic에 저장됨.
  - Topic 는 파티션으로 분할됨. 각 Topic 에 대해 카프카는 미니 파티션을 하나의 파티션으로 유지함. 이러한 각 파티션에는 메시지가 변경 불가능한 순서로 구성됨. 파티션은 동일한 크기의 세그먼트 파일 세트로 구현됨.
- **Partition**
  - Topic 에는 많은 파티션이 있을 수 있으므로 임의의 양의 데이터를 처리 할 수 있음.
- **Brokers**
  - 브로커는 게시된 데이터를 유지 관리하는 시스템임. 각 브로커에는 Topic 당 0 개 이상의 분할 영역이 있을 수 있음. 주제에 N 개의 파티션이 있고 N 개의 브로커가 있는 경우 각 브로커에는 하나의 파티션이 있다고 가정함.

Kafka 용어 및 구조

- **Producers**
  - Producers 하나 이상의 카프카 (Kafka) Topic에 대한 메시지 게시자임. 생산자는 카프카 중개인에게 데이터를 보냄. 생산자가 브로커에 메시지를 게시 할 때마다 브로커는 마지막 세그먼트 파일에 메시지를 간단하게 추가함. 메시지는 파티션에 추가됨. 생산자는 원하는 파티션에 메시지를 보낼 수도 있음.
- **Consumers**
  - Consumers는 브로커로부터 데이터를 읽음. 소비자는 브로커에서 데이터를 가져 와서 하나 이상의 주제를 구독하고 게시 된 메시지를 사용함.

Kafka 용어 및 구조


Topics
Kafka Brokers
Leader
server 1
p1
replica 1
Partition 1
0 1
Consumer group
consumer 1
producer 1
Follower
Server 2
Partition 2
Read data
write data
0 1 2 3
p2
replica 2
Consumer 2
producer 2
Follower
Partition 3
Server 3
Consumer 3
p3
replica 3
0
old ..... New

Kafka Cluster 구조