Apache Kafka

Kafka'nın Amacı

Önce buffer gibi dursun herşeyi oraya yazsın, sonra ben oradan diğeristekleri beslerim. (online ve offline olarak)

Kafka Pub-Sub ve Queueing yapmayı salar.

Neden Kafka?

Hem mesajlaşmayı sağlar hemde üzerinde akan datayı işleyebilir. (streaming)

Hadoop ise datayı işlemek için kolaylık sağlar. Eskiden datayı içeri almak çok zordu. Bu sorunu Kafka ile giderdiler.

Kafka tüm verileri diske aktarır. Yani tüm yazma işleri işletim sisteminin Page Cache'e(RAM) gittiği anlamına gelir. Bu, verilerin Page Cache'ten Network Socket' ine aktarılmasını çok verimli kılar.

Kafka Yararlar

Güvenilirlik - Kafka dağıtım, bölümlenme, kopyalanma ve hataya dayanıklılık sağlar.

Ölçeklenebilirlik - Kafka mesajlaşma sistemi zaman aşımına uğramadan kolayca ölçeklenir.

Dayankllk - Kafka dağıtılmış commit log kullanır; bu, mesajların mümkün olduğunca hızlı bir şekilde diskte kalması anlamına gelir, bu nedenle dayanklıdır.

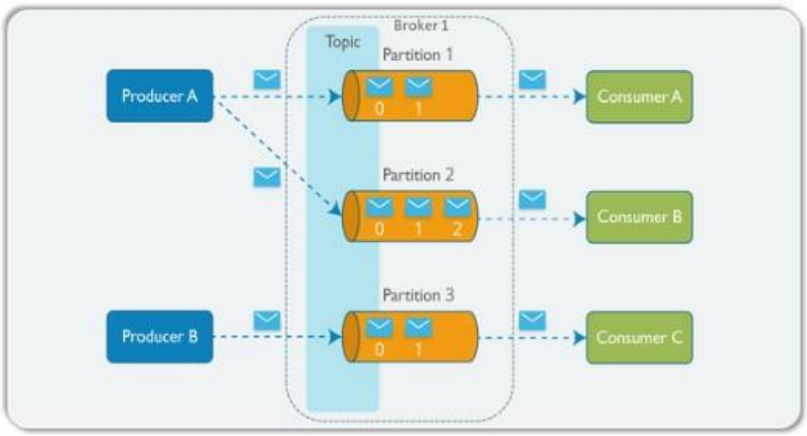
Performans - Kafka, hem mesaj yayınlama hem de abone olma konusunda yüksek verime sahiptir. Birçok TB mesajının bile saklanması durumunda istikrarlı bir performans sağlar.

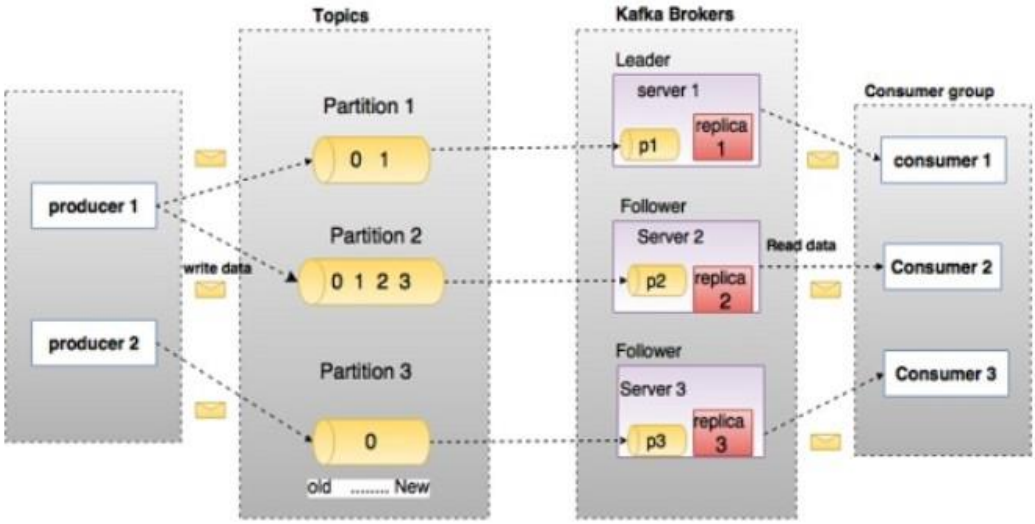
Kafka çok hızlıdır. Sıfır kesinti ve sıfır veri kaybını garanti eder.

Not: Commit log; kullanıcının yaptığı değişiklikleri tarif ettiği, anlattığı kayıttır.

Kafka Temel Kavramlar:

- Mesajlar **“Topic”**lerde tutulur.
- **“Topic”**lere mesaj gönderenler **“producer”**lardır.
- **“Topic”**leri okuyanlar ise **“consumer”**lardır.
- Kafka’nın çalıştığı her bir server ise **“broker”**dır. Birçok Broker bir araya gelerek **Cluster** yapısını oluşturur.
- Sunucular ve istemciler arası haberleşme tamamen dil bağımsız bir TCP protokolü üzerinde olur. Böylece popüler dillerde client kütüphaneleri mevcuttur.





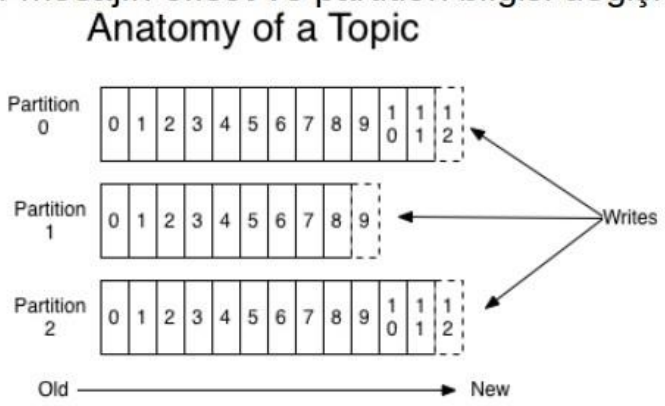
- Kafka Cluster, kesinti olmadan genişletilebilir.

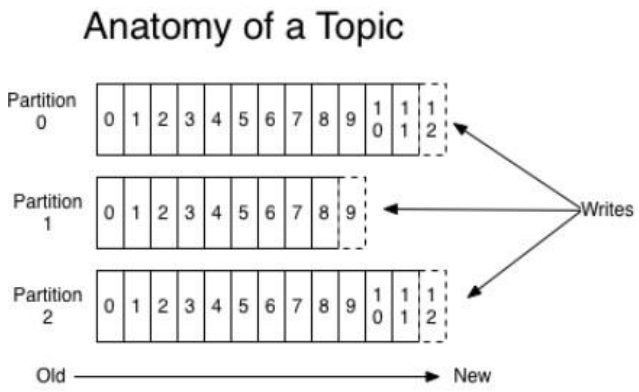
Topic Mimarisi:

•**Topics:** Mesajların tutulduğu yerdir.Kafka içerisinde birçok topic olabilir, isim ile tanımlanırlar.

•**Partitions:** Topic içerisinde bir ya da birden fazla bölümden oluşur. Her Partition'ın 0'dan başlayan numaraları vardır.

•**Offsets:** Topic'e gelen her yeni mesaj sona eklenir ve offset numarası bir bir artar. Bir kere bir bölüme yazılan bir mesajın offset ve partition bilgisi değişmez.



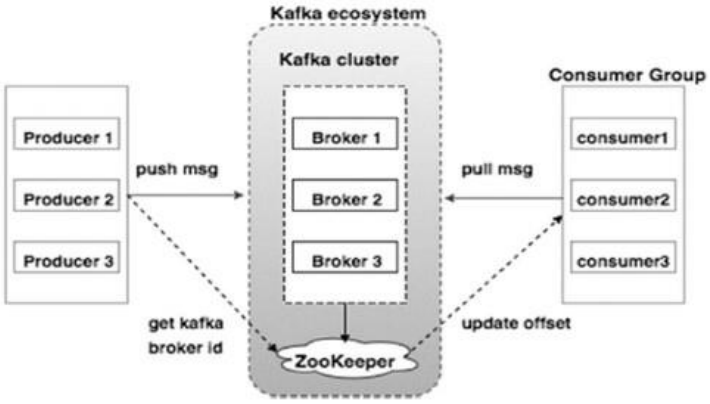


- Topic bazında bir ordering sağlamaz.

  Yani Partition 0'ın 11. Offset'i 12. Offset'ten önce gelir ama Partition 1'in 9. Offset'i daha yeni atılmış olabilir.

  Mesaj okuma Partition 2'den de başlayabilir.

Kafka Cluster Diyagram



- **Brokerlar,** vatansızdır(stateless). Bu yüzden ZooKeeper'ı cluster durumlarını korumak için kullanırlar.
- Lider seçimi **Zookeeper** tarafından yapılır.

Zookeeper

- Kafka Brokerını **yönetmek** ve **koordine** etmek için kullanılır.
- Kafka Zookeeper'da **topicler**, **brokerlar**, **consumer offsetleri** (queue readers) vb. bilgiler gibi temel metadatalar depolar.
- Bir Broker'a birşey olursa (fail) Partitionlar daha önce kopyaladığı (replica) için diğer Broker'dan (sunucu) devam eder. Bunu Zookeeper kontrol eder. Lider seçimini yapar.

Not: Eğer Leader'ı kaybedersen herşey biter.

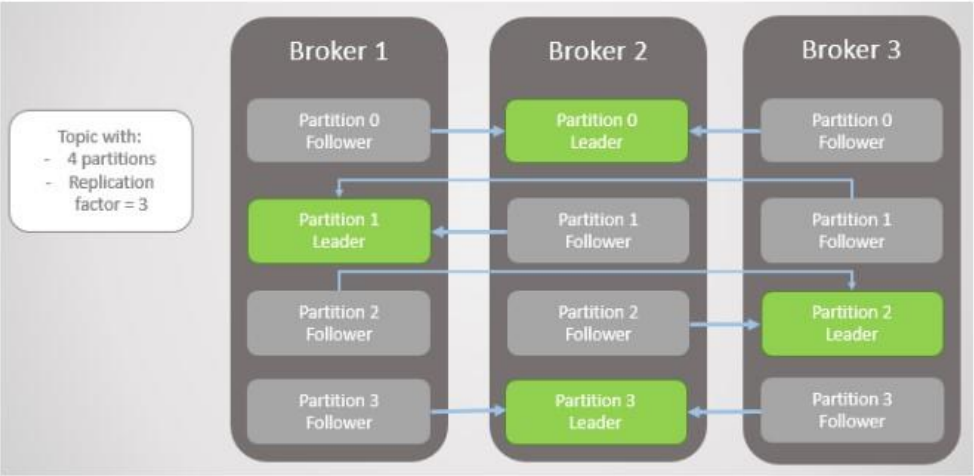
- Bir Broker fail olurca performans kaybı olur ama çok büyük değildir. Bunun sebebi kullanılan farklı tekniklerdir. Mesela; sıkıştırıp diske yazar ve bunu sırayla okur. (RAM üzerinden erişmeye çalmaz.)

Not: Diskten random access çok yavaştır.

- Consumer ve Producer Kafka sistemindeki yeni bir brokerın var olunca veya Broker'ın Kafka sistemindeki başarısızlığı konusunda bilgilendirir. Consumer başka bir Broker ile görevleri koordine etmeye başlama kararı alır eğer hata olursa.

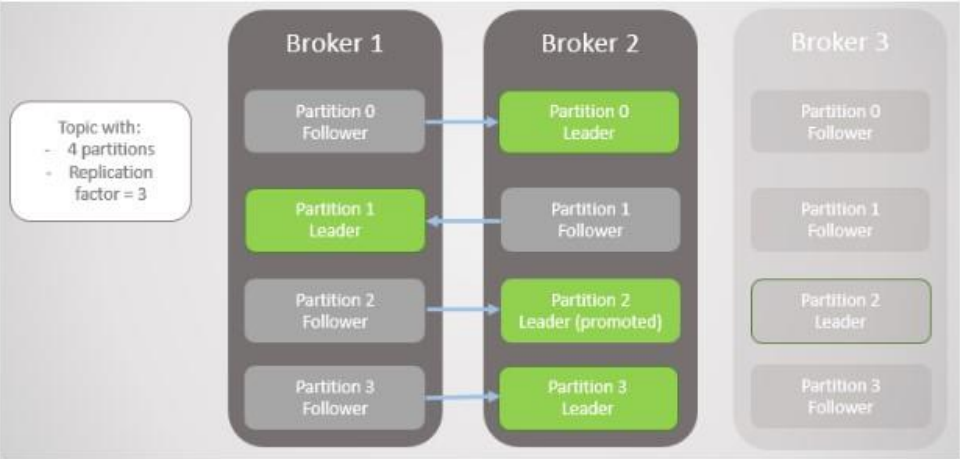
Kafka ile Mesajlar Nasıl İletilir?

- Mesajlar ilk olarak yeşil ile gösterilen Leader bölümlerine gönderilir.
- Leader aldığı mesajı gri ile gösterilen follower bölümlerine iletir. Böylece bir mesaj birden fazla broker üzerinde saklanır , makinelerden bir tanesi çökmüş olsa bile mesajları kaybetmemiş oluruz

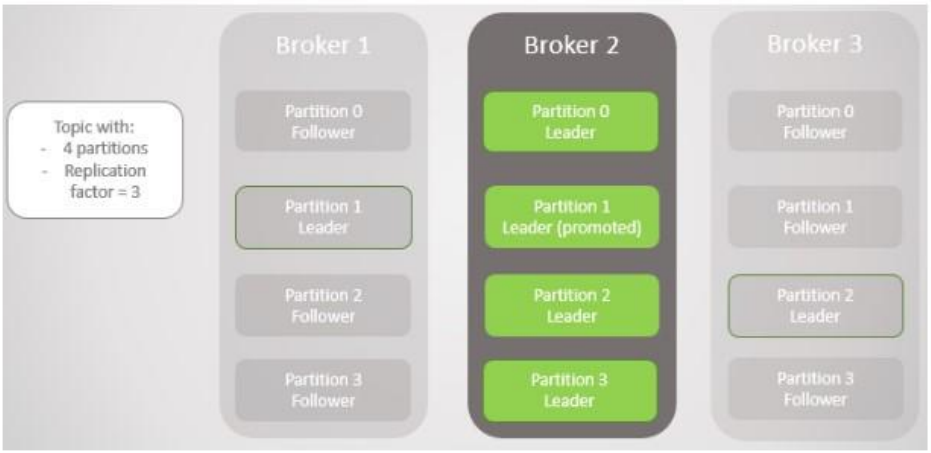


Partition Başarısızlığı

- Broker 3 öldüğünde, muhtemelen birden fazla partition liderinin ev sahibi(host) olacaktır. Liderini kaybeden her partition için, kalan node'daki bir follower Liderliğe terfi ettirilecektir.



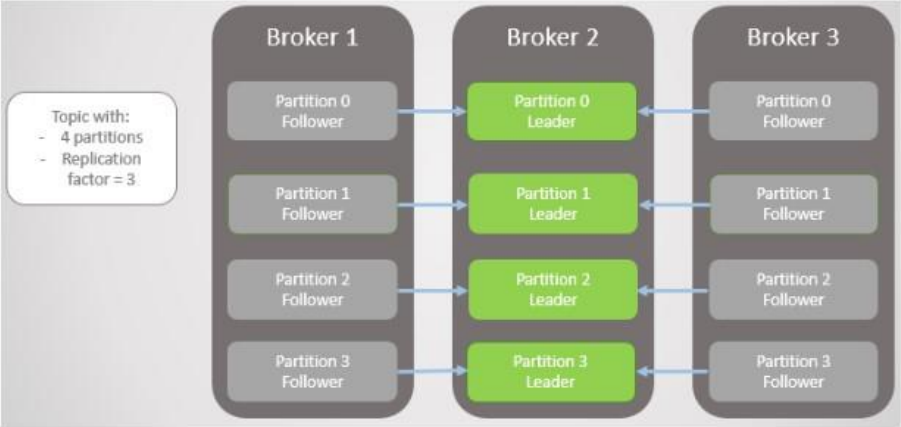
- Broker 1'de öldüğünde Partition 1 Liderini kaybeder ve Broker 2'deki Partition Follower iken Lider olur.



Partition Düzeltmeleri

- Brokerlar çevrimiçi oldukça aşağıdaki followerlar geri gelir ama Liderler ilk oldukları Brokerlara geri dönmezler en son kaldıkları Broker 2'de kalırlar.

  Not: Tüm Brokerların çökmesi çok nadir bir olaydır. Böyle bir durum olursa ilk Broker'ı yeniden başlatır. Eğer hala sorun devam ediyorsa Kafka Cluster içinde yeni bir broker oluşturmaya çalışır ancak bu karmaşıktır ve DevOps ekibinin süreci uygulayıp sürdürmesini gerektirir.



- Kafka eski haline getirmek için RabbitMQ'dan daha iyi araçlar sunar.
- Bunun 2 yöntemi vardır:
  - Auto.leader.rebalance.enable = true ile yeniden lider eski haline atanır.
  - Bir admin kafka-preferred-replica-election.sh scriptini manuel olarak canlandırmak için kullanabilir .