네트워크 부하를 고르게 분포시키기 위한

리더 파티션 밸런싱

@Current 2024 / Lightning Talk

Is your Kafka 'truly' load balancing?

Pro tips for leader partition distribution with network usage

jwsong@spitha.io

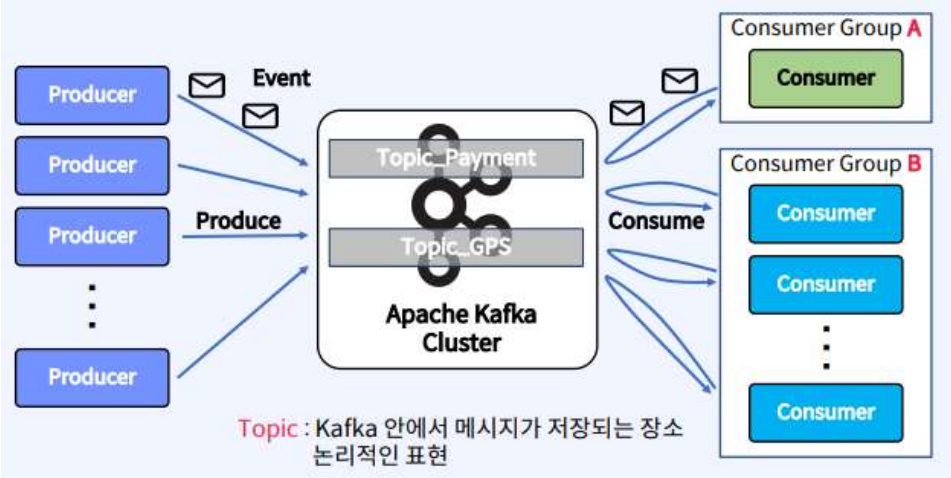
Overview

SPITHA

Felice

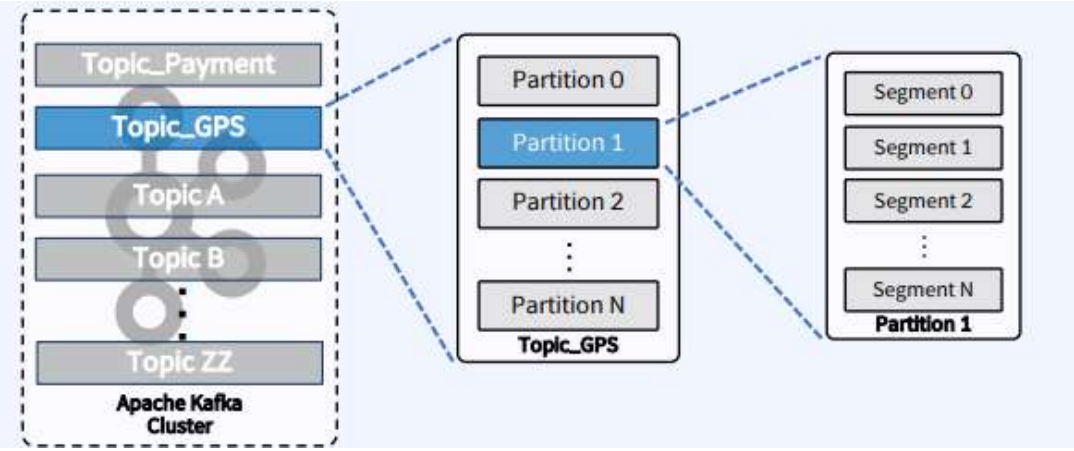
- 토픽 / 파티션 / 리더 파티션
- 리더 파티션 밸런싱 / 선호하는 리더
- 간단한 실험
- 트래픽 기반 리더 밸런싱

카프카 overview



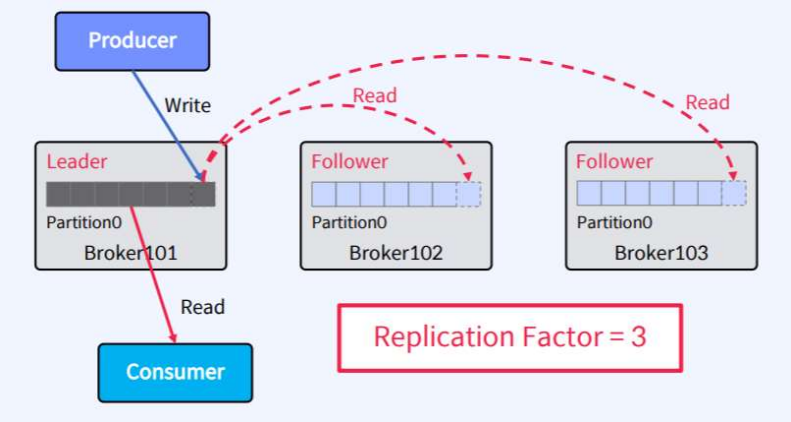
- 프로듀서는 이벤트 메시지를 생성하여 하나 이상의 브로커로 구성된 카프카 클러스터의 특정 토픽으로 보내고
- 브로커는 이 메시지를 토픽에 저장
- 컨슈머 그룹에 속한 컨슈머는 특정 토픽에 도착한 메시지를 읽어 가서 처리

토픽/파티션



- 토픽은 논리적인 단위
- 하나의 토픽은 하나 이상의 파티션으로 구성
- 하나의 파티션은 하나 이상의 세그먼트로 구성

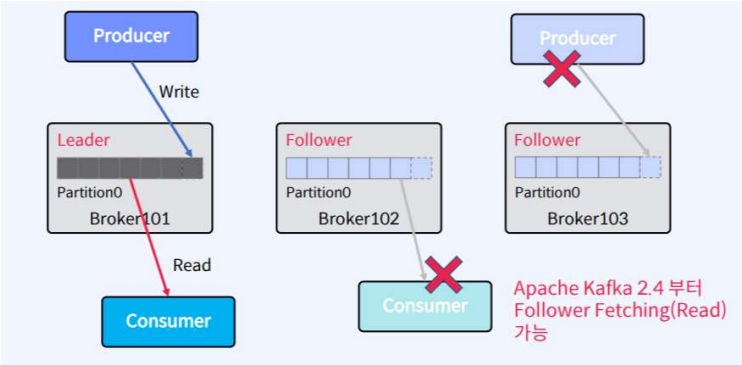
파티션 복제



- 파티션을 복제하여 다른 브로커에 복제본을 생성하여 브로커 장애에 대비
- 프로듀서가 하나의 파티션 복제본에 데이터를 write 하면
- 다른 브로커에서 데이터를 가져가서 복제본에 write
- 복제 수 (Replication Factor)는 토픽을 생성할 때 결정

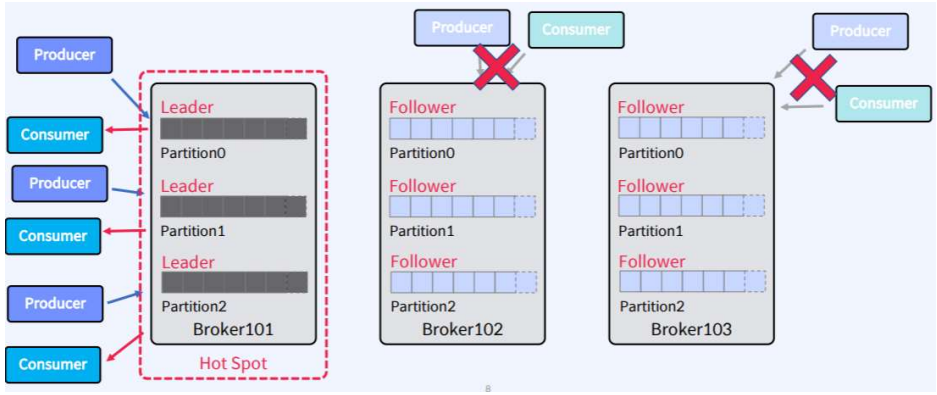
리더 파티션

- 복제본 중 하나를 Leader 로 선정
- 프로듀서와 컨슈머는 리더에서 Write/Read
- Follower는 장애에 대비하기 위해서만 유지됨
- 리더는 팔로워의 상태도 확인
- Kafka 2.4.0부터 Follower Fetching이 가능해짐

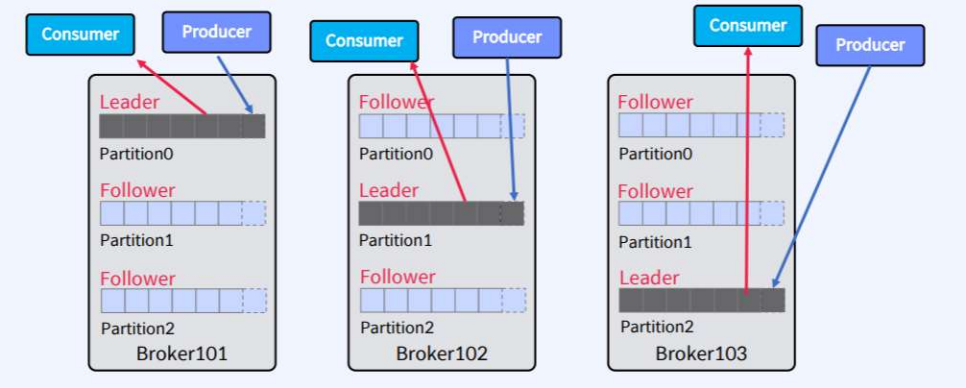


리더 파티션 밸런싱

- 동일한 쓰기
  - 파티션의 모든 복제본에서 쓰기 양은 동일
- 읽기는 리더 파티션
  - 읽기 작업은 리더 파티션에 집중 됨
- 따라서 부하는 집중됨
  - 3개의 복제본에 2개의 컨슈머 그룹이 있는 경우 리더는 최대 5배의 부하를 가지게 됨
- 핫 스팟 리스크
  - 리더가 하나의 브로커에 몰린 경우 해당 브로커는 대부분의 부하를 감당하게 됨

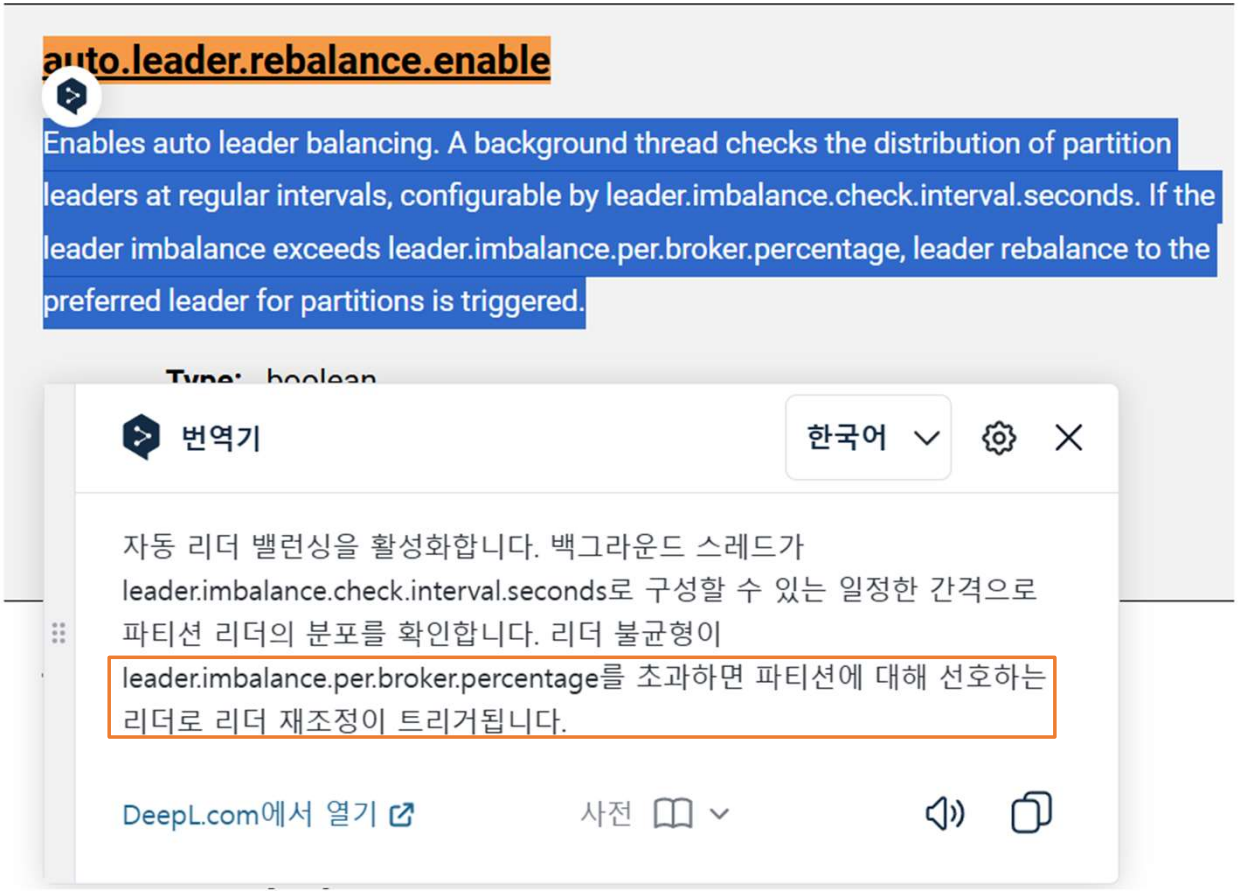


리더 파티션 밸런싱



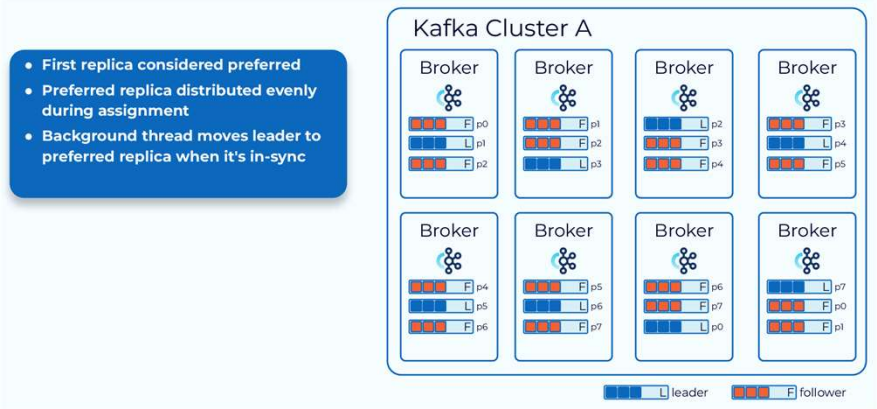
- 카프카 클러스터는 주기적으로 리더 분포의 불균형을 확인하고 조정
  - *auto.leader.rebalance.enable=true*
  - *leader.imbalance.check.interval.seconds=300*
  - *leader.imbalance.per.broker.percentage=10*

auto.leader.rebalance.enable



리더 선출

- 균형 분포
  - 브로커 사이에 부하를 고르게 분산 시키기 위해
- 리더 선출
  - 만약 리더 파티션이 있는 브로커가 문제가 생기면 → 팔로워 파티션을 리더로 선출
- 리더 복구
  - 복구된 원래 리더는 팔로워로 출발, 현재 리더로부터 데이터 복제
- 리더 복귀
  - 모든 데이터를 따라 잡으면 → 거의 대부분 원래 리더로 복귀 됨 (선호하는 리더이기 때문에)



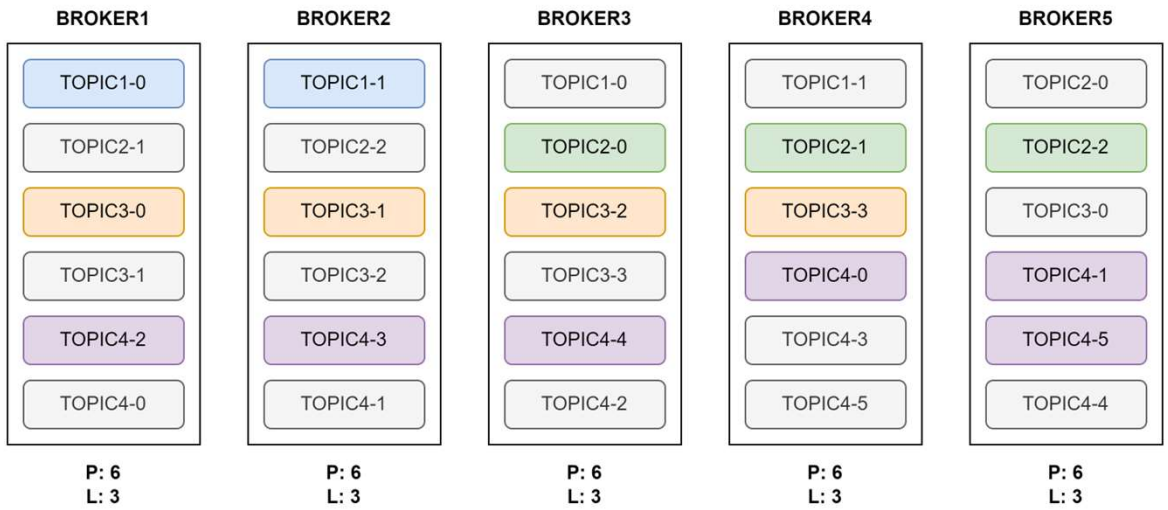
선호하는 리더 선정

SPITHA
Felice

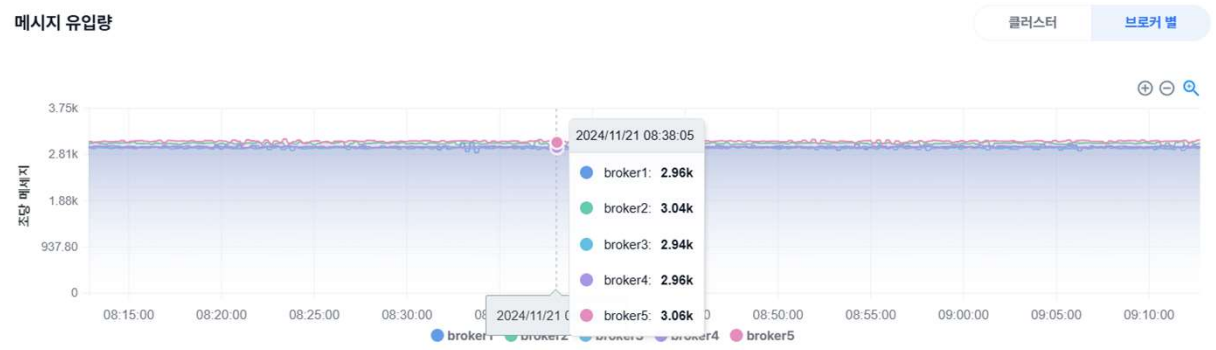
- 선호하는 리더(Preferred leader)
  - 토픽 구성에서 Replica 목록의 첫 번째 브로커에 있는 복제본
  - 브로커에 문제가 없으면 거의 99% 리더 파티션으로 선택됨
  - 자동으로 이루어지는 리더 리밸런스에서 선호하는 리더는 변경되지 않음
- 토픽을 생성할 때
  - 적당한 브로커들에 새로운 토픽의 파티션을 골고루 배치
  - 복제 설정이 2 이상인 경우, 선호하는 리더도 골고루 배치
- 운영 중에
  - 리더 파티션의 분포가 임계치를 넘더라도 **선호하는 리더를 변경하여 균형을 자동으로 맞추어 주지 않음**

실험을 위한 구성

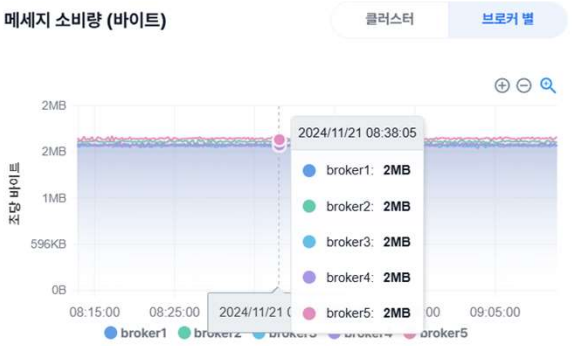
- 4 topic : 2 replicas
  - TOPIC1 : 2 partitions
  - TOPIC2 : 3 partitions
  - TOPIC3 : 4 partitions
  - TOPIC4 : 6 partitions
- 5 brokers
  - 6 partitions each
  - 3 leader partitions each



기대하는 결과


메시지 유입량
클러스터
브로커 별
3.75k
2.81k
1.88k
937.80
0
조당 메세지
2024/11/21 08:38:05
broker1: 2.96k
broker2: 3.04k
broker3: 2.94k
broker4: 2.96k
broker5: 3.06k
08:15:00
08:20:00
08:25:00
08:30:00
08:50:00
08:55:00
09:00:00
09:05:00
09:10:00
broker5


메세지 유입량 (바이트)
클러스터
브로커 별
2MB
2MB
1MB
594KB
0B
조당 바이트
2024/11/21 08:38:05
broker1: 2MB
broker2: 2MB
broker3: 2MB
broker4: 2MB
broker5: 2MB
08:15:00
08:25:00
09:05:00
broker1
broker5


메세지 소비량 (바이트)
클러스터
브로커 별
2MB
2MB
1MB
596KB
0B
조당 바이트
2024/11/21 08:38:05
broker1: 2MB
broker2: 2MB
broker3: 2MB
broker4: 2MB
broker5: 2MB
08:15:00
08:25:00
09:05:00
broker1
broker5

현실은

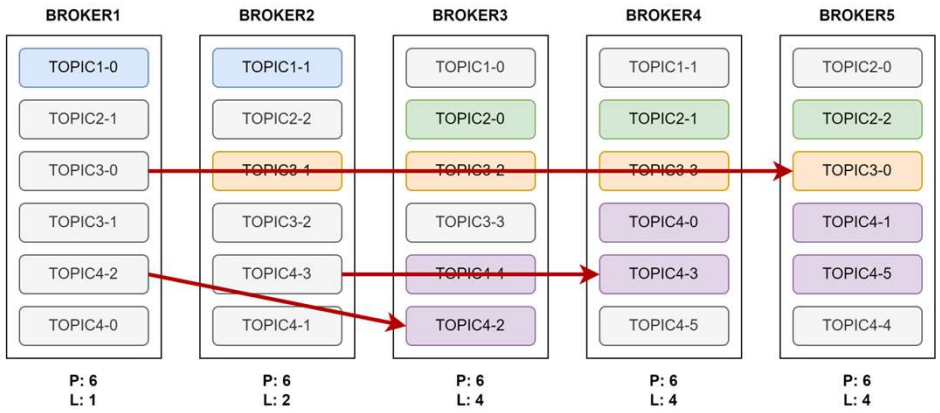

메시지 유입량
클러스터
브로커 별
Observe
Rate adjustment
조당 메세지
7.39k
5.54k
3.69k
1.85k
0
08:50:00
08:55:00
09:00:00
09:05:00
09:10:00
09:15:00
09:35:00
09:40:00
09:45:00
2024/11/21 09:28:40
broker1: 5.91k
broker2: 6.08k
broker3: 2.96k
broker4: 2.96k
broker5: 3.10k
broker1
broker2
broker3
broker4
메세지 유입량 (바이트)
클러스터
브로커 별
Observe
조당 바이트
5MB
3MB
2MB
1MB
0B
08:50:00
09:00:00
2024/11/21 09:28:40
broker1: 4MB
broker2: 4MB
broker3: 2MB
broker4: 2MB
broker5: 2MB
09:40:00
broker1
broker2
메세지 소비량 (바이트)
클러스터
브로커 별
Observe
조당 바이트
5MB
3MB
2MB
1MB
0B
08:50:00
09:00:00
2024/11/21 09:28:40
broker1: 4MB
broker2: 4MB
broker3: 2MB
broker4: 2MB
broker5: 2MB
09:40:00
broker1
broker2

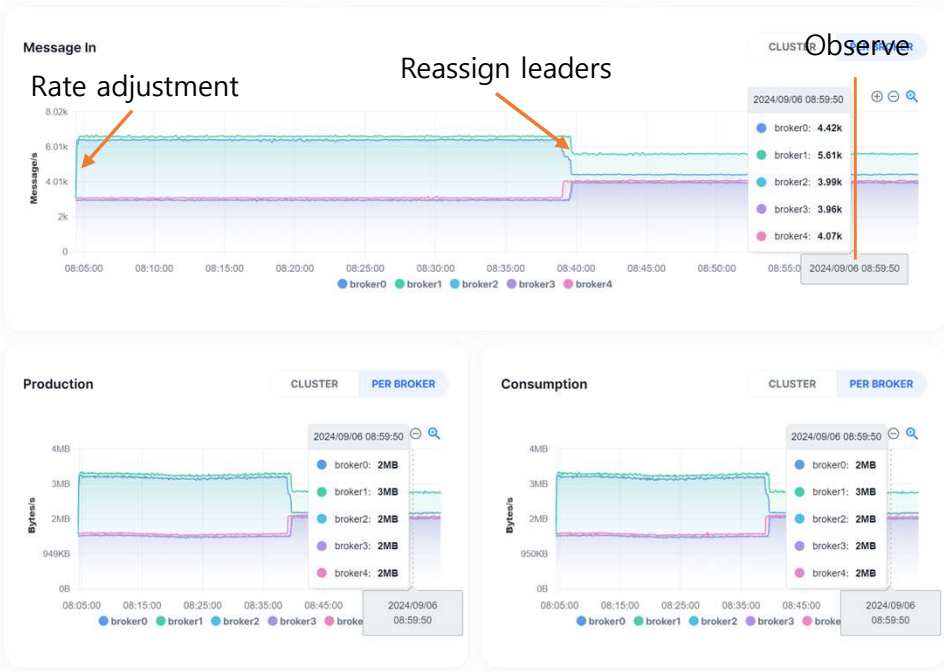
트래픽 기반 리더 밸런싱

SPITHA
Felice

- 트래픽 기반 밸런싱
  - 단순 개수 기준이 아니라 각 브로커에 트래픽을 고르게 분산 시킬 수 있도록 배치 하는 것이 필요함
- Kafka Metrics
  - 카프카는 이미 충분한 메트릭을 제공하고 있음
    - *kafka_server_brokertopicmetrics_(replication)bytes[in|out]_total*
- 리더 재할당
  - CLI도구와 Admin API를 이용해 동적으로 재할당 가능
    - *bin/kafka-reassign-partitions.sh*
    - *org.apache.kafka.clients.admin.KafkaAdminClient.alterPartitionReassignments()*
  - 브로커간 데이터 복제가 필요 없으며 최소한의 오버헤드로 빠르게 수행됨

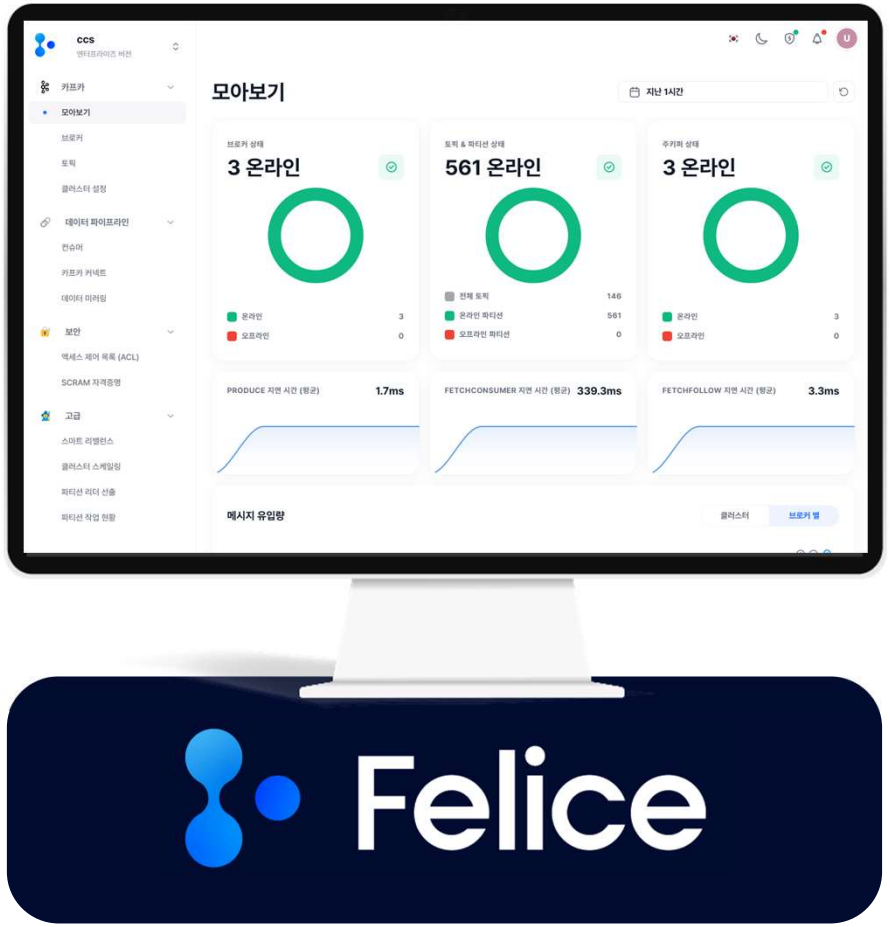
더 나은 결과


BROKER1
BROKER2
BROKER3
BROKER4
BROKER5
TOPIC1-0
TOPIC2-1
TOPIC3-0
TOPIC3-1
TOPIC4-2
TOPIC4-0
TOPIC1-1
TOPIC2-2
TOPIC3-2
TOPIC4-3
TOPIC4-1
TOPIC1-0
TOPIC2-0
TOPIC3-3
TOPIC4-2
TOPIC4-5
TOPIC1-1
TOPIC2-1
TOPIC4-0
TOPIC4-3
TOPIC4-5
TOPIC2-0
TOPIC2-2
TOPIC3-0
TOPIC4-1
TOPIC4-5
TOPIC4-4
P: 6
L: 1
P: 6
L: 2
P: 6
L: 4
P: 6
L: 4
P: 6
L: 4


Message In
Rate adjustment
Reassign leaders
Observe
Production
Consumption
CLUSTER
PER BROKER

Kafka도 쉬워질 수 있습니다

Kafka도 쉬워질 수 있습니다



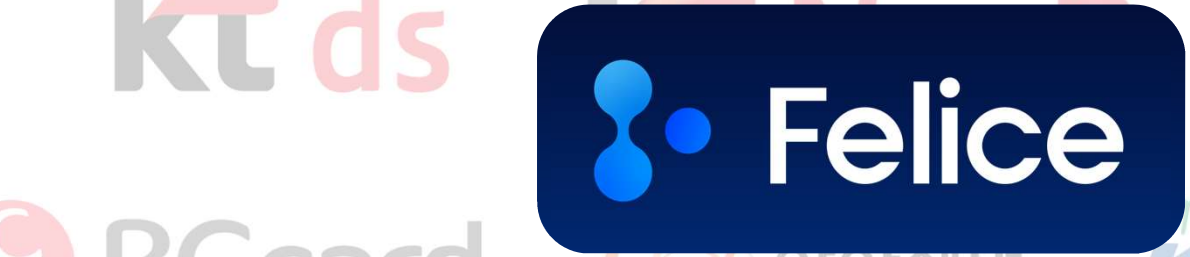
치트키

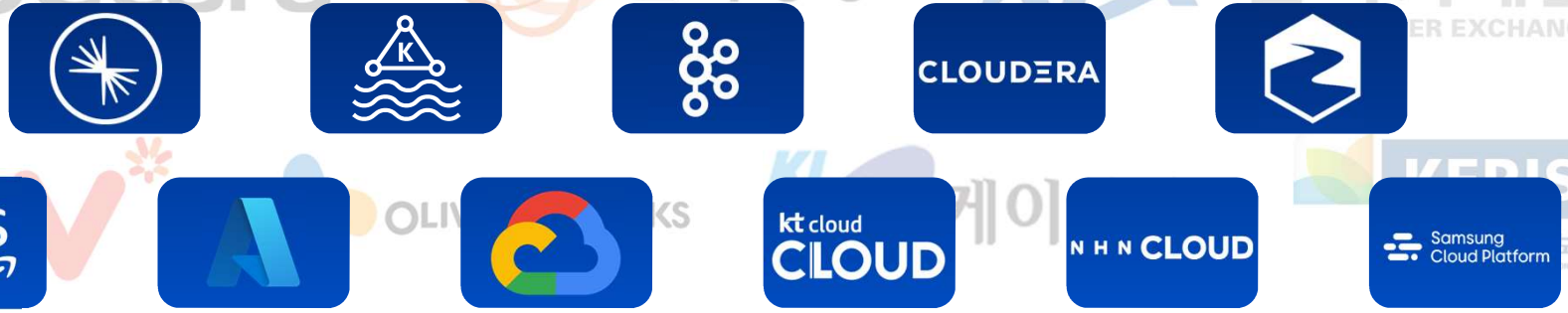
Unlock the full potential of your Kafka

Kafka 관리 운영 솔루션 | 토픽 생성 및 메타데이터 관리 | 스마트 파티션 리밸런싱 | 직관적인 파티션 관리

RBAC | 데이터 미러링 | 커넥트/커넥터 관리 | 개인화된 알림 | SCRAM / ACL 관리

컨슈머 랙 | CMPS | 노드 메트릭 모니터링 | 감사로그 | 상신/결재

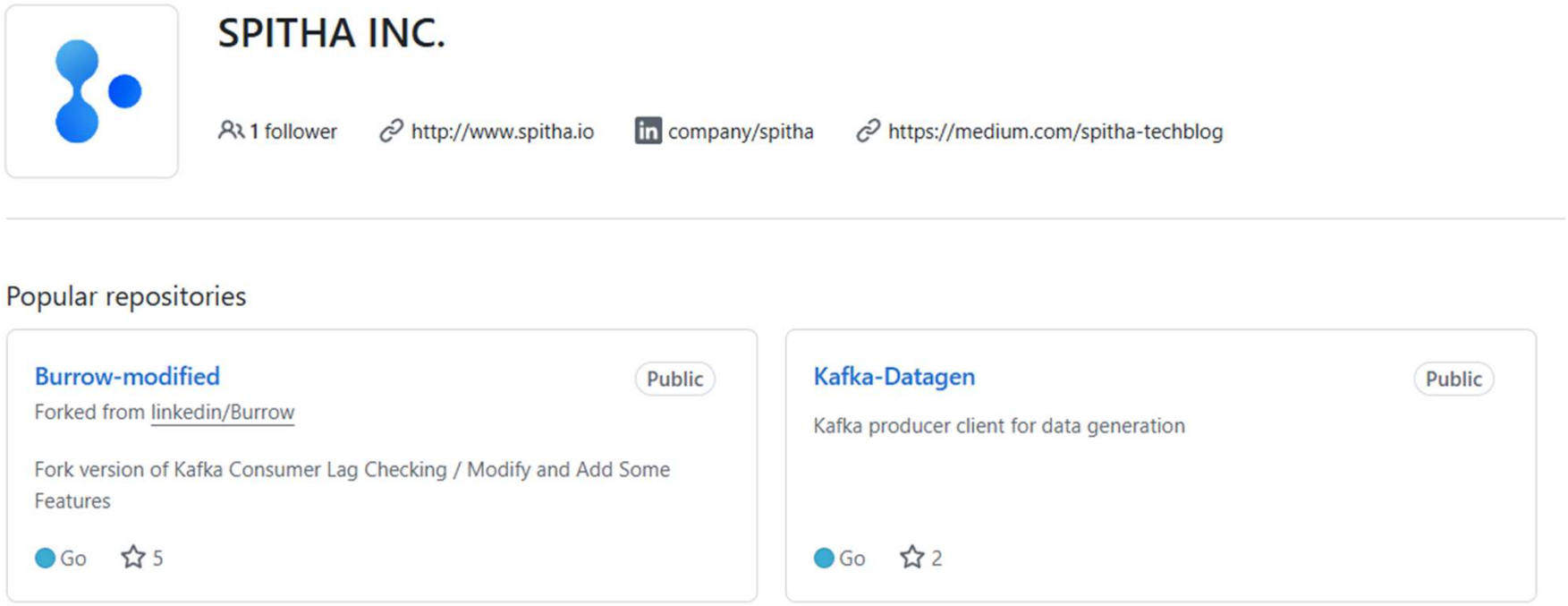




유용한 툴

SPITHA
Felice

- https://github.com/spithainc



Q&A

- 감사합니다!