광고 클릭 이벤트 집계

광고의 효율성을 측정하기 위한 집계 서비스를 설계하자

▼ 1️⃣ 설계 범위 확정

질문 거리

- 입력 데이터
- 데이터 양
- 중요 질의
- 지연 시간 요건

기능 요구사항

- 지난 M분 동안의 특정 광고 클릭 수 집계
- 매 분 가장 많이 클릭된 상위 N개 광고 아이디 반환
- 다양한 속성에 따른 집계 필터링
- 페이스북 / 구글 규모의 데이터 양

비기능 요구사항

- 집계 결과 정확성이 중요! RTB 및 광고 과금에 사용
- 지연 or 중복 데이터의 적절한 처리
- **견고성**: 부분적인 장애 감내
- 전체 처리 시간은 수 분 이내

개략적 추정

- DAU는 10억 명
- 각 사용자는 하루 평균 1개 광고 클릭
- 광고 클릭 QPS = 10,000
- 최대 광고 클릭 QPS = 50,000

- 광고 클릭 이벤트 하나 당 `0.1KB` 저장 용량 필요
  - **일일** 저장소 요구량 `0.1KB * 10억 = 100GB`
  - **월간** 저장소 요구량 `약 3TB`

▼ 2️⃣ 개략적 설계 (1)

질의 API

우리의 클라이언트는 최종 사용자가 아닌 데이터 엔지니어, 광고주, 제품 관리자 등이다 ...

API 1) 지난 M분 간 각 ad_id에 발생한 클릭 수 집계

GET /v1/ads/{:ad_id}/aggregated_count

request

필드명	설명	자료형
from	집계 시작 시간 (default: 1분 전)	long
to	집계 종료 시간 (default: 현재)	long
filter	필터링 전략 식별자	long

response

필드명	설명	자료형
ad_id	광고 식별자	string
count	집계된 클릭 횟수	long

API 2) 지난 M분 간 가장 많은 클릭이 발생한 상위 N개 ad_id 목록

GET /v1/ads/popular_ads

request

필드명	설명	자료형
count	상위 몇 개 광고를 반환할 것인가	integer
window	분 단위로 표현된 집계 윈도 크기	integer
filter	필터링 전략 식별자	long

response

필드명	설명	자료형
ad_ids	광고 식별자 목록	array

데이터 모델

원시 데이터 vs 집계 데이터 무엇을 어디에 저장할 것인가

원시 데이터

광고 식별자, 클릭 시간, 사용자, ip, 국가 등

집계 데이터

광고 식별자, 클릭 시간(분 단위), 클릭수

광고 필터링 위해 filter_id도 필요

🤔 필터링은 어떻게 할 것인가

필터 테이블 예시

filter_id	region	ip	user_id
0001	US	*	*
0002	*	123.1.2.3	0002

1) 무엇을 저장할 것인가?

	원시 데이터	집계 데이터
장점	- 원본 데이터 보관 가능 - 데이터 필터링 지원 - 데이터 재계산 지원	- 데이터 용량 절감 - 빠른 질의 성능
단점	- 막대한 데이터 용량 - 낮은 질의 성능	- 데이터 손실 가능성

→ 둘 다 저장하자

💡 문제 발생 시 **디버깅에 활용**할 수 있으므로 원시 데이터를 보관하는 게 좋다.
💡 오래된 원시 데이터는
냉동 저장소로 옮겨 비용을 절감하자.
💡 집계 결과 데이터는 질의 성능을 높이기 위해
튜닝하자.

2) 어디에 저장할 것인가?

데이터베이스 선택 기준

- 데이터는 어떤 모습인가?

    - 관계형, 문서, BLOB, ...
- 읽기 중심인가? 쓰기 중심인가? 둘 다인가?
- 트랜잭션 지원이 필요한가?
- 질의 과정에서 온라인 분석 처리(SUM, COUNT 등)이 필요한가?

원시 데이터

- 데이터 엔지니어가 질의할 수 있음
- 쓰기 중심 시스템

집계 데이터

- 시계열 데이터
- 읽기 / 쓰기 모두 많이 사용

→ 같은 유형의 데이터베이스 활용 가능. **쓰기 및 시간 범위 질의에 최적화된 카산드라**를 활용하자.

▼ 3️⃣ 개략적 설계 (2)

비동기 처리

메시지 큐를 도입하여 생산자-소비자 간 결합을 끊고 비동기 방식 프로세스를 활용하자

MQ1) 광고 클릭 이벤트 데이터 기록

광고 식별자	클릭 시간	사용자	ip	국가

원시 데이터베이스에 저장

MQ2) 집계 데이터 기록

정확하게 한 번 데이터 처리를 위해 집계 서비스가 집계 결과를 MQ에 입력 후 데이터베이스에 기록한다.

분 단위로 집계된 광고 클릭 수

광고 식별자	클릭 시간(분 단위)	클릭 수

분 단위로 집계한, 가장 많이 클릭한 상위 N개 광고

클릭 시간(분 단위)	광고 식별자

집계 서비스

맵리듀스 프레임워크

- 유향 비순환 그래프(DAG) 모델 사용
- 시스템을 `맵`, `집계`, `리듀스` 노드 등 작은 컴퓨팅 단위로 세분화

- 각 노드는 한 가지 작업만 처리
- 처리 결과를 다음 노드에 인계

맵 노드

- 데이터 출처에서 읽은 데이터를 필터링하고 변환
- 광고 수 집계의 경우 입력 데이터를 어느 집계 노드에 전달할 지 결정
- **입력 데이터를 정리하거나 정규화해야 하는 경우** 맵 노드가 필요하다!
  - 데이터가 생성되는 방식에 대한 제어권이 없는 경우 동일한 광고 식별자를 갖는 이벤트가 서로 다른 파티션에 입력될 수도 있다.

집계 노드

- 광고 별 클릭 이벤트 수를 매 분마다 메모리에서 집계

리듀스 노드

- 모든 집계 노드가 산출한 결과를 최종 결과로 축약

사용 예시

1) 클릭 이벤트 수 집계

- 맵 노드는 시스템에 입력되는 이벤트를 `ad_id % 3`을 기준으로 분배
- 분배한 결과를 각 집계 노드가 집계

2) 가장 많이 클릭된 상위 N개 광고 반환

- 맵 노드는 시스템에 입력되는 이벤트를 `ad_id % 3`을 기준으로 분배
- 3개의 각 집계 노드는 힙을 내부적으로 사용하여 상위 N개 광고를 효율적으로 식별
- 마지막 리듀스 노드는 집계 노드로부터 전달받은 `3 * N`개 광고 중 가장 많이 클릭된 N개 광고 필터링

3) 데이터 필터링

- 필터링 기준을 사전에 정의
  - 스타 스키마라고도 부를 수 있음 → 데이터 웨어하우스에서 쓰이는 기법
    - 이해하기 쉽고 구축이 간단
    - 결과를 미리 계산하여 데이터에 빠르게 접근 가능
    - 필터링 기준이 많아질수록 버킷과 레코드가 많이 생성되는 단점

▼ 4️⃣ 상세 설계

스트리밍 vs 일괄 처리

스트리밍) 실시간으로 집계된 결과를 생성
일괄 처리) 이력 데이터를 백업

람다 아키텍처 vs 카파 아키텍처

둘 다 스트리밍, 일괄 처리를 동시에 지원하는 아키텍처

- **람다:** 유지 관리해야 할 코드가 두 벌
- ⭐**카파:** 일괄 + 스트리밍 결합, 단일 스트림 처리 엔진으로 실시간 데이터 처리와 데이터 재처리 문제를 모두 해결

데이터 재계산

1. 재계산 서비스는 원시 데이터 저장소에서 데이터를 검색한다 (일괄 처리 프로세스)
2. 추출된 데이터는 전용 집계 서비스로 전송된다.
    - 실시간 데이터 처리 과정이 과거 데이터 재처리 프로세스와 간섭하는 일을 막기 위해
3. 집계 결과를 두 번째 메시지 큐에 전송하여 집계 결과 데이터베이스에 반영한다.

→ 데이터 집계 서비스를 재사용하지만 처리 대상 데이터는 원시 데이터를 직접 읽는다는 점이 다르다.

시간

광고가 클릭된 시간을 사용할 것이냐, 집계 서버가 이벤트를 처리한 시각을 사용할 것이냐

	이벤트 발생 시각	이벤트 처리 시각
장점	- 광고 클릭 시점은 클라이언트가 더 정확하게 안다.	- 서버 타임스탬프가 클라이언트보다 더 안정적이다.
단점	- 클라이언트가 생성한 타임스탬프에 의존하게 된다. (조작 가능성)	- 이벤트 처리 시간에 지연이 생기면 집계 결과가 부정확해진다.

→ **이벤트 발생 시각**을 사용하여 데이터 정확도를 높이고 워터마크 기술로 늦게 도착한 이벤트를 처리하자 !

집계 윈도

텀블링 윈도 vs 슬라이딩 윈도

- 텀블링 윈도
    - 시간을 같은 크기의 겹치지 않는 구간으로 분할

    - 매 분 발생한 클릭 이벤트 집계에 유리
- 슬라이딩 윈도
    - 데이터 스트림을 미끄러져 나아가며 같은 시간 구간 안에 있는 이벤트를 집계
    - 지난 M분 간 가장 많이 클릭된 상위 N개 광고 집계에 유리

워터마크

- 집계 윈도를 일정량 확장해서 집계 결과의 정확도를 높이는 방법
    - **데이터의 정확도는 높아지지만 전반적인 지연 시간은 늘어난다.**
- 비즈니스 요구사항에 따라 다르게 잡는다.
- 워터마크가 길수록 늦게 도착하는 이벤트를 포착할 수 있지만 이벤트 처리 시간이 늘어난다.
- 워터마크 기법으로도 시간이 한참 흐른 후에 시스템에 도달하는 이벤트는 처리할 수 없다.
    - 하루치 데이터 처리를 마감할 때 조정하도록 하자.

전달 보장

이벤트의 중복 처리를 어떻게 피할 것인가?
모든 이벤트의 처리를 어떻게 보장할 것인가?

어떻게 전달할 것인가

- 카프카는 최대 한 번, 최소 한 번, 정확히 한 번의 전달 방식을 지원한다.
- 약간의 중복이 괜찮다면 `최소 한 번`이 적절하지만,
- 집계 결과는 정확성과 무결성이 매우 중요하므로 `정확히 한 번` 방식을 권장한다.

중복 데이터 제거

- 클라이언트: 광고 사기/위험 제어 컴포넌트가 적합
- **서버**
    - HDFS나 S3 같은 외부 파일 저장소에 오프셋 저장
        - 오프셋 저장 직후 집계 서비스에 장애가 발생하면 해당 오프셋 이벤트를 다시 처리할 수 없다.
    - 분산 트랜잭션을 이용하여 다운스트림에서 집계 결과 수신 확인 응답을 받은 후에 오프셋 저장
        - `정확히 한 번` 처리 가능

시스템 규모 확장

메시지 큐

4장 분산 메시지 큐에서 다룬 내용

- 생산자: 생산자 인스턴스 수에는 제한이 없으므로 쉽게 확장 가능
- 소비자: 소비자 그룹 내 재조정 매커니즘으로 쉽게 규모 조정 가능
  - 소비자가 많으면 재조정 작업 시간이 길어짐
  - 많은 소비자를 추가하는 작업은 시스템 사용량이 많은 시간에 실행하여 영향을 최소화하자

브로커

- 해시 키
  - ad_id를 해시 키로 사용해서 같은 ad_id를 갖는 이벤트를 같은 파티션에 저장하자.
- 파티션 수
  - 파티션 수가 변하면 같은 ad_id를 갖는 이벤트가 다른 파티션에 기록될 수 있다.
  - 사전에 충분한 파티션을 확보하고
  - 파티션 수가 동적으로 늘어나는 일을 피하자
- 토픽의 물리적 샤딩
  - 장점: 시스템의 처리 대역폭이 높아진다. 단일 토픽에 대한 소비자 수가 줄면 재조정 시간이 단축된다.
  - 단점: 복잡성, 유지 관리 비용 증가

집계 서비스

본질적으로 맵리듀스 연산으로 구현

집계 서비스의 처리 대역폭을 높이기 위해 **1) ad_id마다 별도 처리 스레드**를 두거나 **2) 집계 서비스 노드를 자원 공급자에 배포해서 다중 프로세싱을 활용**할 수 있다. 1번은 쉽고 2번은 더 많이 쓰인다.

데이터베이스

카산드라는 안정 해시와 비슷한 방식으로 수평적 규모 확장을 지원한다.

핫스팟 문제

돈을 많이 쓴 광고는 더 많은 클릭이 발생하겠지 ...

→ 더 많은 집계 서비스 노드를 할당하여 완화한다.

1. 한 노드가 감당할 수 있는 이벤트 양을 초과한 경우 자원 관리자에게 추가 자원을 신청한다.
2. 자원 관리자는 추가 자원을 할당한다.
3. 원래 집계 서비스 노드는 각 서비스 노드가 적당한 양의 이벤트를 처리할 수 있도록 이벤트를 그룹으로 분할한다.
4. 집계가 끝난 결과는 다시 원래 집계 서비스 노드에 기록된다.

결함 내성

집계는 메모리에서 이루어지므로 집계 노드에 장애가 생기면 집계 결과도 손실된다 !

1. 업스트림 카프카 브로커에서 이벤트를 다시 받아온다.
2. 시스템 상태를 스냅샷으로 저장한다.
    - 업스트림 오프셋, 지난 M분간 가장 많이 클릭된 광고 같은 데이터가 시스템 상태에 해당한다.
3. 집계 서비스 노드에 장애 발생 시 해당 노드를 새 것으로 대체한 다음 마지막 스냅샷에서 데이터를 복구한다.

데이터 모니터링

지속적 모니터링

- **지연 시간**: 시스템의 중요 부분마다 timestamp 추적이 가능하도록 한다. 지연 시간 지표로 시각 차를 변환해서 모니터링 한다.
- **메시지 큐 크기**: 카프카를 사용하는 경우 `레코드 처리 지연 지표`를 대신 추적한다.
- **집계 노드의 시스템 자원**: CPU, 디스크, JVM, ...

조정

- 매일 각 파티션에 기록된 클릭 이벤트를 이벤트 발생 시각에 따라 정렬한 결과를 일괄 처리로 만든다.
- 실시간 집계 결과와 비교한다.
- 더 높은 정확도가 필요하면 더 작은 집계 윈도를 사용한다.
- 윈도 크기에 상관없이 일부 이벤트는 늦게 도착할 수 있으므로 배치 작업 결과가 실시간 집계 결과와 정확히 일치하지 않을 수 있다는 점을 유념하자.

▼ 5️⃣ 마무리

사고 프로세스를 설명하고 타협할 수 있는 선택지 간 장단점을 설명하는 능력이 중요. 갈수록 어려워지는듯

광고 클릭 이벤트 집계 시스템은 전형적인 빅데이터 처리 시스템이다.

- 맵리듀스 패러다임을 통해 광고 이벤트를 집계하는 방안
- 규모 확장 방안
- 핫스팟 문제 해결
- 시스템 모니터링
- 데이터 조정
- 결함 내성