06. 키-값 저장소 설계

문제 이해 및 설계 범위 확정

- 키-값 쌍의 크기는 10kb 이하
- 큰 데이터를 저장할 수 있어야
- 높은 가용성을 제공해야
  - 장애가 있어도 빨리 응답해야
- 높은 규모 확장성을 제공해야
  - 트래픽 양에 따라 자동적으로 서버 증설/삭제가 되어야
- 데이터 일관성 수준은 조정이 가능해야
- 응답 지연시간이 짧아야

단일 서버 키-값 저장소

- 모든 데이터를 메모리에 해시 테이블로 저장하는 것은 불가능
  - 해결책
    - 데이터 압축
    - 자주 쓰이는 데이터만 메모리에 두고, 나머지는 디스크에 저장
- 한 대 서버로 부족한 때가 곧 찾아옴
- 많은 데이터를 저장하려면 분산 키-값 저장소가 필요

분산 키-값 저장소

분산 시스템 설계시에는 CAP 정리를 이해해야

CAP 정리

- 데이터 일관성(consistency)
  - 분산 시스템에 접속하는 모든 클라이언트는 어떤 노드에 접속하느냐에 관계없이 언제나 같은 데이터를 보아야
- 가용성(availability)
  - 일부 노드에 장애가 발생해도 항상 응답을 받을 수 있어야
- 파티션 감내(partition tolerance)
  - 네트워크에 파티션(두 노드 사이에 통신 장애가 발생)이 생기더라도 시스템은 계속 동작해야
- 데이터 일관성, 가용성, 파티션 감내 세 가지를 동시에 만족하는 분산 시스템 설계는 불가능
- CAP 정리: 두 가지를 충족하려면 나머지 하나는 반드시 희생되어야

데이터 파티션

- 대규모 애플리케이션은 전체 데이터를 한대 서버에 넣는 것이 불가능.
  - 작은 파티션들로 분할하여 여러 서버에 저장해야
- 고려해야 할 문제
  - 데이터를 여러 서버에 고르게 분산할 수 있는가
  - 노드가 추가/삭제될 때 데이터의 이동을 최소화할 수 있는가
- 문제 해결에 안정 해시가 적절한 기술

데이터 다중화

- 높은 가용성, 안정성 확보를 위해서는 데이터를 여러 서버에 비동기적으로 다중화해야

데이터 일관성

- 여러 노드에 다중화된 데이터는 적절히 동기화 되어야
- 정족수 합의 프로토콜을 사용하면, 읽기/쓰기 연산 모두에 일관성 보장가능
- N: 사본 개수
- W: 쓰기 연산에 대한 정족수
- R: 읽기 연산에 대한 정족수
    - W=1 또는 R=1이라면 응답속도가 빠를 것.
    - R=1, W=N: 빠른 읽기 연산에 최적화
    - W=1, R=N: 빠른 쓰기 연산에 최적화
    - W+R>N: 강한 일관성이 보장
    - W+R<=N: 강한 일관성이 보장되지 않음

일관성 모델

- 강한 일관성: 모든 읽기 연산은 가장 최근에 갱신된 결과를 반환
  - 달성 방법: 모든 사본에 현재 쓰기 연산의 결과가 반영될 때까지, 해당 데이터에 대한 읽기/쓰기
    금지
    - 고가용성 시스템에는 부적합. 새로운 요청의 처리가 중단되기 때문
- 약한 일관성: 읽기 연산은 가장 최근 데이터를 반환하지 못할 수 있다
  - 최종 일관성: 약한 일광선의 한 형태. 갱신 결과가 결국에는 모든 사본에 반영
    - 쓰기 연산이 병렬로 발생하면, 저장된 값의 일관성이 깨질 수 있음. 이 문제는 클라이언트가 해결해야

비 일관성 해소 기법: 데이터 버저닝

- 버저닝: 데이터를 변경할 때마다, 해당 데이터의 새로운 버전을 만드는 것
- 벡터 시계: 버전 충돌시, 충돌을 발견하고 자동으로 문제를 해결
    - [서버, 버전]의 순서쌍을 데이터에 매단다.
    - 단점
        - 충돌 감지/해소 로직이 클라이언트에 들어가야 함. 구현이 복잡해짐
        - [서버,버전] 순서쌍 개수가 굉장히 빨리 증가

장애 감지

- 분산 시스템에서는, 두 대 이상의 서버가 똑같이 서버A의 장애를 보고해야, 장애가 발생했다고 간주
    - 멀티캐스팅이, 서버 장애를 감지하는 가장 손쉬운 방법.
        - 하지만, 서버가 많으면 비효율적. 분산형 장애감지 솔루션(예: 가십 프로토콜) 채택이 보다 효율적.

일시적 장애 처리

- 장애를 감지한 시스템은, 가용성 보장을 위해 필요한 조치를 해야
  - 엄격한 정족수: 읽기와 쓰기 연산을 금지해야
  - 느슨한 정족수: 정족수 조건을 완화하여 가용성을 높임. W와 R 선정시, 장애 상태인 서버는 무시
- 임시 위탁
  - 장애 상태인 서버로 가는 요청은 다른 서버가 임시로 처리.
  - 그동안 발생한 변경 사항은 해당 서버 복구 시 일괄 반영하여 데이터 일관성 보존
  - 이를 위해, 임시로 쓰기 연산을 처리한 서버에는 그에 관한 단서를 남겨둠

영구적 장애 처리

- 반-엔트로피 프로토콜: 영구적인 노드의 장애상태 처리를 위함
  - 사본들을 비교하여 최신 버전으로 갱신
- 머클 트리
  - 각 노드에 그 자식 노드들에 보관된 값의 해시, 또는 자식 노드들의 레이블로부터 계산도니 해시값을 레이블로 붙여두는 트리
  - 대규모 자료구조의 내용을 효과적이고 보안상 안전한 방법으로 검증

쓰기 경로

0. 클라이언트의 쓰기 요청
1. 쓰기 요청이 커밋 로그 파일에 기록
2. 데이터가 메모리 캐시에 기록
3. 메모리 캐시가 가득차거나 임계치에 도달하면, 데이터는 디스크에 있는 SSTable에 기록

읽기 경로

0. 읽기 요청
1. 데이터가 메모리에 있는지 검사. 없으면 2로
2. 블룸 필터 검사
3. 블룸 필터를 통해 어떤 SSTable에 키가 보관되어 있는지 확인
4. SSTable에서 데이터 가져옴
5. 데이터를 클라이언트에 반환