대용량 시스템 설계 기초

7.분산 시스템을 위한 유일 ID생성기

@OolongTea620
김혜린

2024.03.30

들어가기 전에

이런 내용이겠지..?

👉분산 시스템에서도 동작하는 유일 ID 생성 시스템을 구축하는 방법?

해당 챕터의 목적

Auto_Increasement 속성이 설정된
관계형 데이터 베이스의 기본키를 사용하면 되지 않을까?...🤔

하지만...

Auto_Increasement 만으로는 부족하다😂

- 분산 환경에서 사용이 어려움 -> 특기 DB저장 환경이 분산되어 있는 경우
- 데이터베이스 서버 한대로 요구 감당이 어려움
- 여러 데이터베이스 서버를 사용하는 경우, 지연 시간을 낮추기가 무척 어려움

1단계 - 문제 이해 및 설계 범위 확정

책에서 가정하는 시스템 요구사항

ID는 유일해야
ID는 숫자로만 구성이 되어야 하고
ID는 64비트로 표현될 수 있는 값이여야한다.
ID는 발급 날짜에 따라 정렬 가능해야
초당 10,000개 정도의 ID를 만들 수 있어야

2단계 - 개략적인 설계안 제시 및 동의 구하기

분산시스템에서 유일성이 보장되는 ID를 만드는 방법

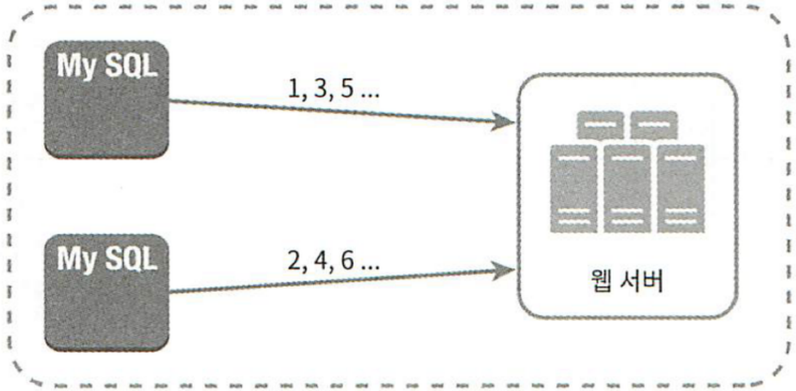
- 다중 마스터 복제 (multi-master replication)
- UUID(Universally Unique Identifier)
- 티켓 서버(ticket server)
- 트위터 스노플레이크(twitter snowflake) 접근법

등..

다중 마스터 복제(multi-master replication)

- 데이터베이스의 auto_increasement 활용
- 다음 ID의 값을 구할 때, 1만큼 증가가 아닌, k만큼 증가 시킨다

(k : 현재 사용 중인 데이터베이스 서버의 수)



다중 마스터 복제(multi-master replication)
치명적 단점

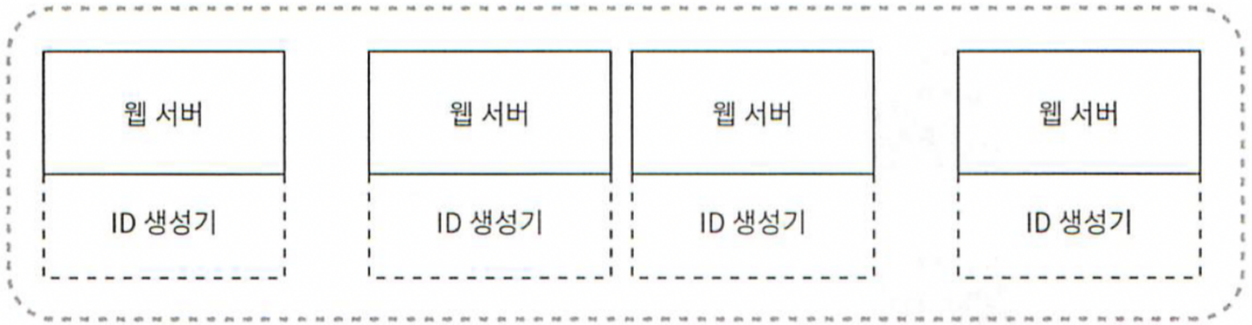
여러 데이터 센터에 걸쳐 규모를 늘리기 어렵다.
ID의 유일성은 보장되겠지만 그 값이 시간 흐름에 맞추어 커지도록 보장할 수는 없다.
서버를 추가하거나 삭제할 때도 잘 동작하도록 만들기 어렵다. (k의 수는 고정되어야)

UUID(Universally Unique Identifier)

컴퓨터 시스템에 저장되는 정보를 유일하게 식별하기 위한 128비트 짜리 수

장점

- UUID를 만들기 간단
- 서버 사이의 조율이 필요 없다 - 동기화 이슈
- 각 서버가 자신이 쓸 ID를 알아서 만드는 구조 - 쉬운 구조 확장

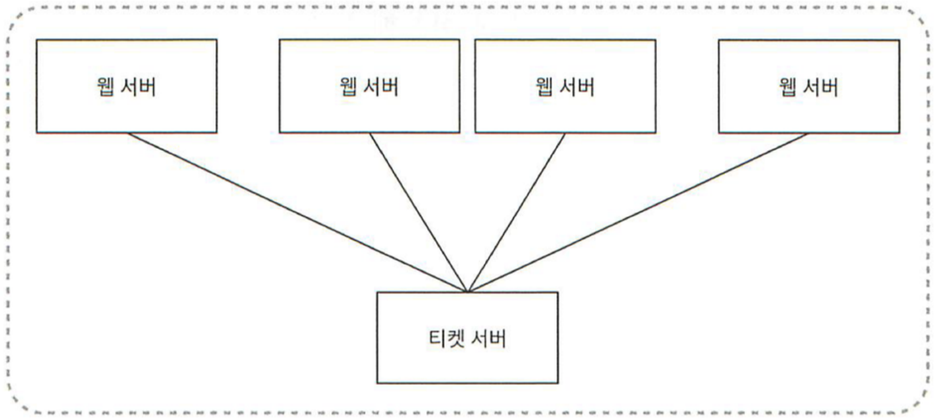


UUID(Universally Unique Identifier)

단점

(내 생각에는 이번 장 가정된 요구 사항을 조금 넘어서 그런 듯 하다)
- ID가 128비트로 길다.(이번 장에서 다루는 문제의 요구사항은 64비트)
- ID를 시간순으로 정렬할 수 없다.
- ID에 숫자(numeric) 아닌 값이 포함 될 수 있다.

티켓 서버(ticket server)



플리커(Flickr)에서 분산 기본 키(distributed primary key)를 만들어 내기 위해 이 기술을 이용

auto_increasement 기능을 갖춘
데이터베이스 서버, 티켓 서버를
중앙 집중형으로 하나만 사용하는 것

장점

일성이 보장되는 숫자로만 구성된 ID를 쉽게 만들 수 있다.
구현하기 쉽고, 중소 규모 애플리케이션에 적합하다.

티켓 서버(ticket server)

단점

티켓 서버가 SPOF(Single-Point-of-Failure)가 된다
이 서버에 장애가 발생하면, 해당 서버를 이용하는 모든 시스템이 영향을 받는다.

이 이슈를 피하려면 티켓 서버를 여러 대 준비해야 한다.
하지만 그렇게하면 데이터 동기화 같은 새로운 문제가 발생할 것이다.

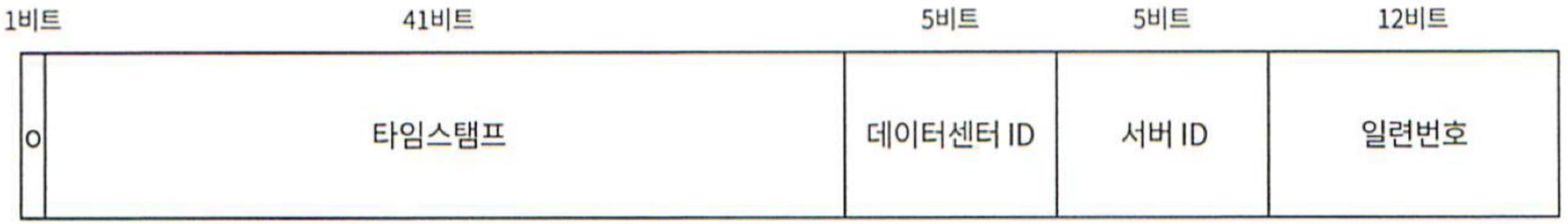
트위터 스노플레이크(twitter snowflake) 접근법

트위터에서 사용하는 기술
책에서는 이 방법이 가장 요구사항에 적합한 케이스라고 설명 중
바로 ID를 생성하는 대신, 분할정복 적용

분할정복(divide and conquer)이란?

이 접근법은 분할정복을 적용
생성해야 하는 ID 구조를 여러 절로 분할

분할정복(divide and conquer)이란?



- 사인(sign)비트 : 1비트 할당. 음수와 양수를 구별하는데 사용 할 수 있음
- 타임스템프(timestamp) : 41비트 할당. 기원 시각(epoch)이후로 몇 밀리초가 경과했는지를 나타내는 값.
- 데이터센터 ID : 5비트 할당. 따라서 2^5 = 32개 데이터센터를 사용할 수 있음
- 서버 ID : 5비트 할당. 따라서 데이터 센터 당 32개 서버를 사용할 수 있음
- 일련번호 : 12비트 할당. 각 서버에는 ID를 생성할 때마다 일련번호를 1씩 증가시킨다.
  이 값은 1밀리초가 경과할 때 마다 0으로 초기화(reset) 된다.

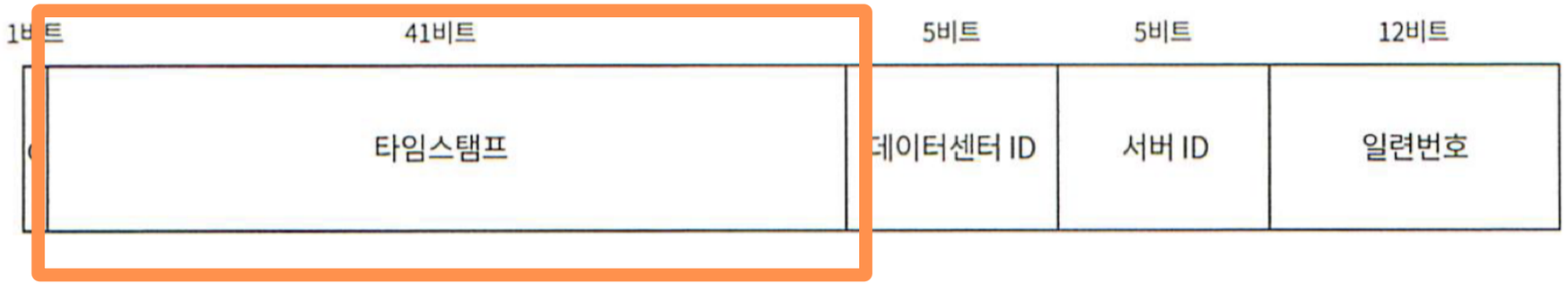
분할정복(divide and conquer)이란?

책에서는 Twitter SnowFlake 전략을 채택

- 사인(sign)비트 : 1비트 할당. 음수와 양수를 구별하는데 사용 할 수 있음
- 타임스템프(timestamp) : 41비트 할당. 기원 시각(epoch)이후로 몇 밀리초가 경과했는지를 나타내는 값.
- 데이터센터 ID : 5비트 할당. 따라서 2^5 = 32개 데이터센터를 사용할 수 있음
- 서버 ID : 5비트 할당. 따라서 데이터 센터 당 32개 서버를 사용할 수 있음
- 일련번호 : 12비트 할당. 각 서버에는 ID를 생성할 때마다 일련번호를 1씩 증가시킨다.
  이 값은 1밀리초가 경과할 때 마다 0으로 초기화(reset) 된다.

3단계 - 상세 설계 : 타임스탬프

트위터 스노우플레이크 적용을 했다고 할 때,
가장 중요한 41비트를 차지하고 있음
시간 순으로 정렬 가능(타임스템프는 시간에 따라 점점 큰 값을 가지게 됨)



3단계 - 상세 설계 : 타임스탬프

41비트로 표현될 수 있는 타임스탬프의 최대값
== 2^41 - 1 = 2199023255551 밀리초
~ 대략 69년 동안 정상 동작

69년이 지나면 기원 시각을 바꾸거나
ID 체계를 다른 것으로 이전 해야한다.

3단계 - 상세 설계 : 일련 번호

12비트, 2^12 = 4096개의 값을 가질 수 있음
어떤 서버가 같은 밀리초 동안 하나 이상의 ID를 만들어 낸 경우에만 0보다 큰 값을 갖게 된다

3단계 - 결론

추가적으로 알아보면 좋은 것들

시계 동기화

- 하나의 서버가 여러 코어에서 동작하는 경우라면?(NTP 프로토콜)

각 절의 길이 최적화(section)

- 동시성(concurrency)이 낮고, 수명이 긴 어플리케이션이라면?
  >> 절의 길이는 줄이고, 타임스탬프 절의 길이를 늘려볼까?

4단계 - 결론

추가적으로 알아보면 좋은 것들

시계 동기화

- 하나의 서버가 여러 코어에서 동작하는 경우라면?(NTP 프로토콜)

각 절의 길이 최적화(section)

- 동시성(concurrency)이 낮고, 수명이 긴 어플리케이션이라면?
  >> 절의 길이는 줄이고, 타임스탬프 절의 길이를 늘려볼까?

추가 - 시계 동기화와 NTP(Network Time Protocol)

인터넷에서 라우터 및 기타 하드웨어 디바이스의 클럭을 동기화하는데 사용되는 프로토콜

인터넷을 통해서 컴퓨터 시간을
최상위 동기 클럭원(Master Clcok)에 동기시키는 프로토콜

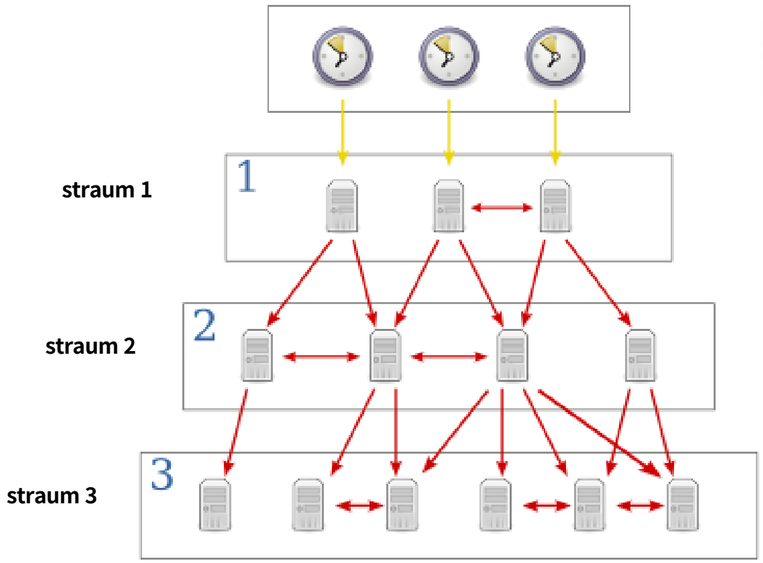
네트워크 상에 분산 된 시간 서버들로부터 클라이언트(호스트, 라우터)등의 동기화
PC의 Local이 아닌, Online을 통해서 시간을 가져오는 것.

- 0.001초 단위 까지 동기화 가능
- UTC 기반으로 이루어짐

출처: 위키백과

추가 - 시계 동기화와 NTP(Network Time Protocol)

인터넷에서 라우터 및 기타 하드웨어 디바이스의 클럭을 동기화하는데 사용되는 프로토콜



MAC Header	IP Header	UDP Header	NTP Mesaage

LI	VN	Mode	Strat	Poll	Prec
Root Delay					
Root Dispersion					
Reference Identifier					
Reference Timestamp					
Originate Timestamp					
Receive Timestamp					
Transmit Timestamp					
확장 필드 (선택 옵션)					
Key/Algorithm Identifier					
Message Hash (64 or 128)					

UDP 기반
서버에서 브로드캐스트 방식으로 시간의 정보가 전달
이를 호스트에 받아서 시간을 설정

서버와 클라이언트 계층을 표현하기 위해
Straum(mean : 암석층, 사회적 계층)이라는 용어를 사용

추가 - 시계 동기화와 NTP(Network Time Protocol)

인터넷에서 라우터 및 기타 하드웨어 디바이스의 클럭을 동기화하는데 사용되는 프로토콜

장점
데이터의 손실 방지
로그에 대한 분석 용이
예약된 작업을 정상적으로 가능하게 한다.

단점
외부 서버를 통해서 시간을 동기화하는 부분으로 인한 보안상의 취약
(해결책으로 별도의 Time Server를 이용하기도)

END