1109 NoSQL 심화

수평적 확장의 의미.

서버의 대수를 늘려 처리 능력을 향상시키는 것.

샤딩(Sharding)/파티셔닝(Partitioning)

테이블 단위로 분리하는 수직적 샤딩과, 테이블 자체를 분리하는 수평적 샤딩이 존재한다.

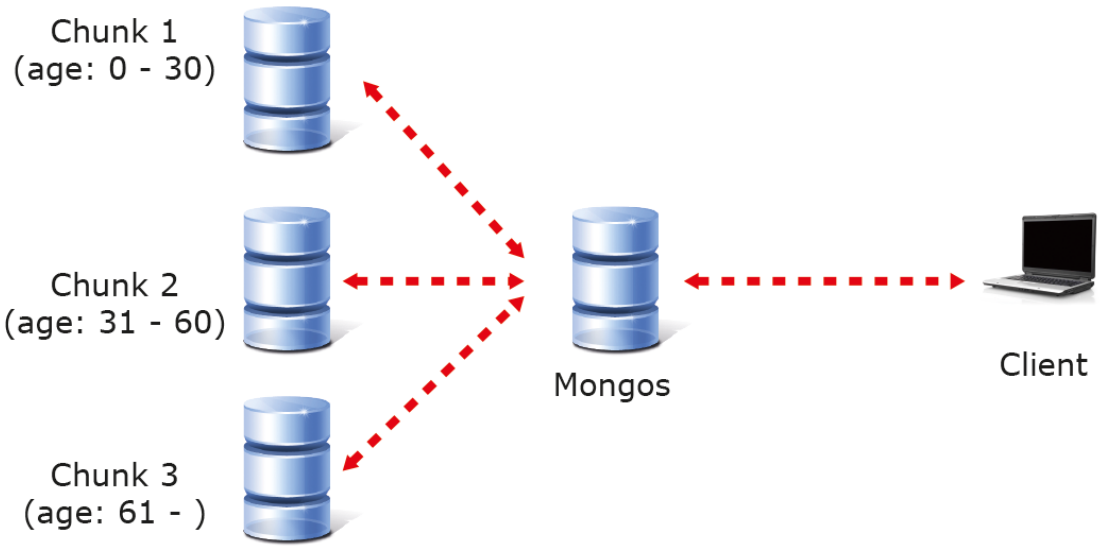
▼ 두가지의 차이가 무엇인가?

관계형 데이터베이스에서의 표현 파티셔닝

NoSQL에서의 표현을 샤딩으로 이해

샤드키(Shard Key)에 대한 의미

클러스터의 샤드들 간에 컬렉션의 도큐먼트를 어떻게 분산할 것인가를 결정하는 것



위의 사진은 age를 샤드키로 Chunk를 나누게 된다.

데이터의 정렬,조회,삭제 등과 같은 작업시엔 Age를 기준으로 나눠진 chunk별로 데이터를 정렬할 수 있기 때문에 데이터 전체를 탐색할 필요가 없어진다.

▼ mongos, chunk

mongos : 여러 chunk들을 이어주는 query routing 기능을 수행

chunk : shard Key에 따라 나눠진 클러스터(저장공간)들을 의미.

어떤 경우에 쓰는가?

읽기(read)처리를 자주하지만, 데이터를 자주 변경하지 않는 경우 (즉, 한번의 변경으로 수십 개의 문서를 수정 할 필요가 없는 경우)

→데이터를 단순히 수집하는 목적으로만으로도 큰 성과가 있는 분야에 유리(빅데이터,크롤링)

다수의 요구를 동시 병행으로 처리할 필요가 있지만 개개의 처리는 비교적 단순한 경우에 적합하다.

→데이터의 변경이 적은 웹서버에 적합

한줄평

결과적으로 데이터의 양으로 승부를 보겠다는 마인드.

관계형 데이터베이스가 CRUD모두를 담당하며 작동하게 된다면 NoSQL의 경우는 데이터의 저장 수집 등에 편중되어있다.

Key-Value Store

NoSQL 데이터베이스 중 가장 간단한 방식

키와 값을 한 쌍으로 저장하며, 더이상의 기능이 없고, 키 기반으로 값을 저장하는 방식이다.

제품	C(탄소)함유량	H(수소)함유량	O(산소)함유량
ProductA	0.2%	0.1%	0.9%
ProductB	0.3%	0.2%	0.6%

기존 관계형 데이터베이스의 모습

Key	Value
ProductA	{C=0.2% H=0.1% O=0.9%}
ProductB	{C=0.3% H=0.2% o=0.6%}

Key-Value Store의 모습

찾고자 하는 검색키가 들어오면 주소계산을 통하여 발생한 인덱스를 기초하여 값을 찾아낸다.

확장성의 방식

읽기가 많이 일어나는 구조

→ 마스터-슬레이브 복제

읽기/쓰기를 모두 수행하는 마스터서버를 기점으로 읽기요청에만 응답하는 슬레이브 서버로 구성

단점) 마스터서버가 고장날시에 슬레이브중 하나를 마스터로 세워야함.

쓰기가 많이 일어나는 구조

→ 마스터 없는 복제

데이터를 모든 서버에서 읽고 쓰는 작업을 처리하므로 각각의 서버에서 처리한 내용을 공유하게 된다.

속도를 증대시키는 방법

빠른 연산을 위해 메모리에 데이터를 관리하는 방식을 사용하게된다.

조회작업의 경우 디스크에서 데이터를 가져오지만 같은 key값에 대해 조회했던 이력을 통해 더 빠른 결과를 전송

입력값의 경우 - 데이터가 입력됬다는 사실을 사용자에게 회신한뒤, 실제적인 데이터를 DB에 입력하는 방식으로 빠른 결과를 전송

발생할 수 있는 문제점.

해시 충돌(Hash Collision)

해시 함수가 서로 다른 두개의 입력값에 대해 동일한 출력값을 내는 상황

→ 해시함수의 경우는 아무리 잘 설계가 되어도 잠재적인 충돌 가능성을 안고 있다.

Key-Value Store의 경우

분산 서버에 데이터를 저장하고 검색하기 위해 해시 함수에서 키를 기반으로 값의 위치를 계산하는 로직의 성능이 중요하다.

이때 같은 위치의 값에 접근하게 되면 충돌이 발생하게 되며, 이런 충돌을 감지하고 이를 해결하기 위한 준비가 필요

해결방안

삽입에 대한 충돌 감지 및 처리

새로운 위치를 결정하기 위해 보조 해시를 사용하거나 선형 검색을 수행.

어느 경우라도 DB의 반응성을 감소시키게 된다.

해결방안 : 파티션에서 다음 이용 가능한 슬롯에 값을 저장하는것.

검색할 때 충돌을 감지하고 데이터를 처리

검색 시 계산된 위치에 유지키를 검사하게 되는데, 키가 정확하지 않을경우 삽입시의 다음 슬롯과 동일한 전략을 통해 데이터를 조사하게 된다.

사용 사례

세션 스토어 : 사용자가 로그인을 할시에 세션을 시작하고, 세션을 초과 혹은 로그아웃시까지 계속 유지되게 되고, 세션 관련 데이터를 모두 DB에 저장하게 된다.

이때 Key-Value Store의 Key값은 해당 사용자의 세션을 의미하게된다.

장바구니 : 수백만명의 서비스를 제공하는 동시에 대량의 데이터처리와 매우 많은 양의 상태 변경 작업을 처리할 수 있게 된다.

빅데이터 : 인터넷의 넘쳐나는 데이터를 저장하고 검색하고
관리

참조

https://12bme.tistory.com/323

https://ojava.tistory.com/130

https://ryufree.tistory.com/208

https://ko.wikipedia.org/wiki/데이터베이스_분할

https://bcho.tistory.com/742

https://asfirstalways.tistory.com/66

https://devuna.tistory.com/73

https://coding-start.tistory.com/275

https://cinema4dr12.tistory.com/508