15주차 발표_민서연

NoSQL
: MongoDB

01 NoSQL

Not Only SQL

비관계형 데이터베이스

관계형 데이터 모델을 지양하고 대량의 분산된 데이터를 조회하는데 특화되어 있음
스키마 없이 사용 가능하거나 느슨한 스키마 제공

단순 검색 및 추가 작업에 있어서 매우 최적화된 키 저장 기법을 사용하기 때문에
응답속도나 처리 효율에 있어서 뛰어난 성능을 가지고 있음

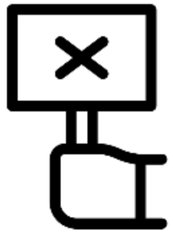
01 NoSQL

RDB와 차이점

- 데이터 간의 관계를 정의하지 않는다.
- 대용량의 데이터를 저장한다.
- 분산형 구조다.
- 고정되지 않은 테이블 스키마를 가진다.
- open source로 제공한다.

※ RDB : Relational Database
key와 value 들의 간단한 관계를 테이블화 시킨
매우 간단한 원칙의 전산정보 데이터베이스

01 NoSQL



단점

ACID 트랜젝션을 지원하지 않는다.

= 데이터 일관성이 항상 보장되지 않는다.

A: Atomicity (원자성)
C: Consistency (일관성)
I : Isolation (격리성)
D: Durability (지속성)

01 NoSQL

종류

Key-Value Database

Key와 Value로 이루어져 있음.
저장과 조회라는 원칙에 충실
Redis, AWS Dynamo DB, Oracle Berly,....

Column Family Database

대용량의 데이터 읽기와 쓰기의 성능, 고가용성을 위해 설계됨.
RDB와 같이 Column과 Row를 사용해서 스키마 정의
원하는 시점에 컬럼을 추가할 수 있음
Hbase, Cassandra, GCP,.....

Document Database

Key Value Type을 사용하지만 Document 타입으로 저장됨 (JSON, XML)
스키마를 별도로 저장하지 않으며 document가 스키마가 됨
MongoDB, CouchDB, Couchbase,.....

Graph Database

Graph 이론을 활용한 것이며 노드들과 관계를 구성된 개념.
간단하고 직관적인 데이터 모델을 가짐
Neo4j, Titan, AllegroGrapth ,.....

02 MongoDB

NoSQL의 Document Database

ACID 대신 BASE를 택해서 성능과 가용성을 우선시함

BASE

Basically Avaliable
: 기본적으로 언제든지 사용할 수 있음
Soft state
: 외부의 개입이 없어도 정보가 변경될 수 있음
Eventually consistent
: 일시적으로 일관적이지 않은 상태가 되어도 일정 시간 후 일관적인 상태가 되어야 함.

02 MongoDB

구성요소

Database

각 파일 시스템에 따라
여러 파일들이 저장되는 곳.
Collection들의 집합

Collection

Document들이 합쳐져서
하나의 Collection을 이루고 있음

Document

모든 Document는 id 필드를
기본값으로 반드시 가지고 있어야 하며
Document를 구별하는 역할을 함

02 MongoDB

1) Document 입력

insertOne: 한개의 document 생성

```JSON
db.books.insertOne(
  {
    title : "Snow White",
    autor : "someone",
    genre : "fairy tale"
  }
)
```

Collection

field : value → document

02 MongoDB

1) Document 입력

insertOne: 한개의 document 생성

```JSON
db.books.insertOne(
  {
    title : "Snow White",
    autor : "someone",
    genre : "fairy tale"
  }
)
```

```
INSERT INTO bookes(title, author, genre)
VALUES("Snow White", "someone", "fairy tale")
```

mongoDB insert와 SQL insert 문법 비교

02 MongoDB

1) Document 입력

insertMany: list of document 생성

```
db.bookes.insertMany(
  [
    { title : "Snow White", author : "someone", genre : "fairy tale" },
    { title : "Cosmos", author : "Carl Sagan", genre : "science" }
  ]
)
```

02 MongoDB

2) Document 읽기

findOne: 매칭되는 document 검색

```
db.books.find({ title : "Snow White" }) // mongoDB 검색

SELECT * FROM books WHERE title = "Snow White" // SQL 검색
```

02 MongoDB

2) Document 읽기

비교 문법

$eq	=
$gt	>
$gte	>=
$in	
$lt	<
$lte	<=
$ne	!=

02 MongoDB

3) Document 수정

updateOne: 매칭되는 한개의 document 업데이트

updateMany: 매칭되는 list of document 업데이트

```
db.books.updateOne( Collection
  { genre : "fairy tale" },update filter
  { $set : { title : "Cinderella" } } update action
)
```

02 MongoDB

3) Document 수정

updateOne: 매칭되는 한개의 document 업데이트

updateMany: 매칭되는 list of document 업데이트

```
db.books.updateOne(
  { genre : "fairy tale" },
  { $set : { title : "Cinderella" } }
)
```

```
UPDATE books SET title = "Cinderella" WHERE genre = "fairy tale"
```

mongoDB update와 SQL update문법 비교

02 MongoDB

3) Document 삭제

removeOne: 매칭되는 한개의 document 삭제

removeMany: 매칭되는 list of document 삭제

```
db.books.deleteMany(
  { author : "Anne Anderson" }
)
```

Collection

delete filter

02 MongoDB

3) Document 삭제

removeOne: 매칭되는 한개의 document 삭제

removeMany: 매칭되는 list of document 삭제

```
db.books.deleteMany(
  { author : "Anne Anderson" }
)
```

```
DELETE FROM books WHERE author = "Anne Anderson"
```

mongoDB delete와 SQL delete문법 비교

04 출처

https://dev-jwblog.tistory.com/149
https://azderica.github.io/00-db-nosql/
https://kciter.so/posts/about-mongodb
https://velog.io/@baek1008/DB-MongoDB-Nosql%EC%9D%84-%ED%99%9C%EC%9A%A9%ED%95%B4%EC%84%9C-DB%EB%A5%BC-%EC%82%AC%EC%9A%A9%ED%95%B4%EB%B3%B4%EC%9E%90
https://www.fun-coding.org/post/mongodb_basic4.html#gsc.tab=0