

Slow Query 개선

참고 : <https://rockset.com/blog/handling-slow-queries-in-mongodb-part-2-solutions/>

1. 사용자 설정 index 사용 (Collection Scan 감소 전략)

대부분의 거대 Production Database 에서는 Collection Scan 이 이루어지기 매우 어렵거나 불가능에 가깝다.

관계형 데이터베이스와 마찬가지로 MongoDB 도 인덱스를 사용하여 쿼리 속도를 높일 수 있다.

```
> db.collection.createIndex( <key      index      specification>, <options> )
```

- Single Field Index
- 참고
- **Indexes** are used to index single field in a document
- **Compound Field Indexes** are used to index multiple fields in a document
- **Multikey Indexes** are used to index the content stored in arrays
- **Geospatial Indexes** are used to efficiently index geospatial coordinate data
- **Text Indexes** are used to efficiently index string content in a collection
- **Hashed Indexes** are used to index the hash values of specific fields to support hash-based sharding

대부분 상황에서는 Hashed Index 를 _id 필드로 설정하기 마련이다. 하지만 사실 _id 필드는 생성된 시간과 관련이 있는 상태로 생성 된다.

5f49475943	42bf9a4e	a20b9e
timestamp	random value	counter

이 뜻은 무엇이냐면 ...!! 특정시간대에 몰려서 생긴 document 의 경우는 적절한 분배가 이루어 지지 않을 수 있다는 것... 주의하도록 하자

2. Covered Queries (전체 Document Scans 방지)

: <https://www.mongodb.com/docs/manual/core/query-optimization/#covered-query>

3. Avoiding Application-Level JOINS using Denormalization

4. Speed Up Queries and Perform Fast JOINS using External Indexes