Ch.5
Queries and aggregation

19-2 ybigta 엔지니어링팀 스터디
14기 안주영

Queries

- 몽고DB에서는 RDB에서 사용하는 SQL언어를 사용하는 것이 아니라 JSON-like query 언어를 사용한다

1. slug가 whell-barrow-9092인 상품을 찾는다.

```
db.products.findOne({'slug':'whell-barrow-9092’})
```

2. 해당 상품의 카테고리 정보를 가져온다.

```
db.categories.findOne({'_id':product['main_cat_id']})
```

3. 상품 리뷰를 불러온다.

```
db.reviews.find({'product_id':product['_id']})
```

4. 0~12번째 상품 리뷰를 불러온다.

```
db.reviews.find({'product_id':product['_id']}).skip(0).limit(12)
```

Queries

- findOne()과 find()의 차이점
  - findOne()은 도큐먼트를 리턴한다.하나의 도큐먼트를 얻고자 할 때 사용한다.
  - find()는 커서객체를 리턴한다. 여러개의 도큐먼트를 리턴 할 때 사용한다.
  - findOne()은 db.collection.find().limit(1)과 동일하다

Queries

- 정렬은 sort() 함수를 사용한다.
- 1은 오름차순 -1은 내림차순 정렬.
- 페이지를 나누기위해 Mongodb는 skip()과 limit() 옵션을 제공한다.
- 특정 키정보가 없는 도큐먼트만 가져오고 싶을때는 nil을 사용한다.

1. 0~12번째 상품 리뷰를 추천수가 많은 순서로 불러온다.

```
db.reviews.find({'product_id':product['_id']}).sort({helpful_votes:-1}).skip(0).limit(12)
```

2. parent_id정보가 없는 카테고리 가져오기

```
db.category.find({'parent_id’:nil})
```

Queries

- findOne()을 사용하면 도큐먼트를 결과값으로 받고, 결과가 없으면 아무것도 리턴하지 않는다.
- 필요하지 않은 필드까지 가져옴으로써 성능에 문제가 생길수 있다.
- 도큐먼트에서 가져올 필드를 제한하는 방법으로, 두번째 파라미터로 필요한 필드는 값을1 필요없는 필드는 값을0으로 지정해 준다.

1. _id정보만 가져오기

```
db.category.findOne({slug:'gardening-tools'},{_id:1})
```

2. _id정보만 제외하고 가져오기

```
db.category.findOne({slug:'gardening-tools'},{_id:0})
```

Queries

- RDBMS의 like와 같은 쿼리는 정규식을 이용한다

1. 이름이 ba로 시작하는 사용자 찾기

```
db.users.find({last_name:/^ba/})
```

Queries

- SQL의 비교연산자인 <=, >, >=; 의 경우
  - MongoDB에서는 analogous set of operators $lt, $lte, $gt, $gte 를 사용 할 수 있다
  - Search key를 반복하면 안된다

1. 0세에서 30세 사이의 유저를 조회 (틀린 조회방법)

```
db.users.find({age: {$gte: 0}, age: {$lte: 30})
```

2. 옳은 조회방법

```
db.users.find({age: {$gte: 0, $lte: 30})
```

Queries

- SQL의 비교연산자인 <=, >, >=; 의 경우
  - MongoDB에서는 analogous set of operators $lt, $lte, $gt, $gte 를 사용 할 수 있다
  - Search key를 반복하면 안된다

1. 0세에서 30세 사이의 유저를 조회 (틀린 조회방법)

```
db.users.find({age: {$gte: 0}, age: {$lte: 30})
```

2. 옳은 조회방법

```
db.users.find({age: {$gte: 0, $lte: 30})
```

- $in, $nin, $all, $ne, $not, $or 연산자도 사용 가능하다

Aggregation

- group함수를 이용해서 집계함수를 사용할 수 있다

1. 사용자별 리뷰 수와 리뷰 평점

```
db.reviews.group({
        key : {user_id : true},
        initial : {reviews : 0, votes : 0.0},
        reduce : function(doc, aggregator){
                aggregator.reviews +=1;
                aggregator.votes += doc.votes;
        },
        finalize : function(doc){
                doc.average_votes = doc.votes / doc.reviews;
        }
 })
```

Aggregation

- 대부분의 RDBMS는 합계, 평균, 편차와같은 내장 집계함수를 많이 제공하지만, Mongodb에는 구현될 때까지 group,map-reduce를 사용해야한다.
- group()은 최소한 3개의 파라미터를 받는다.
- 첫번째 파라미터 key는 데이터를 어떻게 그룹으로 묶을것인지를 지정한다.
- 두번째 파라미터는 리듀스(reduce funciton) 함수로, 결과 값에 대해 그룹으로 묶는 자바스크립트 함수다.

Aggregation

- group은 많은 경우 맵-리듀스보다 빠르기 때문에 좋은 선택일 수 있다.
- key : 그룹으로 나누는 기준이 되는 키를 표현한 도큐먼트. 복합키도 가능.
- keyf : 그룹을 위한 키를 계산을 통해 만들어야 하는경우 필요한 자바스크립트 함수. key를 지정하지 않으면 반드시 필요.
- initial : 집계 결과의 초기값으로 사용되는 도큐먼트.(필수)
- reduce : 집계 기능을 수행하는 함수.현재 도큐먼트와, 집계결과를 저장하는 도큐먼트 2개의 파라미터를 받는다.리턴할 필요가 없다.(필수)
- cond : 집계를 수행하기 위한 도큐먼트를 필터링 하는 쿼리 셀렉터.
- finalize : 결과값을 리턴하기전 각 도큐먼트에 적용되는 함수.

map-reduce

- 맵-리듀스는 group의 좀더 유연한 버전

```
1. 2010년 이후 월 판매액을 구하여 total컬렉션에 저장하기
map = function(){
        var shipping_month = this.purchase_date.getMonth() + '-' + this.purchase_date.getFullYear();
        var tmpItems = 0;
        this.line_items.forEach(function(item){
                tmpItems += item.quantity;
        });
                emit(shipping_month, {order_total : this.sub_total, items_total : tmpItems });
}

reduce = function(key, values){
        var tmpTotal = 0;
        var tmpItems = 0;
        tmpTotal += doc.order_total;
        tmpItems += doc.items_total;
        return({total : tmpTotal, items : tmpItems});
}

filter = {purchase_date : {$gte : new Date(2010, 0, 1)}}

db.orders.mapReduce(map, reduce, {query : filter, out : 'totals'});
```

map-reduce

- map : 각 도큐먼트에 적용하는 자바스크립트 함수. 집계에 사용 할 키와 값을 선택하기 위해 emit()을 의무적으로 호출 해야함.
- reduce : 키와 값의 리스트를 받는 자바스크립트 함수. 반드시 받은 값과 같은 구조의 값을 리턴 해야함.
- query : 매핑될 컬렉션을 필터하는 쿼리 셀렉터, group의 cond와 같은 기능.
- sort : 쿼리에 적용할 정렬 조건. limit옵션과 사용할때 유용
- limit : 쿼리와 정렬에 적용되어 결과 값의 개수를 제한하는 정수.
- out : 결과가 어떻게 리턴되는지를 결정.-
- finalize : 리듀스가 완료된 후 각 결과 도큐먼트에 적용될 자바스크립트 함수.
- scope : map, reduce, finalize 함수에 의해 액세스 되는 전역 변서의 값을 지정하는 도큐먼트.
- verbose : true일 경우 맵-리듀스 실행 시간에 대한 통계 데이터를 리턴.