Queries

10 Aralık 2021 Cuma 03:20

Persons.find({}) → Hepsini getir.

Persons.find({}).count() → Sayısını verir.

Equality Query

{"name": "Kitty Snow"}

{"age": 38}

{"gender": "female", "eyeColor": "blue"}

↳ AND CONDITION

Person.find({"age":20})

Person.find({"name": "Aurelia Gonzales"})

Person.find({name: "Aurelia Gonzales"})

Person.find({"gender": "female", "eyeColor": "green"})

↳ AND CONDITION

Person.find({"gender": "female", "eyeColor": "green"})
.count()

Operators

{"favoriteFruit": {"$ne": "apple"}} → not equal

{"age": {"$gt": 25}}

↳ greater than

{"eyeColor": {"$in": ["green", "blue"]}}

↳ In

Comparison Operators

$eq $ne $gt $gte $lt $lte

{"favoriteFruit": {"$ne": "orange"}}

{"age": {"$lt": 35}}

{"age": {"$gte": 23}}

{"age": {"$gt": 23, "$lt": 27}}

{"age" : {"$gt":23 , "$lt":27}}

Person.find({"eyeColor":{"$ne" : "green"}}
  .count()

Person.find({"age":{"$gte":25}})
  .count()

Person.find({"age":{"$gte":25 , "$lte":30}})
  .count

>25 <30

Alfabetik

Person.find({"name",{"$gt":"N"}})
  .sort({"name":1})

↳ Ascending order
Artan sıralama

* -1'se descending order azalan sıralama

Tarih ve Saat:

Person.find({"registered":{"$gt":ISODate("2016-08-__-__")}})
Person.find({"registered":{"$gt":ISODate("2016-08-____")}})
  .count()
Person.find({"registered":{"$lte":ISODate("2016-08-____")}})

$in $nin

{"eyeColor":{"$in":["green","blue"]}}
{"favoritFruit":{"$nin":["banana","apple"]}}

Person.find({"age":{"$in":[21,22]}})
Person.find({"age":{"$nin":[21,22]}})
  .count()

$And

{$and[{"gender":"male"},{"age":25}]}

{$and:[
  {"age":{"$ne":25}},
  {"age":{"$gte":20}}
]}

* Koşul aynı alan veya operatörü içeriyorsa; explicit ($and) kullanılır.
* Koşul aynı alan veya operatörü içeriyorsa; explicit ($and) kullanılır.

]}

explicit ($and) kullanılır.

{$and:[{"age":{"$ne":25}},{"age":{"$gte":20}}]} explicit $and

≠

{"age":{"$ne":25}, "age":{"$gte":20}} implicit $and

Person.find({$and:[{gender:"female"},{favoriteFruit:"banana"}]})

Person.find({$and:[{gender:"female"},{favoriteFruit:"banana"}]})

explicit $and

Person.find({gender:"female", favoriteFruit:"banana"})

implicit $and

* Alanlar farklıysa explicit ve implicit $and aynı sonucu verir.

Person.find({$and:[{age:{$ne:25}},{age:{$gte:25}}]})
.sort({age:1})
.count()

o — o — o — o — o

Person.find({$and:[{age:{"$ne":25}},{age:{$gte:25}}]})
• sort({age:1})
• count() → 26'dan başlar

* Alanlar aynıysa explicit ve implicit $and farklı sonuç döndürür. (2. query).

o — oo — oo — ** — ** — o — oo — o

Person.find({age:{$ne:25}, age:{$gte:25}})

↳ 25'ten başlar

Person.find({age:{$gte:25}, age:{$ne:25}})

↳ 26'dan başlar

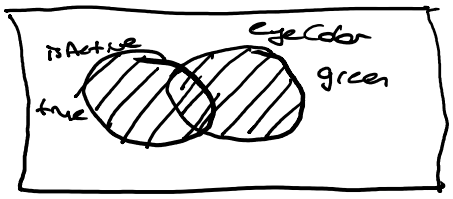
$OR

Person.find({$or:[{"gender":"male"},{"age":25}]})

Person.find({$or:[{"age":20},{"age":27}]}) ↖ Equal

```
Person.find({"age":{"$in":[20,27]}})
```

```
Person.find({$or:[{isActive:true},{eyeColor:"blue"}]})
Person.find({$or:[{isActive:true},{eyeColor:"blue"}]})
```



* $or işleminde ALANLAR aynı ise; $or yerine $in kullanılabilir.

```
Person.find({$or:[{eyeColor:"green"},{eyeColor:"blue"}]})
Person.find({eyeColor:{$in:["green","blue"]}})
Person.find({eyeColor:{$in:["green","blue"]}})
Person.find({$or:[{eyeColor:"green"},{eyeColor:"blue"}]})
```

Query Embedded Documents:

```
Person.find({"company.location.country":"USA"})
Person.find({"company.location.country":"USA"})
Person.find({"company.location.country":"USA","company.title":"DUDE"})
```

$all ve $size operatörleri:

```
Person.find({tags:{$all:["id","ad"]}})
```
→ Sıradan bağımsız sonuçlar gelir.

```
Person.find({tags:{$all:["id","ad"]}})
Person.find({tags:{$all:["id","ad"]}})
Person.find({tags:{$all:["id","ad"]}})
Person.find({tags:{$all:["id","ad"]}})
```

* tags → ARRAY

```
Person.find({tags:{$size:4}})
Person.find({tags:{$siz:4}})
```

```
Person.find ({tags: {$size:4 }})
Person.find ({tags: {$size:4 }})
Person.find ({tags: {$size:4 }})

Person.find({"friends.age" : {$gte:20, $lt:25}})
Person.find ({ "friends.age" : {$gt:20, $lt:25 } })
Person.find ({ "friends.age" : {$gt:20, $lt:25}})
Person.find ({friends : {
                    "name" : "Steve",
                    "age" : 27
            }})
Person.find ({friends: {
                    "name" : "Steve",
                    "age" : 27
            }})

Person.find ({friends : {"name" : "Steve" , "age" : 27}})
Person.find ({friends : {"name": "Steve" , "age" : 27 }})
Person.find ({friends : {"name" : "Steve" , "age" : 27 }})
```

$elemMatch

```
{friends : {$elemMatch : {name : "Bob", registered:false }}}
{friends: { $elemMatch : {registered : false , age:25 }}}
```

] Sıra önemli değil

* iç içe dizilerde, koşulları sağlayan elemanı getirir.

```
{ friends: { $elemMatch: {registered:false , age:25 }}}
Person.find ({ friends : { $elemMatch: {name: "Lora", age:23 }}})
Person.find ({friends : {$elemMatch: {name: "Lora", age:27 }}
```

$EXIST — $TYPE

```
{tags : {$exists : true }}
{name : {$exists : false }}
```

{ posts : { $type : "int" } }

{ name : { $type : 2 } } → 2 = string

{ quantity : { $type : ["int", "double"] } }

{ index : { $type : "int", $eq : 5 } }

Number Id	Type	String ID
2	String	"string"
3	object	"object"
4	Array	"array"
8	Bodean	"bool"
16	32-bit integer	"int"
18	64-bit integer	"long"
1	Double	"double"
9	Date	"date"
7	Object Id	"objectId"

Person.find({name:{$exists:true}})

Person.find({name:{$exists:true}})

Person.find({index:{$type:"int"}})

Person.find({index:{$type:"int", $eq:10}})

Person.find({index:{$type:"int", $eq:10}})

Person.find({index:{$type:"int", $eq:10}})

FILTER FIELDS

Person.find({}, {name:1, age:1, eyeColor:1}) → _id gelir

Person.find({}, {name:1, age:1, eyeColor:1})

Person.find({}, {name:1, age:1, eyeColor:1, "company.location":1})

Person.find({}, {name:1, age:1, eyeColor:1, "company.location":1})

Person.find({}, {name:1, age:1, eyeColor:1, _id:0})

↳ id gelmez

Person.find({}, {name:1, age:1, ejected:1, _id:0})

↳ _id gelmez.

Person.find({}, {name:0, age:0})

↳ name ve age alanı dışında hepsi gelir.

Client ⟷ Server arası hızı artırır.

*Gelen verilerin yorumlanmasına gerek kalmaz.

*PROJECTION

REGEX

{city: {"$regex": /ton/i}}

Şehirler içinde "ton" kombinasyonunu arar.

Büyük-küçük harf hepsini alır.

{tags: {"$regex": /^ad$/, $options: "i"}}

↳ in case sensitive

tağlar içinde "a" ile başlayan, "d" ile biten şeyleri arar

Büyük-küçük harf farketmez.

{state: {"$regex": "ca"}}

statelerde "ca" harfleri bulunanları arar. } case-sensitive

Sadece küçük harfleri getirir.

Person.find({name: {$regex: /rel/i}}) → içinde "rel" olan isimleri getir.

Person.find({name: {$regex: /rel/i}}) → içinde "rel" olan isimleri getir.

Person.find({name: {$regex: /rell, $options: "i"}})

Person.find({name: {$regex: /Aur/}})

↳ case sensitive /^Aur/ → A ile başlayan

Person.find({name: {$regex: "Aur"}})