2-10 Mongo upiti

Tipovi podataka

Kao i u SQL bazama podataka, tako i MongoDB atributi u objektima imaju tipove. Neki od najčešćih tipova su: null, ObjectId, number, double, string, array, binData, bool, date, int, long. U MongoDB dokumentaciji može se pronaći paralelno poređenje sa SQL jezikom i naredbama kao i tipovima u MongoDB: https://docs.mongodb.com/manual/reference/sql-comparison/

Primjer unosa objekta sa različitim tipovima podataka:

```
db.studenti.insert({
   ime: 'Alisa',
   prezime: 'Alisovic',
   datum_rodj: ISODate('2005-03-01'),
   osobine: {
      godine: 15,
      visina: 160,
      tezina: NumberInt(55)
   },
   koeficijent: Number(3.37),
   grupa: 'D'
});
```

Ovo je primjer komplikovanijeg objekta. Za pretraživanje objekta spomenuta je `find({})` metoda. Tako se može pretraživati na nekoliko načina.

```
1. db.studenti.find({ ime: 'Alisa' });
2. db.studenti.find({ prezime: 'Alisovic' });
3. db.studenti.find({ ime: 'Alisa', prezime: 'Alisovic' });
4. db.studenti.find({ datum_rodj: ISODate('2005-03-01 00:00:00.000Z') });
5. db.studenti.find({ 'osobine.godine': 15 });
```

Ovdje se mogu vidjeti primjeri pretraživanja na različite načine, uključujući i pretraživanje po atributima pod objekata. Za izmjenu podataka, koristi se update() metoda:

```
db.studenti.update({ ime: 'Denis' },
   { $set:
      { prezime: 'Izmjena', godine: NumberInt(20) }
   });
```

Na prvom mjestu se naznači objekat za pretraživanje, a na drugom mjestu objekat sa specijalnim operaterom `$set`, i objektom sa atributima koje želimo primjeniti. Efekat ove komande je da

nepoklapajući atributi u orginalnom objektu neće biti uklonjeni već će biti izmjenjeni samo podudarajući atributi ili dodani novi. Treba napomenuti da se `_id` atribut ne može mijenjati.

Recimo da želimo dodati niz objekata za studenta:

```
db.studenti.update({ ime: 'Denis' }, { $set: {
   hobiji: [
     { naziv: 'igre', godine: 3 },
     { naziv: 'ples', godine: 2 },
     { naziv: 'teretana', godine: 4 },
     { naziv: 'fudbal', godine: 8 }
   ]
});
```

Dodan je novi niz, liste hobija za sve studente kojima je ime `Danis`. Na isti način je mogao biti potražen dokument sa specifičnim `_id`. Da se pretraže studenti koji za prvi hobij imaju igre:

```
db.studenti.find({
    'hobiji.0.naziv': 'igre'
});
```

Operatori `$lt, $gt, $lte, $gte, $eq`

Ovi operatori omogućavaju pretraživanje poput manje (`$lt`), veće (`$gt`), manje ili jednako (`$lte`), veće ili jednako (`$gte`) i jednako (`$eq`). Ovi operatori se mogu primjeniti na sve tipove podataka.

```
db.studenti.find({
    godine: { $gte: 20 }
});
```

Operator `$in` i `$all`

Ukoliko objekat ima niz brojeva poput:

```
db.studenti.insert({
   ime: 'Kenan', prezime: 'Kenanovic',
   koeficijenti: [ 3, 8, 10, 11, 18, 21, 27 ]
});
```

I želimo da pretražimo sve objekte koji u atributu koeficijent imaju brojeve 11 ili 13, u te svrhe se može koristiti operator `$in`.

```
db.studenti.find({ koeficijenti: { $in: [11, 13] } });
```

Za razliku operatora `$in`, operator `$all` zahtjeva da atribut ima sve vrijednosti navedene u upitu. Npr. da smo koristili `$all` u prethodnom upitu, ne bi pronašao Kenana jer među koeficijentima ima samo 11 ali nema 13.

```
db.studenti.find({ koeficijenti: { $all: [11, 13] } });
```

Jedinstvene vrijednosti

Ukoliko želimo pronaći sva imena u kolekciji studenti, možemo koristiti `distinct()` funkciju, kojoj se proslijeđuje naziv atributa za kojeg je potrebno pronaći jedinstvene vrijednosti.

```
db.studenti.distinct('ime');
```

Ukoliko je potrebno tražiti jedinstvene vrijednosti na podskupu podataka, može se dodati upit kojim se sužava skup unutar kojeg se traži jedinstvena vrijednost.

```
db.studenti.distinct('ime', { razred: 3 }});
```

Tako da će ovaj upit pronaći jedintvena imena ali samo za studente za koje je razred=3.

Operatori `$and` i `$or`

Ovo su logički operatori koji mogu pomoći definisanje preciznijih zahtjeva za pretraživanjem. Npr. svi studenti koji su prvi ili treći razred:

```
db.studenti.find({ $or: [{ razred: 1 }, { razred: 3 }] });
```

Dakle, za ove operatore potrebno je definisati niz objekata koji predstavljaju uslove nad kojim se ovaj logički operator definiše. `$and` i `$or` se mogu kombinovati i sa ostalima pa npr traženje studenata koji imaju između 18 i 25 bi izgledalo ovako:

```
db.studenti.find({
      $and: [{ godine: { $gte: 18 } }, { godine: { $lt: 25 } } ]
});
```

Moguće je i kombinovati sve ove operatore. Npr. svi student koji su prvi ili treći razred a upisani su na matematiku i fiziku:

```
db.studenti.find({
      $and: [
                  { predmeti: { $all: ['Matematika', 'Fizika'] } },
                  { $or: [{ razred: 1 }, { razred: 3 }] }
      ]
});
```