Mongo DB

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

KMI/DATAB Databáze

Příklad

- Uvažujme kolekci:


```
[
  {
    "_id": {"$oid": "619df992d9d68acaaf511640"},
    "jmeno": "Jiří",
    "prijmeni": "Zacpal",
    "rocnik": 1,
    "datum_zahajeni_studia": {"$date": "2021-01-14T23:00:00Z"},
    "stav": "studuje",
    "pohlavi": "muž"
  },
  {
    "_id": {"$oid": "619df992d9d68acaaf511642"},
    "jmeno": "Martina",
    "prijmeni": "Kučerová",
    "rocnik": 1,
    "stav": "studuje",
    "pohlavi": "žena"
  },
  {
    "_id": {"$oid": "619df992d9d68acaaf511643"},
    "jmeno": "Karel",
    "prijmeni": "Svoboda",
    "rocnik": 1,
    "datum_zahajeni_studia": {"$date": "2021-11-24T08:36:34.169Z"},
    "stav": "studuje",
    "pohlavi": "muž"
  },
 {
     "_id": {"$oid": "61a4de0abfd514a4ff9f38d5"},
     "jmeno": "Pavel",
     "prijmeni": "Novák",
     "rocnik": 1,
     "stav": "nestuduje",
     "pohlavi": "muž"
   }
]
```

Regulární výraz

- Nový typ hodnoty: regulární výraz
- Zapisuje se:/pattern/
- používá se PCRE (Perl compatible regular expressions)
- Podmínka, zda hodnota klíče odpovídá regulárnímu výrazu:

  { <key>: { $regex: <regular_expression> } }

Příklad

- Najdeme všechny osoby, jejichž jméno začíná na J:

```
db.osoby.find({ jmeno: { $regex: /^J/ } })
```

- Najdeme všechny osoby, jejichž jméno končí na el:

```
db.osoby.find({ jmeno: { $regex: /el$/ } })
```

- Najdeme všechny osoby, jejichž jméno obsahuje e:

```
db.osoby.find({ jmeno: { $regex: /e/ } })
```

- Najdeme všechny osoby, jejichž jméno začíná na J nebo P:

```
db.osoby.find({ jmeno: { $in: [/^P/,/^J/] } })
```

Zjednodušení konjunkce

- Vezměme konujnkci:

  { $and: [<condition1>, ..., <conditionN>] }

- kde podmínky <condition1>, ..., <conditionN> jsou objekty, které mají jedinečné klíče.

- Pak můžeme konjunkci ekvivalentně zapsat jako objekt:

  { <key1>: <value1>, ..., <keyM>: <valueM>}

- kde páry<key1>: <value1>, ..., <keyM>: <valueM>jsou všechny páry v objektech <condition1>, ..., <conditionN>.

Příklad

- Najdeme všechny osoby, které studují a jsou v prvním ročníku:

```
db.osoby.find({ rocnik: { $eq:1  },stav:{$eq:"studuje"}})
```

- Lze to napsat i pomocí implicitní konjukce:

```
db.osoby.find({ rocnik: 1 ,stav:"studuje"})
```

Úkol

- Vytvořte kolekce predmety a vložte do ní dokumenty podle tabulky.
- Vyhledejte dokumenty, ve kterých název předmětu obsahuje řetězec „geb“.
- Pomocí implicitní konjukce najděte dokumenty z katedry KMI, které se učí na učebně 5.003.

nazev	katedra	zkratka	ucitel	ucebna
Databáze	KMI	DATAB	Zacpal	5.004
Základy programování 1	KMI	ZPC1	Večerka	5.003
Algebra 1	KAG	ALG1	Zacpal	5.003
Struktura počítačů	KMI	STRUP	Zacpal	5.003
Algebra 2	KAG	ALG2	Večerka	5.004

Příklad

- Do dokumenty můžeme vkládat i dokumenty. Například vložíme adresu k osobě:

```
db.osoby.updateMany({prijmeni:"Zacpal"},{$set:{adresa:{mesto:"Olomo
uc",ulice:"Jeremiášova",cp:874,psc:"779 00"}}})
```

- Doplníme adresy i ostatním osobám:

```
db.osoby.updateMany({prijmeni:"Novák"},{$set:{adresa:{mesto:"Prostě
jov",ulice:"Olomoucká",cp:45}}})
db.osoby.updateMany({prijmeni:"Kučerová"},{$set:{adresa:{mesto:"Olo
mouc",ulice:"Komenského",cp:43,psc:"770 00"}}})
db.osoby.updateMany({prijmeni:"Svoboda"},{$set:{adresa:{mesto:"Olom
ouc",ulice:"Palackého",cp:12,psc:"779 00"}}})
```

Pořadí klíčů

- Databáze uvažuje i pořadí klíčů v objektu.
- Proto se dva objekty rovnají, když mají stejné klíče (uvedené ve stejném pořadí) a hodnoty přiřazené objekty každému společnému klíči se rovnají.

Příklad

- Vyhledáme osobu podle adresy:

```
db.osoby.find({adresa:{mesto:"Olomouc",ulice:"Palackého",c
p:12,psc:"779 00"}})
```

- Pokud zaměníme klíče:

```
db.osoby.find({adresa:{ulice:"Palackého", mesto:
"Olomouc", cp:12,psc:"779 00"}})
```

Cesta v objektu

- Cesta v objektu je řetězec, který se skládá z kroků.
- Krok v cestě je opět řetězec.
- Jednotlivé kroky jsou v cestě odděleny tečkou.
- Klíč dokumentu je krok v cestě.

- Například cesta s dvěma kroky "adress" a "town": "adress.town "
- Prázdná cesta: ""

Příklad

- Vyhledáme všechny osoby z Olomouce:

```
db.osoby.find({"adresa.mesto":"Olomouc"})
```

- S použitím implicitní konjukce:

```
db.osoby.find({"adresa.mesto":"Olomouc","adresa.ulice":"Komenského"})
```

Úkol

- Do dokumentů v kolekce predmety doplňte dokumenty o zakončení předmětu s klíči: zakonceni (zkouška-zápočet), zapocet_pred_zkouskou (ano-ne), kredity (pocet).
- Vyhledejte dokumenty, které mají předepsanou zkoušku.

nazev	katedra	zkratka	ucitel	ucebna
Databáze	KMI	DATAB	Zacpal	5.004
Základy programování 1	KMI	ZPC1	Večerka	5.003
Algebra 1	KAG	ALG1	Zacpal	5.003
Struktura počítačů	KMI	STRUP	Zacpal	5.003
Algebra 2	KAG	ALG2	Večerka	5.004

Příklad

- Do dokumentů můžeme vkládat pole:

```
db.osoby.updateMany({prijmeni:"Zacpal"},{$set:{predmety:["
KMI/DATAB","KMI/ZPC1","KMI/ZPP1"]}})
```

- Doplním předměty i ostatním osobám:

```
db.osoby.updateMany({prijmeni:"Novák"},{$set:{predmety:["K
MI/DATAB","KAG/ALG1"]}})
db.osoby.updateMany({prijmeni:"Svoboda"},{$set:{predmety:[
"KMI/STRUP","KMI/ZPC1","KMI/ZPP1"]}})
db.osoby.updateMany({prijmeni:"Kučerová"},{$set:{predmety:
["KMI/DATAB","KMI/ZPC1","KMI/ZPP1","KAG/ALG1"]}})
```

Příklad

- Karlovi zapíšeme pouze DATAB a STRUP:

```
db.osoby.updateMany({jmeno:"Karel"},{$set:{predmety:["KMI/DATAB","KMI/STRUP"]}})
```

- Karlovi přidáme předmět ZPC1:

```
db.osoby.updateMany({jmeno:"Karel"},{$push:{predmety:"KMI/ZPC1"}})
```

- Karlovi odebereme STRUP:

```
db.osoby.updateMany({jmeno:"Karel"},{$pull:{predmety:"KMI/STRUP"}})
```

- Karlovi přidá na konec pole předměty ZPP1 a ALG1:

```
db.osoby.updateMany({jmeno:"Karel"},{$push:{predmety:{$each:["KMI/ZPP1","KMI/ALG1"]}}})
```

Příklad

- Vyhledáme osoby, které mají zapsány právě dva předměty:

```
db.osoby.find({predmety:{$size:2}})
```

Operátor $in

- Pole hodnot můžeme použít i v dotazech.
- Operátor $in slouží k vytvoření podmínky na hodnotu:

  { $in: <array> }

- kde <array> je pole hodnot.
- Podmínka je v hodnotě splněna, pokud se hodnota nachází v poli <array>.

Příklad

- Vyhledáme osoby, které se jmenují Pavel nebo Karel:

```
db.osoby.find({jmeno:{$in:["Karel","Pavel"]}})
```

- Vyhledáme osoby, které bydlí v Olomouci nebo Prostějově:

```
db.osoby.find({"adresa.mesto":{$in:["Olomouc","Prostějov"]
}})
```

- Vyhledáme osoby, které mají zapsány předměty DATAB a ZPC1:

```
db.osoby.find({predmety:{$in:["KMI/DATAB","KMI/ZPC1"]}})
```

Index prvku v poli

- Index prvku v poli je krok v cestě.
- Předpokládejme, že hodnota <value> je pole a krok <step> je indexem jejího prvku.
- Pak hodnota po kroku <step> z pole <value> je hodnota pole <value> na indexu <step>.
- Uvažujme podmínku dokumentu:

  { <path> : <condition> }

- Pokud cesta <path> v dokumentu nedává smysl, pak není podmínka dokumentu splněna.

Příklad

- Vyhledáme osoby, které mají první zapsaný předmět DATAB:

```
db.osoby.find({"predmety.0":"KMI/DATAB"})
```

Úkol

- Do dokumentů v kolekci predmety doplňte studenty (z kolekce osoby), kteří mají daný předmět zapsán.
- Vyhledejte předměty, které mají zapsány 2 studenti.
- Vyhledejte předměty, které mají zapsány Pavel nebo Karel.

Bodovaný úkol

Vytvořte kolekci knihy podle této tabulky:

nazev	vydavatel	rok_vydani	pocet_stran	cena	zanr	typ
Stopařův průvodce po galaxii	MF	1985	214	120	scifi	kniha
Pán prstenů - Dvě věže	MF	1948	251	240	fantasy	kniha
Kedrigern a hlas pro princeznu	MF	1996		53	fantasy	kniha
Hobit	MF	1950	410	178	fantasy	kniha
Barva kouzel	Talpress	1989	221	358	fantasy	ebook
Stráže! Stráže!	Talpress	2000		214	fantasy	ebook
Lehké fantastično	Talpress	1999	145	415	fantasy	ebook

- Ke každé knize doplňte autora (jméno, příjmení, národnost) - dokument.
- Ke knihám přidejte osoby (jako pole), které si knihy půjčily.