Laboratorium 2 - zadania z zajęć: MongoDB

Aleksandra Mazur

1 Zadanie 4

Polecenie dodające do stworzonej bazy kolekcję „student". Wprowadzenie własnych danych do kolekcji: imię, nazwisko, obecność (typ bool), ocena z lab. (null), aktualna data, zaliczone przedmioty (min 3 przykładowe).

```
db.students.insert({
    name: "Aleksandra",
    surname: "Mazur",
    presence: true,
    mark: null,
    date: ISODate("2020-03-24"),
    finished: [ {subject: "database"}, {subject: "scala"}, {subject: "physics"} ]
})
```

```
/* 1 */
{
    "_id" : ObjectId("5e79320bb5fbb85afe6afe53"),
    "name" : "Aleksandra",
    "surname" : "Mazur",
    "presence" : true,
    "mark" : null,
    "date" : ISODate("2020-03-24T00:00:00.000Z"),
    "finished" : [
        {
            "subject" : "database"
        },
        {
            "subject" : "scala"
        },
        {
            "subject" : "physics"
        }
    ]
}
```

2 Zadanie 5

2.1 Zapytanie zwracające ilość miejsc ocenianych na 5 gwiazdek (pole stars, kolekcja business).

```
db.business.count({stars:5})
```

0.131 sec.

```
5097
```

2.2 Zapytanie zwracające ilość restauracji w każdym mieście, wynik jest posortowany malejąco na podstawie liczby. Pole categories w dokumencie business musi zawierać wartość Restaurants.

```
db.business.aggregate([
    { $match: { "categories": "Restaurants" } },
    { $group: { _id: "$city", count: { $sum: 1 } } },
    { $sort: { count: -1 } }
])
```

```
/* 1 */
{
    "_id" : "Las Vegas",
    "count" : 3855.0
}

/* 2 */
{
    "_id" : "Phoenix",
    "count" : 2493.0
}

/* 3 */
{
    "_id" : "Edinburgh",
    "count" : 1049.0
}

/* 4 */
{
    "_id" : "Scottsdale",
    "count" : 1023.0
}

/* 5 */
{
    "_id" : "Mesa",
    "count" : 693.0
}

/* 6 */
{
    "_id" : "Madison",
    "count" : 679.0
}
```

2.3 Zapytanie zwracające ilość hoteli (atrybut categories powinien mieć wartość Hotels) w każdym stanie/okręgu (state), które posiadają darmowe Wi-fi (pole attributes, klucz-wartość 'Wi-Fi': 'free') oraz ocenę co najmniej 4.5 gwiazdki.

```
db.business.aggregate([
    { $match: { $and: [ {"categories": "Hotels"},
        {"attributes.Wi-Fi": "free"},
        {"stars": {$gte: 4.5}}]}},
    { $group: { _id: "$state", count: { $sum: 1 } } },

])
```

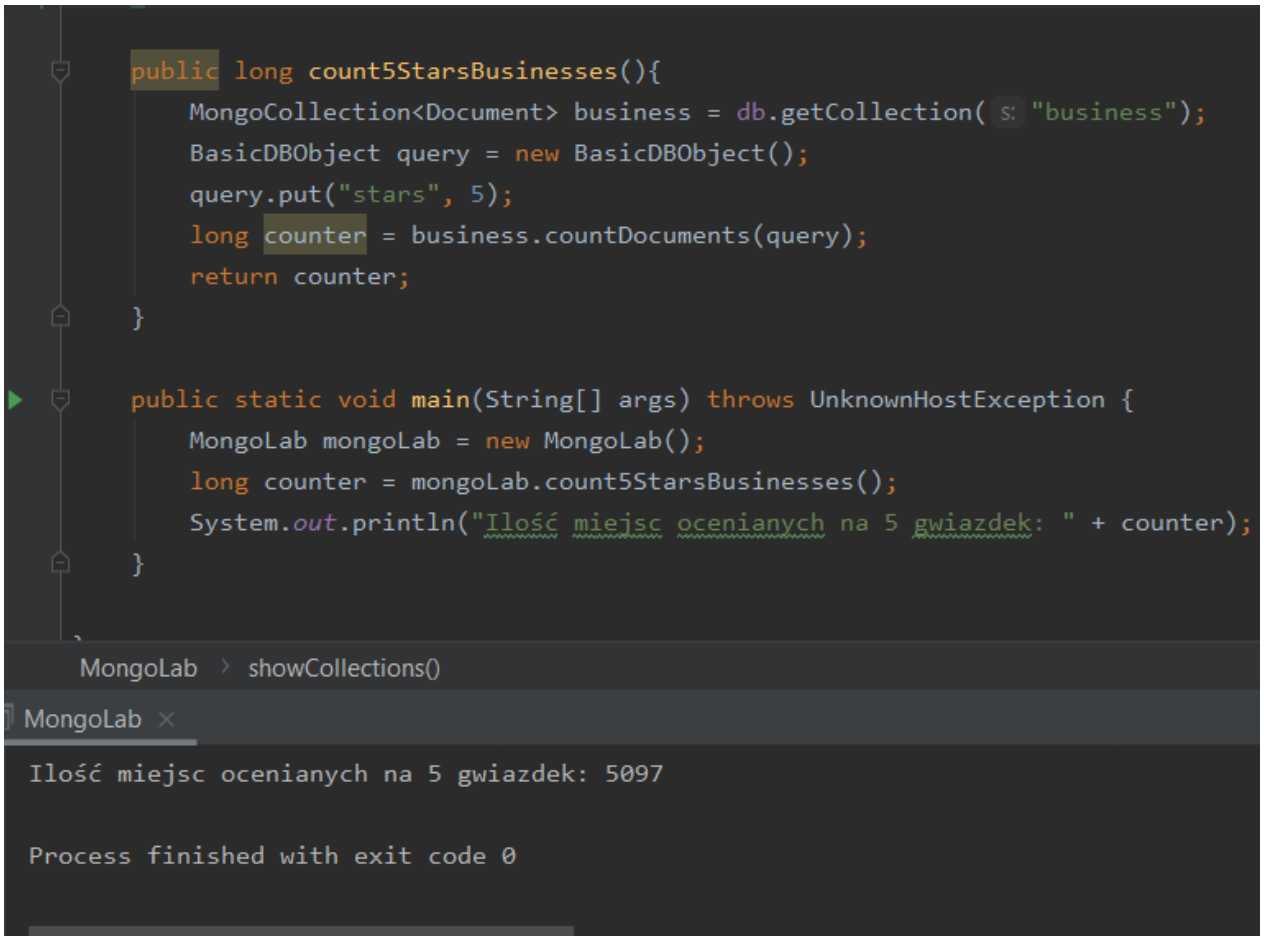
business 0.095 sec.

```
/* 1 */
{
    "_id" : "ON",
    "count" : 2.0
}

/* 2 */
{
    "_id" : "NV",
    "count" : 10.0
}

/* 3 */
{
    "_id" : "MLN",
    "count" : 1.0
}

/* 4 */
{
    "_id" : "EDH",
    "count" : 13.0
}

/* 5 */
{
    "_id" : "WI",
    "count" : 10.0
}

/* 6 */
{
    "_id" : "AZ",
    "count" : 33.0
```

3 Zadanie 6

Zadania z punktu 5 z poziomu języka Java.

3.1 Odpowiednik zapytania z punktu 5a



```
    public long count5StarsBusinesses(){
        MongoCollection<Document> business = db.getCollection( s: "business");
        BasicDBObject query = new BasicDBObject();
        query.put("stars", 5);
        long counter = business.countDocuments(query);
        return counter;
    }

    public static void main(String[] args) throws UnknownHostException {
        MongoLab mongoLab = new MongoLab();
        long counter = mongoLab.count5StarsBusinesses();
        System.out.println("Ilość miejsc ocenianych na 5 gwiazdek: " + counter);
    }
```

MongoLab › showCollections()

MongoLab ×

```
Ilość miejsc ocenianych na 5 gwiazdek: 5097

Process finished with exit code 0
```

3.2 Odpowiednik zapytania z punktu 5b

```
    public void cityRestaurants(){
        MongoCollection <Document> business = db.getCollection( s: "business");
        AggregateIterable aggResult = business.aggregate(Arrays.asList(
                Aggregates.match(Filters.eq( fieldName: "categories", value: "Restaurants")),
                Aggregates.group( id: "$city", Accumulators.sum( fieldName: "count", expression: 1)),
                Aggregates.sort(Sorts.descending( ...fieldNames: "count"))
        ));

        for(Object result: aggResult){
            System.out.println(result);
        }
    }

    public static void main(String[] args) throws UnknownHostException {
        MongoLab mongoLab = new MongoLab();
        mongoLab.cityRestaurants();
    }
}
```

```
Document{{_id=Las Vegas, count=3855}}
Document{{_id=Phoenix, count=2493}}
Document{{_id=Edinburgh, count=1049}}
Document{{_id=Scottsdale, count=1023}}
Document{{_id=Mesa, count=693}}
Document{{_id=Madison, count=679}}
Document{{_id=Tempe, count=672}}
Document{{_id=Henderson, count=564}}
Document{{_id=Chandler, count=548}}
Document{{_id=Glendale, count=422}}
Document{{_id=Gilbert, count=317}}
Document{{_id=Peoria, count=221}}
Document{{_id=North Las Vegas, count=198}}
Document{{_id=Surprise, count=144}}
Document{{_id=Goodyear, count=119}}
Document{{_id=Waterloo, count=117}}
Document{{_id=Avondale, count=100}}
Document{{_id=Kitchener, count=96}}
Document{{_id=Queen Creek, count=82}}
Document{{_id=Middleton, count=66}}
Document{{_id=Cave Creek, count=63}}
Document{{_id=Casa Grande, count=61}}
Document{{_id=Fountain Hills, count=47}}
Document{{_id=Apache Junction, count=44}}
Document{{_id=Buckeye, count=42}}
Document{{_id=Sun Prairie, count=39}}
Document{{_id=Fitchburg, count=38}}
Document{{_id=Maricopa, count=37}}
Document{{_id=Monona, count=32}}
Document{{_id=Wickenburg, count=31}}
Document{{_id=Sun City, count=31}}
```

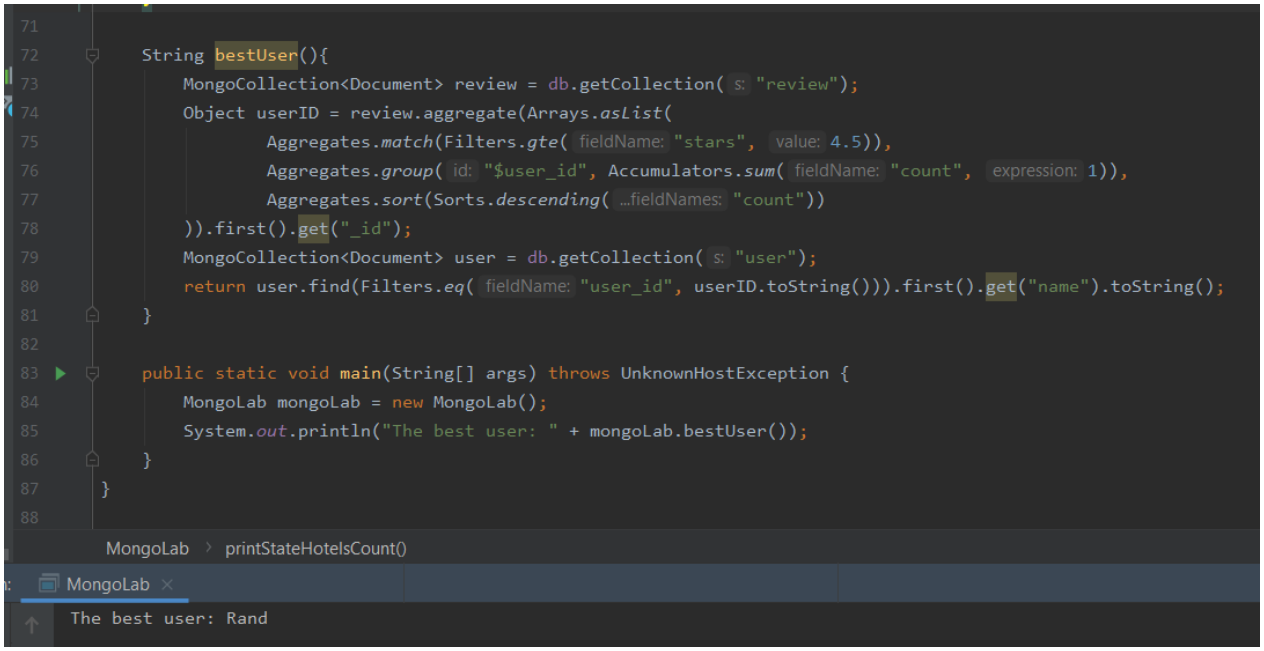
3.3 Odpowiednik zapytania z punktu 5c

```
public void printStateHotelsCount(){
    MongoCollection<Document> business = db.getCollection( s: "business");
    AggregateIterable aggResult = business.aggregate(Arrays.asList(
            Aggregates.match(Filters.and(Filters.eq( fieldName: "categories",  value: "Hotels"),
                    Filters.eq( fieldName: "attributes.Wi-Fi",  value: "free"),
                    Filters.gte( fieldName: "stars",  value: 4.5)
            )),
            Aggregates.group( id: "$state", Accumulators.sum( fieldName: "count",  expression: 1))
    ));
    for(Object res: aggResult){
        System.out.println(res);
    }
}

public static void main(String[] args) throws UnknownHostException {
    MongoLab mongoLab = new MongoLab();
    mongoLab.printStateHotelsCount();
}
```

```
Document{{_id=MLN, count=1}}
Document{{_id=NV, count=10}}
Document{{_id=WI, count=10}}
Document{{_id=EDH, count=13}}
Document{{_id=AZ, count=33}}
Document{{_id=ON, count=2}}
```

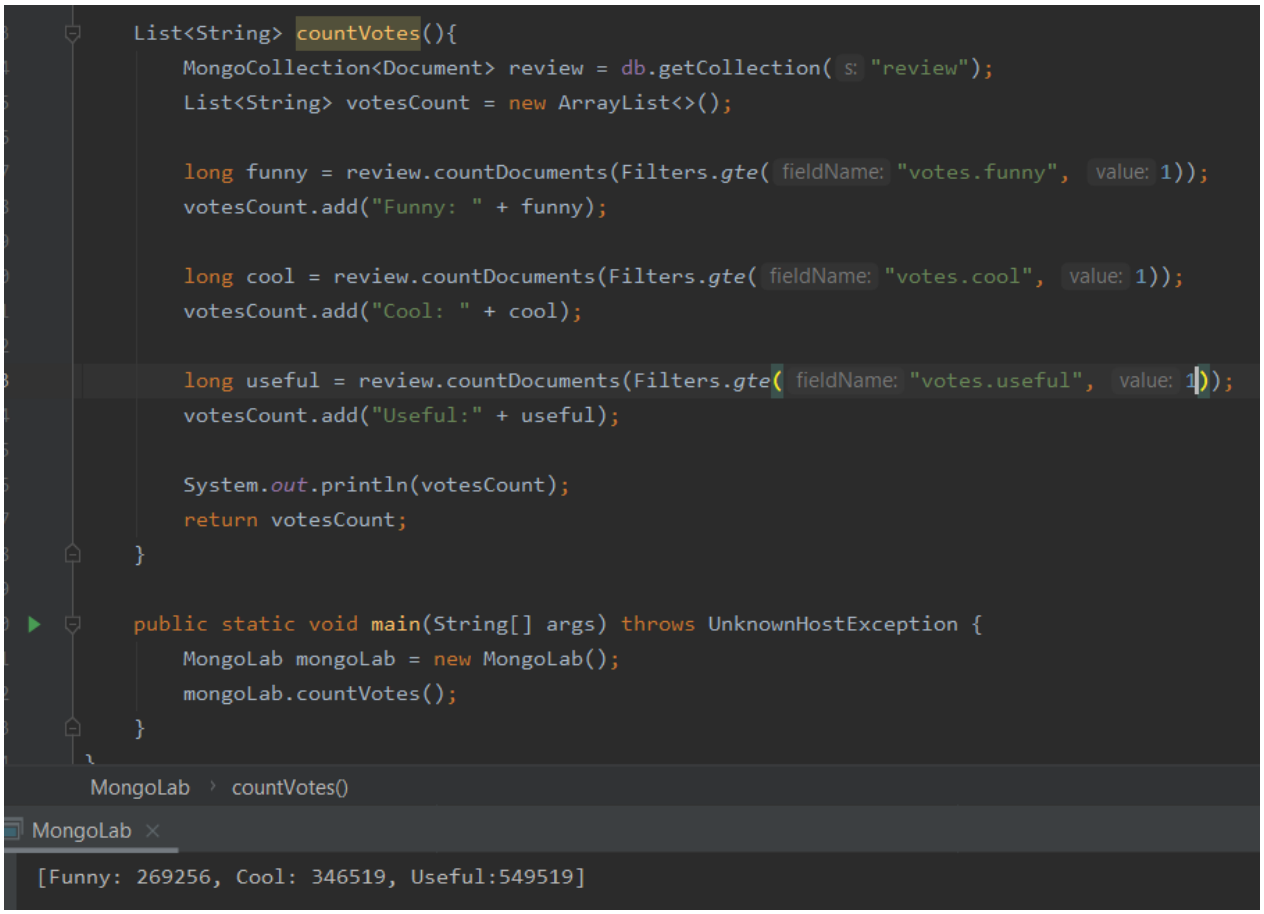
4 Zadanie 7

Kod w języku Java, który zwróci użytkownika (nazwa użytkownika) o największej liczbie pozytywnych recenzji (ocena co najmniej 4.5).



5 Zadanie 8

Kod w języku Java, który zwrócił, ile recenzji posiadają oceny z każdej kategorii: funny, cool, useful. Przypisanie recenzji do kategorii oznacza, że przynajmniej jedna osoba zagłosowała na recenzje w tej kategorii).


List<String> countVotes(){
MongoCollection<Document> review = db.getCollection(s: "review");
List<String> votesCount = new ArrayList<>();
long funny = review.countDocuments(Filters.gte(fieldName: "votes.funny", value: 1));
votesCount.add("Funny: " + funny);
long cool = review.countDocuments(Filters.gte(fieldName: "votes.cool", value: 1));
votesCount.add("Cool: " + cool);
long useful = review.countDocuments(Filters.gte(fieldName: "votes.useful", value: 1));
votesCount.add("Useful:" + useful);
System.out.println(votesCount);
return votesCount;
}
public static void main(String[] args) throws UnknownHostException {
MongoLab mongoLab = new MongoLab();
mongoLab.countVotes();
}
MongoLab › countVotes()
MongoLab
[Funny: 269256, Cool: 346519, Useful:549519]