Bazy Danych
NoSQL - MongoDB

Barbara Wojtarowicz
grupa czw. 14:40 B

Raport z wykonania ćwiczenia

1. Zadania: 1, 2, 3

Uruchomiłam usługę MongoDB, zaimportowałam zbiór danych yelp_dataset oraz połączyłam się z bazą z użyciem Studio 3T for MongoDB.

2. Zadanie 4

a. Stworzyłam kolekcję “students”. Dodałam swoje dane do kolekcji:

```
db.students.insert({
      firstName: "Barbara",
      lastName: "Wojtarowicz",
      presence: true,
      mark: null,
      currentDate: ISODate("2021-29-04"),
      positiveCourses: [
      "algorithms and data structures",
      "data bases",
      "maths"
      ]
})
```

b. Wyświetliłam dodane dane w formie JSON:

```
{
    "_id" : ObjectId("608abc2b9f613b45a177ee30"),
    "firstName" : "Barbara",
    "lastName" : "Wojtarowicz",
    "presence" : true,
    "mark" : null,
    "currentDate" : ISODate("2023-05-04T00:00:00.000+0000"),
    "positiveCourses" : [
        "algorithms and data structures",
        "data bases",
        "maths"
    ]
}
```

3. Zadanie 5

Za pomocą narzędzia Studio 3T wykonałam zapytania, które pozwoliły uzyskać następujące wyniki:

a. Ile miejsc ocenianych na 5 gwiazdek (pole stars, kolekcja business).

```
business = db.getCollection("yelp_academic_dataset_business")
business.find({stars: 5.0}).count()
```

```
23  WojtarowiczBarbara.yelp_academic_dataset_business
24  5097
```

b. Ile restauracji znajduje się w każdym mieście (pole categories w dokumencie business musi zawierać wartość Restaurants).

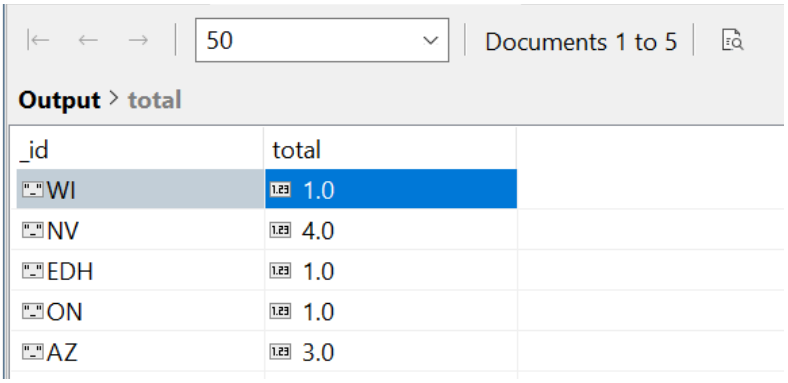
```
business.aggregate([
      { $match: { categories: "Restaurants" } },
      { $group: { _id: "$city", total: { $sum: 1 } } },
])
```

Poniżej pierwsze 10 ze 108 rezultatów dla tego zapytania:

_id	total
Carefree	16.0
Black Canyon City	1.0
Sun City West	9.0
Boulder City	7.0
Coolidge	6.0
Fitchburg	38.0
Cave Creek	63.0
Cottage Grove	6.0
Laveen	25.0
Higley	1.0

c. Podaj liczbę hoteli (w atrybucie categories powinno być wymienione Hotels jako jedna z wartości) w każdym stanie/okręgu (state), które posiadają darmowe Wi-fi (w polu attributes powinna znaleźć się wartość ‘Wi-Fi’:’free’) oraz ocenę co najmniej 4.5 gwiazdki (pole stars).

```
db.getCollection("yelp_academic_dataset_business").aggregate([
      { $match: {categories: {$regex : ".*Hotels.*"},attributes:
{"Wi-Fi" : "free"}, stars: {$gte : 4.5} } },
      { $group: {_id: "$state", total: {$sum: 1} } }
])
```

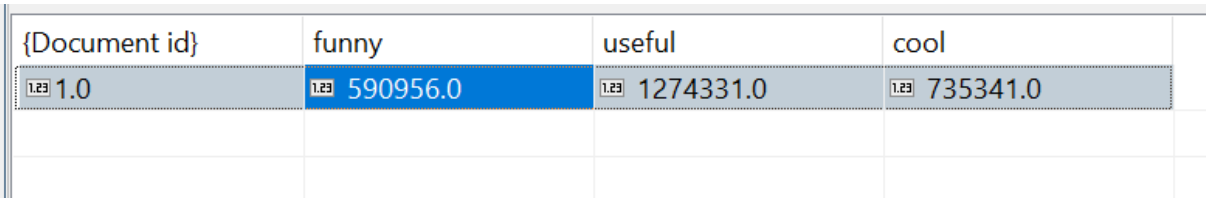


d. MapReduce: Recenzje mogą być oceniane przez innych użytkowników jako cool, funny lub useful (jedna recenzja może mieć kilka głosów w każdej kategorii). Napisz zapytanie, które zwraca dla każdej z tych kategorii, ile sumarycznie recenzji zostało oznaczonych przez te kategorie.

```
var mapFun = function(){
      var key = 1;
      var value = { funny: this.votes.funny, useful: this.votes.useful,
cool: this.votes.cool }

      emit(key, value);
};

var reduceFun = function(key, vals){
      reducedVal = { funny: 0, useful: 0, cool: 0 };

      for (var idx = 0; idx < vals.length; idx++){
      reducedVal.funny += vals[idx].funny;
      reducedVal.useful += vals[idx].useful;
      reducedVal.cool += vals[idx].cool;
      }

      return reducedVal;

};

db.getCollection("yelp_academic_dataset_review").mapReduce(
      mapFun,
      reduceFun,
      {
      out: "categories_totals"
      }
);
```

Wynik zapytania zapisałam do dokumenty “categories_totals”, dlatego końcowym elementem zapytania jest:

```
db.getCollection("categories_totals").find({})
```



{Document id}	funny	useful	cool
1.0	590956.0	1274331.0	735341.0

4. Zadanie 6

Wykonałam zadania z punktu 5. z poziomu języka Java, wykorzystując środowisko Eclipse.

a. Odpowiednik zapytania z punktu 5a.

```
      public long fiveStarBusiness() {
       MongoCollection<Document> business =
db.getCollection("yelp_academic_dataset_business");
       BasicDBObject query = new BasicDBObject();
       query.put("stars", 5);
       long result = business.countDocuments(query);
       return result;
    }

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

```
maj 04, 2021 3:40:07 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:38}] to 127
Liczba miejsc ocenionych na 5 gwiazdek: 5097
```

b. Odpowiednik zapytania z punktu 5b.

```
      /* Create our pipeline operations */
    public void restaurantsInCities() {

       MongoCollection coll =
db.getCollection("yelp_academic_dataset_business");

       // First with $match
```

```
        DBObject match = new BasicDBObject("$match", new
BasicDBObject("categories", "Restaurants"));

        // Now build $projection operation
        DBObject fields = new BasicDBObject("city", 1);
        fields.put("count", 1);
        fields.put("_id", 0);
        DBObject project = new BasicDBObject("$project", fields);

        // Now the $group operation
        DBObject groupFields = new BasicDBObject("_id", "$city");
        groupFields.put("count", new BasicDBObject("$sum", 1));
        DBObject group = new BasicDBObject("$group", groupFields);

        // Now the $sort operation
        DBObject sort = new BasicDBObject("$sort", new
BasicDBObject("count", -1));

        // Now run aggregation
        List<DBObject> pipeline = Arrays.asList(match, project, group,
sort);
        AggregateIterable output = coll.aggregate(pipeline);

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

    }



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

```
<terminated> MongoLab [Java Application] C:\Users\basia\.p2\pc
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}}
```

c. Odpowiednik zapytania z punktu 5c.

```
      public void stateHotels() {
       MongoCollection<Document> business =
db.getCollection("yelp_academic_dataset_business");
       AggregateIterable aggRes = business.aggregate(Arrays.asList(

Aggregates.match(Filters.and(Filters.regex("categories",
"[/.*Hotel.*/i]"),
                                Filters.eq("attributes.Wi-Fi", "free"),
                                Filters.gte("stars", 4.5)
                    )),
                    Aggregates.group("$state", Accumulators.sum("count",
1))
       ));

       for (Object res: aggRes) {
```

```
            System.out.println(res);
        }
    }

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

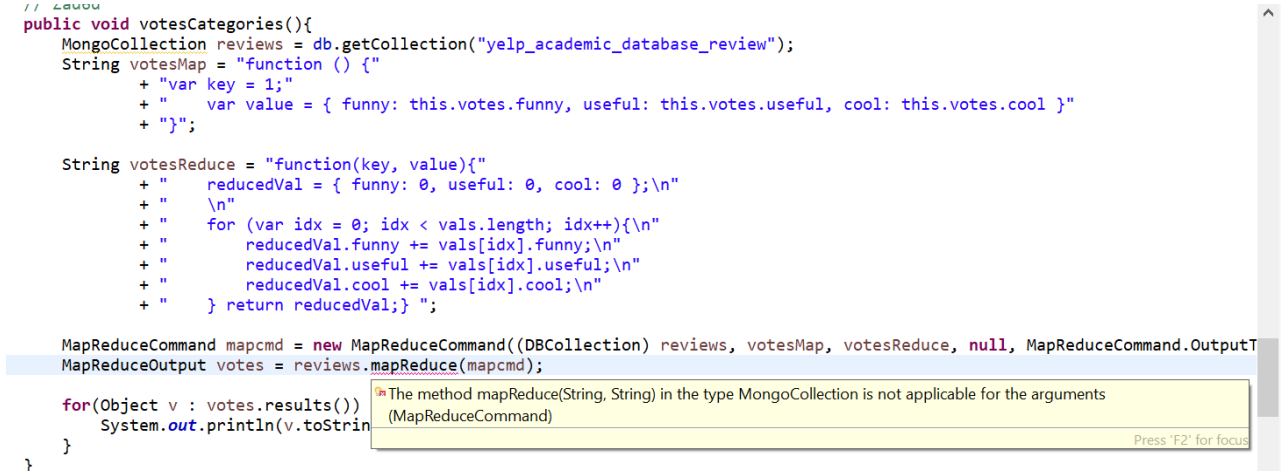
```
Document{{_id=MLN, count=4}}
Document{{_id=AZ, count=363}}
Document{{_id=ELN, count=1}}
Document{{_id=ON, count=13}}
Document{{_id=EDH, count=76}}
Document{{_id=NV, count=167}}
Document{{_id=WI, count=58}}
```

d. Odpowiednik zapytania z punktu 5d.

```
      public void votesCategories(){
       MongoCollection reviews =
db.getCollection("yelp_academic_database_review");
       String votesMap = "function () {"
                   + "var key = 1;"
                   + "  var value = { funny: this.votes.funny, useful:
this.votes.useful, cool: this.votes.cool }"
                   + "}";

       String votesReduce = "function(key, value){"
                   + "  reducedVal = { funny: 0, useful: 0, cool: 0 };\n"
                   + "  \n"
                   + "  for (var idx = 0; idx < vals.length; idx++){\n"
                   + "        reducedVal.funny += vals[idx].funny;\n"
                   + "        reducedVal.useful += vals[idx].useful;\n"
                   + "        reducedVal.cool += vals[idx].cool;\n"
                   + "  } return reducedVal;} ";

       MapReduceCommand mapcmd = new MapReduceCommand((DBCollection)
reviews, votesMap, votesReduce, null,
MapReduceCommand.OutputType.INLINE, null);
       MapReduceOutput votes = reviews.mapReduce(mapcmd);

       for(Object v : votes.results()) {
             System.out.println(v.toString());
```

```
        }
    }
```

Niestety nie udało mi się napisać skutecznego zapytania. Wyświetlał się błąd, informujący, że funkcja MapReduceCommand nie jest możliwa do zrealizowania na MongoCollection:



5. Zadanie 7

Napisałam kod w języku Java (metoda), który zwrócił użytkownika (nazwa użytkownika) o największej liczbie pozytywnych recenzji (ocena co najmniej 4.5).

```
      String bestUser() {
       MongoCollection<Document> review =
db.getCollection("yelp_academic_dataset_review");
       Object userID = review.aggregate(Arrays.asList(
                   Aggregates.match(Filters.gte("stars", 4.5)),
                   Aggregates.group("$user_id",
Accumulators.sum("count", 1)),
                   Aggregates.sort(Sorts.descending("count"))
                   )).first().get("_id");
       MongoCollection<Document> user =
db.getCollection("yelp_academic_dataset_user");
       return user.find(Filters.eq("user_id",
userID.toString())).first().get("name").toString();
    }



    public static void main(String[] args) throws UnknownHostException {
       MongoLab mongoLab = new MongoLab();
       System.out.println("The best user: " + mongoLab.bestUser());
    }
```

```
78
79      String bestUser() {
80          MongoCollection<Document> review = db.getCollection("yelp_academic_dataset_review");
81          Object userID = review.aggregate(Arrays.asList(
82                  Aggregates.match(Filters.gte("stars", 4.5)),
83                  Aggregates.group("$user_id", Accumulators.sum("count", 1)),
84                  Aggregates.sort(Sorts.descending("count"))
85                  )).first().get("_id");
86          MongoCollection<Document> user = db.getCollection("yelp_academic_dataset_user");
87          return user.find(Filters.eq("user_id", userID.toString())).first().get("name").toString();
88      }
89
90
91
92      public static void main(String[] args) throws UnknownHostException {
93          MongoLab mongoLab = new MongoLab();
94          System.out.println("The best user: " + mongoLab.bestUser());
95      }
96
```

Problems Javadoc Declaration Console

```
<terminated> MongoLab [Java Application] C:\Users\basia\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v202102
INFO: Cluster description not yet available. Waiting for 30000 ms before timing out
maj 04, 2021 9:24:28 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:74}] to 127.0.0.1:27017
maj 04, 2021 9:24:28 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=127.0.0.
maj 04, 2021 9:24:28 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:75}] to 127.0.0.1:27017
The best user: Rand
```