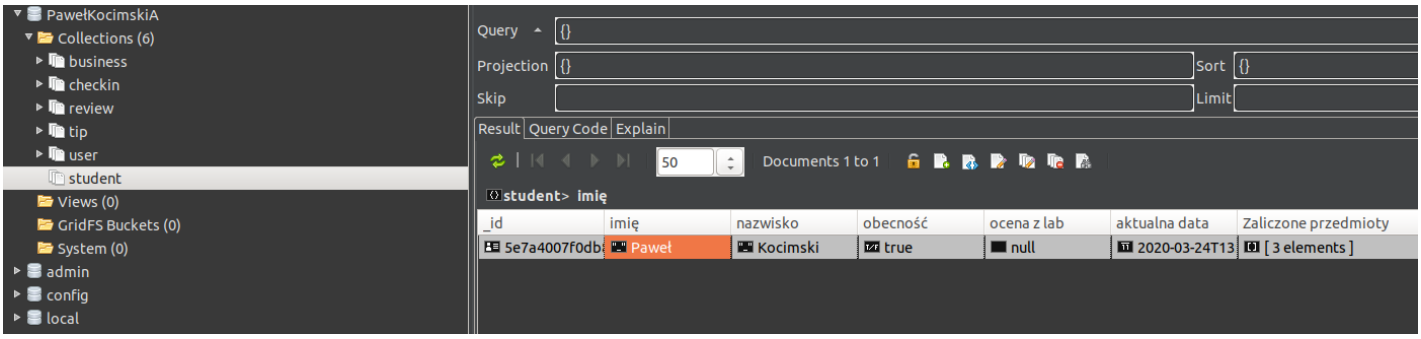
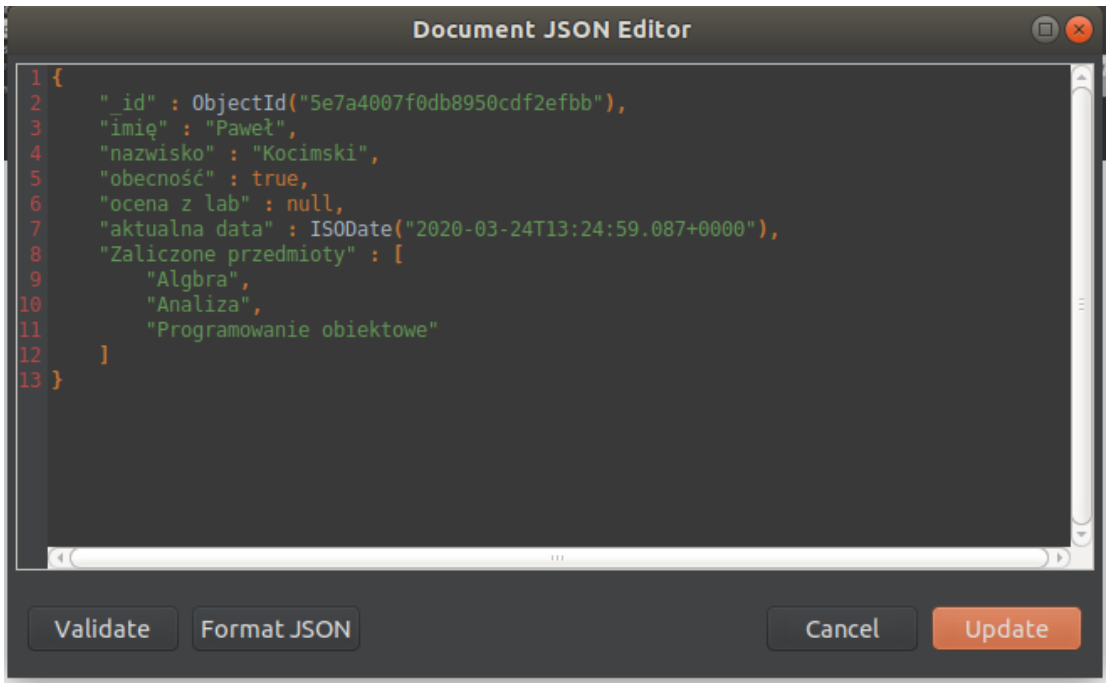
Paweł Kocimski

# Raport z ćwiczeń mongoDB

# 1. Kolekcja Student



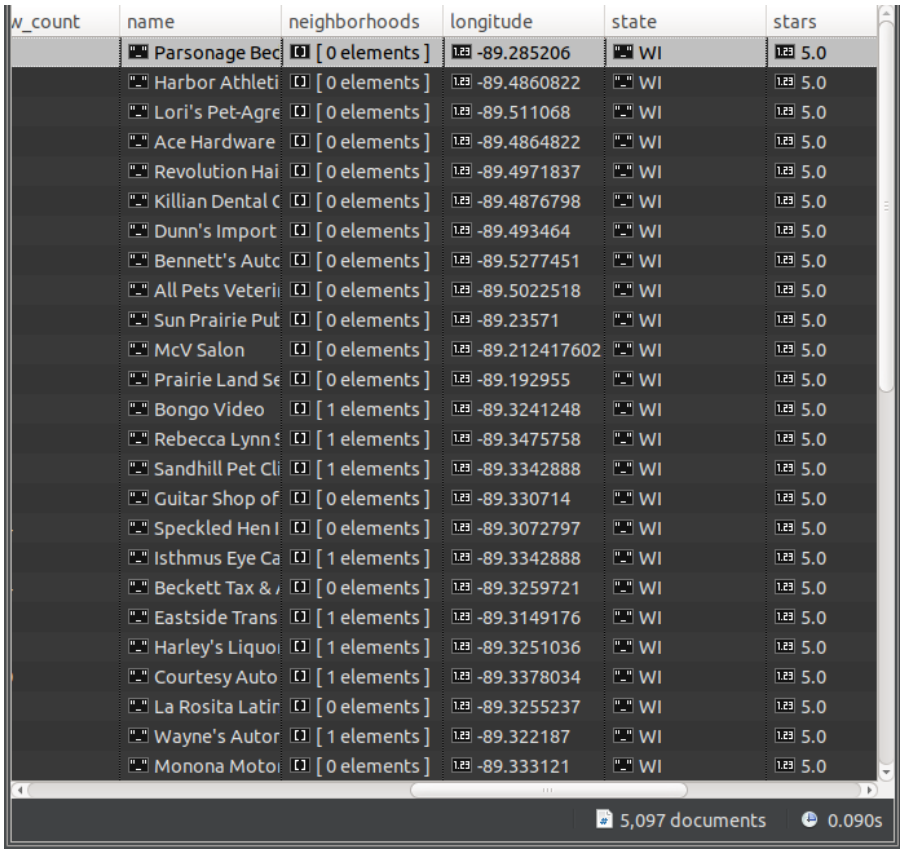




# 2. Zapytania

## a)Ilość miejsc ocenianych na 5 gwiazdek

```
// Requires official MongoShell 3.6+
use PawełKocimskiA;
db.getCollection("business").find(
    {
        "stars" : NumberLong(5)
    }
);

```



| w_count | name | neighborhoods | longitude | state | stars |
|---|---|---|---|---|---|
| | Parsonage Bec | [ 0 elements ] | -89.285206 | WI | 5.0 |
| | Harbor Athleti | [ 0 elements ] | -89.4860822 | WI | 5.0 |
| | Lori's Pet-Agre | [ 0 elements ] | -89.511068 | WI | 5.0 |
| | Ace Hardware | [ 0 elements ] | -89.4864822 | WI | 5.0 |
| | Revolution Hai | [ 0 elements ] | -89.4971837 | WI | 5.0 |
| | Killian Dental C | [ 0 elements ] | -89.4876798 | WI | 5.0 |
| | Dunn's Import | [ 0 elements ] | -89.493464 | WI | 5.0 |
| | Bennett's Autc | [ 0 elements ] | -89.5277451 | WI | 5.0 |
| | All Pets Veteri | [ 0 elements ] | -89.5022518 | WI | 5.0 |
| | Sun Prairie Put | [ 0 elements ] | -89.23571 | WI | 5.0 |
| | McV Salon | [ 0 elements ] | -89.212417602 | WI | 5.0 |
| | Prairie Land Se | [ 0 elements ] | -89.192955 | WI | 5.0 |
| | Bongo Video | [ 1 elements ] | -89.3241248 | WI | 5.0 |
| | Rebecca Lynn S | [ 1 elements ] | -89.3475758 | WI | 5.0 |
| | Sandhill Pet Cli | [ 1 elements ] | -89.3342888 | WI | 5.0 |
| | Guitar Shop of | [ 0 elements ] | -89.330714 | WI | 5.0 |
| | Speckled Hen I | [ 0 elements ] | -89.3072797 | WI | 5.0 |
| | Isthmus Eye Ca | [ 1 elements ] | -89.3342888 | WI | 5.0 |
| | Beckett Tax & A | [ 0 elements ] | -89.3259721 | WI | 5.0 |
| | Eastside Trans | [ 1 elements ] | -89.3149176 | WI | 5.0 |
| | Harley's Liquo | [ 1 elements ] | -89.3251036 | WI | 5.0 |
| | Courtesy Auto | [ 1 elements ] | -89.3378034 | WI | 5.0 |
| | La Rosita Latir | [ 0 elements ] | -89.3255237 | WI | 5.0 |
| | Wayne's Autor | [ 1 elements ] | -89.322187 | WI | 5.0 |
| | Monona Motor | [ 0 elements ] | -89.333121 | WI | 5.0 |

5,097 documents 0.090s

b) Ilość restauracji w każdym mieście, wynik posortowany malejąco

Output> totalRestaurants

| _id | totalRestaurants |
|---|---|
| Las Vegas | 3855.0 |
| Phoenix | 2493.0 |
| Edinburgh | 1049.0 |
| Scottsdale | 1023.0 |
| Mesa | 693.0 |
| Madison | 679.0 |
| Tempe | 672.0 |
| Henderson | 564.0 |
| Chandler | 548.0 |
| Glendale | 422.0 |
| Gilbert | 317.0 |
| Peoria | 221.0 |
| North Las Vega | 198.0 |
| Surprise | 144.0 |
| Goodyear | 119.0 |
| Waterloo | 117.0 |
| Avondale | 100.0 |

```
// Requires official MongoShell 3.6+
use PawełKocimskiA;
db.getCollection("business").aggregate(
    [
        {
            "$match" : {
                "categories" : "Restaurants"
            }
        },
        {
            "$group" : {
                "_id" : "$city",
                "totalRestaurants" : {
                    "$sum" : 1.0
                }
            }
        },
        {
            "$sort" : {
                "totalRestaurants" : -1.0
            }
        }
    ],
    {
        "allowDiskUse" : false
    }
);

```

c) ilość hoteli w każdym stanie/okręgu (state), które posiadają darmowe Wi-fi oraz ocenę co najmniej 4.5 gwiazdki

Output> totalHotels

| _id | totalHotels |
|---|---|
| MLN | 1.0 |
| EDH | 13.0 |
| WI | 10.0 |
| ON | 2.0 |
| AZ | 33.0 |
| NV | 10.0 |

3. Zapytania

a)Ilość miejsc ocenianych na 5 gwiazdek

```
public static void stars5(){

   try (MongoClient client = new MongoClient("localhost", 27017)) {

      MongoDatabase database = client.getDatabase("Pawe\u0142KocimskiA");
      MongoCollection<Document> collection =
database.getCollection("business");


      Document query = new Document();
      query.append("stars", 5L);

      Consumer<Document> processBlock = new Consumer<Document>() {
         @Override
         public void accept(Document document) {
            System.out.println(document);
         }
      };

      collection.find(query).forEach(processBlock);

   } catch (MongoException e) {
      // handle MongoDB exception
   }
}
```

b) Ilość restauracji w każdym mieście, wynik posortowany malejąco

```
public static void restaourantsInCity() {

   try (MongoClient client = new MongoClient("localhost", 27017)) {

      MongoDatabase database = client.getDatabase("Pawe\u0142KocimskiA");
      MongoCollection<Document> collection =
database.getCollection("business");



      Consumer<Document> processBlock = new Consumer<Document>() {
         @Override
         public void accept(Document document) {
            System.out.println(document);
         }
      };
```

```
      List<? extends Bson> pipeline = Arrays.asList(
            new Document()
                  .append("$match", new Document()
                        .append("categories", "Restaurants")
                  ),
            new Document()
                  .append("$group", new Document()
                        .append("_id", "$city")
                        .append("totalRestaurants", new Document()
                              .append("$sum", 1.0)
                        )
                  ),
            new Document()
                  .append("$sort", new Document()
                        .append("totalRestaurants", -1.0)
                  )
      );

      collection.aggregate(pipeline)
            .allowDiskUse(false)
            .forEach(processBlock);

   } catch (MongoException e) {
      // handle MongoDB exception
   }
}
```

c) ilość hoteli w każdym stanie/okręgu (state), które posiadają darmowe Wi-fi oraz ocenę co najmniej 4.5 gwiazdki

```
public static void hotelsInStateWiFiMore45Stars(){

   try (MongoClient client = new MongoClient("localhost", 27017)) {

      MongoDatabase database = client.getDatabase("Pawe\u0142KocimskiA");
      MongoCollection<Document> collection =
database.getCollection("business");


      Consumer<Document> processBlock = new Consumer<Document>() {
         @Override
         public void accept(Document document) {
            System.out.println(document);
         }
      };
```

```
        List<? extends Bson> pipeline = Arrays.asList(
                new Document()
                        .append("$match", new Document()
                                .append("$and", Arrays.asList(
                                        new Document()
                                                .append("categories", "Hotels"),
                                        new Document()
                                                .append("stars", new Document()
                                                        .append("$gte", 4.5)
                                                ),
                                        new Document()
                                                .append("attributes.Wi-Fi", "free")
                                        )
                                )
                        ),
                new Document()
                        .append("$group", new Document()
                                .append("_id", "$state")
                                .append("totalHotels", new Document()
                                        .append("$sum", 1.0)
                                )
                        )
        );

        collection.aggregate(pipeline)
                .allowDiskUse(false)
                .forEach(processBlock);

    } catch (MongoException e) {
        // handle MongoDB exception
    }
}
```

## 4. Użytkownik (nazwa użytkownika) o największej liczbie pozytywnych recenzji

```
public static void bestUser() {

    try (MongoClient client = new MongoClient("localhost", 27017)) {

        MongoDatabase database = client.getDatabase("Pawe\u0142KocimskiA");
        MongoCollection<Document> collection = database.getCollection("user");



        Consumer<Document> processBlock = new Consumer<Document>() {
            @Override
            public void accept(Document document) {
                System.out.println(document);
            }
```

```
      };

      List<? extends Bson> pipeline = Arrays.asList(
            new Document()
                  .append("$match", new Document()
                        .append("average_stars", new Document()
                              .append("$gte", 4.5)
                        )
                  ),
            new Document()
                  .append("$sort", new Document()
                        .append("review_count", -1.0)
                  ),
            new Document()
                  .append("$limit", 1.0)
      );

      collection.aggregate(pipeline)
            .allowDiskUse(false)
            .forEach(processBlock);

   } catch (MongoException e) {
      // handle MongoDB exception
   }
}
```

5. Ilość recenzji posiadają oceny z każdej kategorii: funny, cool, useful

```
6. public static void numberOfRcenzent() {

      try (MongoClient client = new MongoClient("localhost", 27017)) {

         MongoDatabase database = client.getDatabase("Pawe\u0142KocimskiA");
         MongoCollection<Document> collection =
   database.getCollection("user");


         Consumer<Document> processBlock = new Consumer<Document>() {
            @Override
            public void accept(Document document) {
               System.out.println(document);
            }
         };

         List<? extends Bson> pipeline = Arrays.asList(
               new Document()
```

```
                    .append("$group", new Document()
                          .append("_id", new BsonNull())
                          .append("funnyTotal", new Document()
                                .append("$sum", "$compliments.funny")
                          )
                          .append("coolTotal", new Document()
                                .append("$sum", "$compliments.cool")
                          )
                          .append("usefulTotal", new Document()
                                .append("$sum", "$compliments.useful")
                          )
                    )
        );

        collection.aggregate(pipeline)
              .allowDiskUse(false)
              .forEach(processBlock);

    } catch (MongoException e) {
        // handle MongoDB exception
    }
}
```