Programowanie systemów rozproszonych – MongoDB

Opracowano na podstawie materiałów znajdujących się na stronach: https://www.mongodb.com

1 Przykładowa aplikacja

1.1 Pobranie MongoDB

Wersję "MongoDB Commnunity Server" można pobrać ze strony: https://www.mongodb.com/download-center/community Dla systemu operacyjnego Windows dostępne są dwa warianty:

- MSI - wersja z instalatorem,
- ZIP - wersja bez instalatora.

Poniższe wskazóki dotyczą wersji bez instalatora (ZIP). Po pobraniu pliku należy rozpakować do wybranego katalogu (nazwijmy go *MONGODB_HOME*).

1.2 Uruchomienie MongoDB

Należy uruchomić wiersz poleceń, przejść do katalogu *MONGODB_HOME*. Jeżeli nie istnieje katalog gdzie będą przechowywane dane, to należy go utworzyć, np.

```
mkdir data
```

Uruchomienie *mongod* bez włączonej autoryzacji:

```
bin\mongod.exe --dbpath=data
```

Uruchomienie *mongod* z włączoną autoryzacją:

```
bin\mongod.exe --dbpath=data --auth
```

Podczas uruchomienia można wskazać plik konfiguracyjny dodając opcję:

```
--config=mongod.cfg
```

1.3 Użytkownicy

Utworzenie konta administratora:

```
use admin
db.dropUser("MongoDBAdmin")
db.createUser({ user: "MongoDBAdmin", pwd: "MongoDBAdmin", roles: ["userAdminAnyDatabase", "↩
    dbAdminAnyDatabase", "readWriteAnyDatabase"]})
```

Utworzenie konta użytkownika:

```
db.dropUser("student01")
use database01
db.dropDatabase()
use database01
db.createUser(
  {
    user: "student01",
    pwd: "student01",
    roles: [ { role: "readWrite", db: "database01" } ]
  }
)
```

1.4 Przykład

```
1  package pl.kielce.tu.mongodb;
2  import static com.mongodb.client.model.Filters.and;
3  import static com.mongodb.client.model.Filters.elemMatch;
4  import static com.mongodb.client.model.Filters.eq;
5  import static com.mongodb.client.model.Filters.exists;
6  import static com.mongodb.client.model.Filters.gt;
7  import static com.mongodb.client.model.Filters.lt;
8  import static com.mongodb.client.model.Filters.or;
9  import static com.mongodb.client.model.Projections.include;
10 import static com.mongodb.client.model.Updates.inc;
11
12 import java.util.ArrayList;
13 import java.util.Arrays;
14 import java.util.List;
15
16 import org.bson.Document;
17
18 import com.mongodb.MongoClient;
19 import com.mongodb.MongoClientURI;
20 import com.mongodb.client.MongoCollection;
21 import com.mongodb.client.MongoDatabase;
22 import com.mongodb.client.result.DeleteResult;
23 import com.mongodb.client.result.UpdateResult;
24
25 public class TestMongoDB {
26     public static void main(String[] args) {
27
28         String user = "student01";
29         String password = "student01";
30         String host = "localhost";
31         int port = 27017;
32         String database = "database01";
33
34         String clientURI = "mongodb://" + user + ":" + password + "@" + host + ":" + port + "/" + ↩
               database;
35         MongoClientURI uri = new MongoClientURI(clientURI);
36
37         MongoClient mongoClient = new MongoClient(uri);
38
39         MongoDatabase db = mongoClient.getDatabase(database);
40
41         db.getCollection("people").drop();
42
43         MongoCollection<Document> collection = db.getCollection("people");
44
45         Document nowak = new Document("_id", 1)
46                 .append("lastname", "Nowak")
47                 .append("names", "Jan")
48                 .append("age", 21)
49                 .append("grades", Arrays.asList(new Document("programming", 5.0), new Document("↩
                       mathematics", 4.0), new Document("physics", 3.0)));
50         collection.insertOne(nowak);
51
52         Document polak = new Document("_id", 2)
53                 .append("lastname", "Polak")
54                 .append("names", Arrays.asList("Piotr","Adam"))
55                 .append("age", 22)
56                 .append("grades", new Document("programming", 4.5).append("mathematics", 4.0).↩
                       append("physics", 3.5));
```

```
57          collection.insertOne(polak);
58
59          List<Document> documents = new ArrayList<Document>();
60          for (int i = 0; i < 2; i++)
61              documents.add(new Document("_id", 10 + i));
62          collection.insertMany(documents);
63
64          Document first = collection.find().first();
65          System.out.println("find().first() " + first.toJson());
66
67          for (Document doc : collection.find())
68              System.out.println("find() " + doc.toJson());
69
70          Document myDoc = collection.find(lt("_id", 2)).first();
71          System.out.println("lt(\"_id\", 2) " + myDoc.toJson());
72
73          for (Document d : collection.find(or(
74                  eq("grades.programming", 5.0),
75                  eq("grades.programming", 4.5))))
76          System.out.println("or(eq(\"grades.programming\", 5.0),eq(\"grades.programming\", 4.5)) " ↩
                + d.toJson());
77
78          for (Document d : collection.find(or(
79                  eq("grades", Document.parse("{programming : 5.0}")),
80                  eq("grades", Document.parse("{programming : 4.5}")))))
81          System.out.println("or(eq(\"grades\", Document.parse(\"{programming : 5.0}\")),eq(\"grades↩
                \", Document.parse(\"{programming : 4.5}\"))) " + d.toJson());
82
83          for (Document d : collection.find(or(
84                  elemMatch("grades", Document.parse("{programming : 5.0}")),
85                  elemMatch("grades", Document.parse("{programming : 4.5}")))))
86              System.out.println("find(or(elemMatch(\"grades\", Document.parse(\"{programming : ↩
                    5.0}\")),elemMatch(\"grades\", Document.parse(\"{programming : 4.5}\"))) " + d.↩
                    toJson());
87
88          for (Document d : collection.find(exists("names.1")))
89              System.out.println("find(exists(\"names.1\")) " + d.toJson());
90
91          for (Document d : collection.find(exists("grades.programming", false)))
92              System.out.println("find(exists(\"grades.programming\", false)) " + d.toJson());
93
94          for (Document doc : collection.find().projection(include("firstname", "names")))
95              System.out.println("find().projection(include(\"firstname\", \"names\")) " + doc.toJson↩
                    ());
96
97          for (Document doc : collection.find(and(exists("lastname", true), exists("names", true))).↩
                projection(include("firstname", "names")))
98              System.out.println("find(and(exists(\"firstname\", true), exists(\"name\", true))).↩
                    projection(include(\"firstname\", \"names\")) " + doc.toJson());
99
100         for (Document doc : collection.find().sort(new Document("_id", -1)))
101             System.out.println("find().sort(new Document(\"_id\", -1))) " + doc.toJson());
102
103         for (Document doc : collection.find().sort(new Document("_id", -1)).limit(2))
104             System.out.println("find().sort(new Document(\"_id\", -1)).limit(2) " + doc.toJson());
105
106         collection.updateOne(eq("_id", 10), new Document("$set", new Document("lastname", "Kowal")↩
                .append("firstName", "Adam")));
107         for (Document doc : collection.find())
108             System.out.println("updateOne(eq(\"_id\", 10), new Document(\"$set\", new Document(\"↩
                    lastname\", \"Kowal\").append(\"firstName\", \"Adam\")) " + doc.toJson());
109
```

```
110         UpdateResult updateResult = collection.updateMany(exists("age"), inc("age", 1));
111         System.out.println(updateResult.getModifiedCount());
112         for (Document doc : collection.find())
113             System.out.println("updateMany(exists(\"age\"), inc(\"age\", 1)) " + doc.toJson());
114 
115         collection.deleteOne(eq("_id", 11));
116         for (Document doc : collection.find())
117             System.out.println("deleteOne(eq(\"_i\", 11)) " + doc.toJson());
118 
119         DeleteResult deleteResult = collection.deleteMany(gt("_id", 0));
120         System.out.println(deleteResult.getDeletedCount());
121         for (Document doc : collection.find())
122             System.out.println("deleteMany(gt(\"_id\", 0)) " + doc.toJson());
123 
124         mongoClient.close();
125     }
126 }
```

Przykład 1: src/LabMongoDB/src/main/java/pl/kielce/tu/mongodb/TestMongoDB.java {link}

1.5 Kompilacja

```
1 mvn clean compile
```

1.6 Uruchomienie

```
1 mvn exec:java
```