Modern Adatbázis Rendszerek Gyakorlat

MongoDB Feladat megoldások

1. Alap műveletek

a, Hozzunk létre egy mdbgyak adatbázist.

MognodDB Compassban 1 gombnyomás.

b, Készítsünk egy auto kollekciót mely rendelkezik egy típus, szín, ár és gyártás_év adattagokkal.

```
db.createCollection("auto") use auto
```

c, Vegyünk fel az auto táblába egy, majd egyszerre több rekordot is.

1., Egy rekord felvétele:

```
db.auto.insert({'tipus' : 'Opel', 'szin' : 'piros' , 'ár' : 500011, 'gyév':2001});
db.auto.insert({'tipus' : 'Suzuki', 'szin' : 'sárga' , 'ár' : 600000, 'gyév':2005, "állapot" : "jó"});
db.auto.find();
```

2 , Töb rekord felvétele:

```
db.auto.insertMany([{'tipus' : 'Suzuki', 'szin' : 'sárga' , 'ár' : 600000, 'gyév':2005} ,
        {'tipus' : 'Toyota', 'szin' : 'fehér' , 'ár' : 1600000, 'gyév':2010},
        {'tipus' : 'Wolkswagen', 'szin' : 'zöld' , 'ár' : 904000, 'gyév':2007},
        {'tipus' : 'Audi', 'szin' : 'kék' , 'ár' : 3220000, 'gyév':2014},
        {'tipus' : 'Opel', 'szin' : 'piros' , 'ár' : 500011, 'gyév':2001}] );
```

d, Írassuk ki azoknak az autóknak a típusát, amelyeknek az ára több mint 1.000.000 és számoljuk meg őket

```
db.auto.find({'ár' : {'$gt': 1000000}},{'tipus' : 1 , _id : 0}) // kiírat
db.auto.find({'ár' : {'$gt': 1000000}},{'tipus' : 1 , _id : 0}).count();  //megszámol
```

e, A már létező rekordokhoz adjunk hozzá egy új mezőt „állapot” néven és állítsuk „jó”-ra.

```
db.auto.updateMany ({},{"$set":{"állapot": "jó"}})
```

f, A 2004 előtt gyártott autók állapotát állítsuk sérültre.

```
db.auto.updateMany ({"gyév" : {"$lt" : 2004}},{"$set":{"állapot": "sérült"}})
```

g, A sérült állapotú autó rekordokat töröljük ki a táblából.

```
db.auto.deleteMany({"állapot" : "sérült"});
```

//Több autó felvétele

```
db.auto.insertMany([{'tipus' : 'Suzuki', 'szin' : 'sárga' , 'ár' : 600000, 'gyév':2005} ,
                {'tipus' : 'Toyota', 'szin' : 'fehér' , 'ár' : 1600000, 'gyév':2010},
                {'tipus' : 'Wolkswagen', 'szin' : 'zöld' , 'ár' : 904000, 'gyév':2007},
                {'tipus' : 'Audi', 'szin' : 'kék' , 'ár' : 3220000, 'gyév':2014},
                {'tipus' : 'Opel', 'szin' : 'piros' , 'ár' : 500011, 'gyév':2001}] )

db.auto.insertMany([{'tipus' : 'Opel', 'szin' : 'sárga' , 'ár' : 600000, 'gyév':2005} ,{'tipus' : 'Opel', 'szin' :
'fehér' , 'ár' : 1600000, 'gyév':2010},
                {'tipus' : 'Wolkswagen', 'szin' : 'zöld' , 'ár' : 904000, 'gyév':2007},
                {'tipus' : 'Toyota', 'szin' : 'kék' , 'ár' : 3220000, 'gyév':2014},
                {'tipus' : 'Toyota', 'szin' : 'piros' , 'ár' : 500011, 'gyév':2001}])
db.auto.find()
```

2. Bonyolultabb műveletek

a, készítsünk egy tulajdonos kollekciót mely név és kor mezőkkel rendelkezik.

```
db.createCollection(„tulajdonos”);
```

b, Készítsünk egy tárolt függvényt, amely új tulajdonosokat tud felvenni a tulajdonos kollekcióba, úgy, hogy az ID-jét is mi adjuk meg.

```
db.system.js.save(
   {
   _id : "save_tulaj",
   value: function(id,nev, kor)
   {
      db.tulajdonos.insert({"_id": id, "név": nev, "kor": kor})
     }
  });

db.loadServerScripts();

save_tulaj("T5","Pál", 20);

db.tulajdonos.find()
```

c, Készítsünk tárolt függvényt „save_auto” néven, amellyel új autó rekordot tudunk felvenni az auto kollekcióba és adjunk hozzá egy tulaj mezőt is mely a tuljdonos tábla elemeinek az ID-jét tartalmazza.

```
db.system.js.save(
   {
   _id : "save_auto",
   value: function(tipus, szin, ar, gyart_ev, allapot , tulaj )
      {
     db.auto.insertOne({"tipus": tipus, "szin": szin, "ár": ar, "gyév": gyart_ev, "állapot": allapot, "tulaj":
tulaj})
     }
  }
);
  db.loadServerScripts();

save_auto("auto2","fehér",3241,2010, "jó", "T5");
```

d, Készítsünk tárolt függvényt „getTulajByName” néven, amely a paraméterként kapott tulajdonos neve alapján kilistázza az összes olyan nevű tulajdonost.

```
db.system.js.save(
   {
   _id : "getTulajByName",
   value: function(nev)
   {
      var n = db.tulajdonos.find({"név": nev})        while(n.hasNext()){
         print(n.next());
         }
     }
  });
```

e, Írassuk ki az összes autót, amely „Pál” -hoz kapcsolódik.

```
var oi = db.tulajdonos.findOne({"név":"Pál"})

db.auto.find({"tulaj":oi._id})
```

f, Csökkentsük a sérült állapotú autók árát 300.000 -el.

```
db.auto.find().forEach(     function(obj){
         if(obj.állapot == "sérült"){
               db.auto.update({_id: obj._id}, {$inc : {'ár': -300000}});
           }
     }
);
db.auto.find();
```

g, A $where használatával, számoljuk meg azokat az autókat, amelyeknek az ára kisebb mint 1.000.000. és a gyártás éve 2010 előtti.

```
  db.auto.find({"$where" : function(){       if(this.ár < 1000000 && this.gyév < 2010 )          return
true;       else
         return false;
   }}).count()
```

h, Vegyünk fel még néhány auto rekordot, és írassuk ki típus szerint csoportosítva az átlag árat.

```
db.auto.insertMany([{"tipus": "BMW" , "szin" : "zöld", "ár" : 12000000, "gyév" : 2008, "állapot" : "sérült"},
        {"tipus": "Audi" , "szin" : "fekete", "ár" : 4001100, "gyév" : 2007, "állapot" : "jó"},
        {"tipus": "Opel" , "szin" : "zöld", "ár" : 12000000, "gyév" : 2008, "állapot" : "sérült"},
        {"tipus": "Toyota" , "szin" : "Fehér", "ár" : 4221100, "gyév" : 2000, "állapot" : "sérült"},
       {"tipus": "BMW" , "szin" : "zöld", "ár" : 6001100, "gyév" : 2008, "állapot" : "jó"}]);


db.auto.aggregate([
    {$group : {"_id":"$tipus","avgp":{"$avg" : "$ár"} }},
    {$sort : {"avgp": -1}}

        ]);
```

i, Írassuk ki csökkenő sorrendben, hogy típusonként mennyi autó rendelkezik sérült státusszal.

```
db.auto.aggregate([
    {$match : {"állapot" : "sérült"}},
    {$group : {"_id": "$tipus" , "sérültek száma" : {"$sum" : 1 }}},        {$sort : {"száma" : -1}}

]);
```

3. JAVA rész következik.

a, Először is hozzunk létre egy MAVEN projektet és egy Teszt osztály az mdbgyak csomagban.

A pom.xml-be illesszük be a

```
<dependency>
        <groupId>org.mongodb</groupId>
        <artifactId>mongo-java-driver</artifactId>
        <version>3.12.10</version>
</dependency>
```

Függőséget.

b, Kapcsolódjunk az mdbgyak adatbázishoz és jelöljük ki az auto kollekciót.

```
MongoClient mongo  = new MongoClient("localhost", 27017);
MongoDatabase database = mongo.getDatabase("mdbgyak");
//Kijelöli az auto kollekciót.
MongoCollection<Document> collection = database.getCollection("auto");
```

c, Listázzuk ki az auto kollekció rekordjait.

```
FindIterable<Document> iterDoc = collection.find();
        int i = 1;
        // Getting the iterator
        Iterator it = iterDoc.iterator();
        while (it.hasNext()) {
            System.out.println(it.next());
             i++;
        }
```

d, Vegyünk fel egy új elemet az auto kollekcióba. Típusa legyen JavaInserted (insertOne())

```
Document document = new Document("tipus", "javaInserted")
       .append("szin", "kék")
       .append("ár", 2000110)
       .append("gyév", 2009)
       .append("állapot", "jó");


       collection.insertOne(document);
```

e, Vegyünk fel egyszerre több elemet az auto kollekcióba. (insertMany())

```
Document document1 = new Document("tipus", "javaInserted")
                     .append("szin", "kék")
                     .append("ár", 2000110)
                     .append("gyév", 2009)
                     .append("állapot", "jó");


Document document2 = new Document("tipus", "javaInserted2")
                     .append("szin", "zöld")
                     .append("ár", 1223)
                     .append("gyév", 2009)
                     .append("állapot", "sérült");
```

g, A JavaInserted típusú autók állapotát állítsuk sérültre.

```
collection.updateMany(Filters.eq("tipus", " javaInserted "),
Updates.set("állapot", "sérült"));
```

h, Töröljük a JavaInserted típusú rekordokat.

```
collection.deleteMany(Filters.eq("tipus", "javaInserted"));
```

Teljes kód:

```
package mdbgyak;

import com.mongodb.MongoClient;
import com.mongodb.client.FindIterable;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.model.Filters;
import com.mongodb.client.model.Updates;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import org.bson.Document;

public class Teszt {

	public static void main(String[] args) {

		 MongoClient mongo  = new MongoClient("localhost", 27017);
		 MongoDatabase database = mongo.getDatabase("mdbgyak");
		 System.out.println("Connected to the database successfully");

		//Kijelöli az auto kollekciót.
		 MongoCollection<Document> collection =
		database.getCollection("auto");
		System.out.println("Collection auto selected successfully");

		//Kilistáz mindent
		 FindIterable<Document> iterDoc = collection.find();
		 int i = 1;

		 Iterator it = iterDoc.iterator();
		 while (it.hasNext()) {
		    System.out.println(it.next());
		     i++;
		  }


//Beilleszt egy elemet

		Document document = new Document("tipus", "javaInserted")
		                .append("szin", "kék")
		                .append("ár", 2000110)
		                .append("gyév", 2009)
		                .append("állapot", "jó");

		                //Inserting document into the collection
		collection.insertOne(document);
		System.out.println("Document inserted successfully");
```

```
//több dokumentum beillesztése kollekcióba.
            Document document1 = new Document("tipus", "javaInserted")
                        .append("szin", "kék")
                        .append("ár", 2000110)
                        .append("gyév", 2009)
                        .append("állapot", "jó");


            Document document2 = new Document("tipus", "javaInserted2")
                        .append("szin", "zöld")
                        .append("ár", 1223)
                        .append("gyév", 2009)
                        .append("állapot", "sérült");


            List<Document> list = new ArrayList<Document>();
            list.add(document1);
            list.add(document2);
            collection.insertMany(list);



//update

           collection.updateMany(Filters.eq("tipus", "javaInserted"),
           Updates.set("állapot", "sérült"));
           System.out.println("Document updated successfully...");


//Törlés DB-ből
            collection.deleteMany(Filters.eq("tipus", "javaInserted"));
            System.out.println("Document deleted successfully...");

            }

        }

}
```