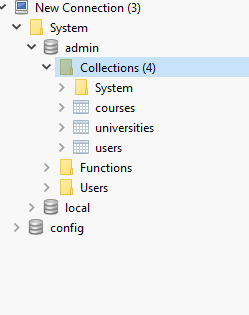
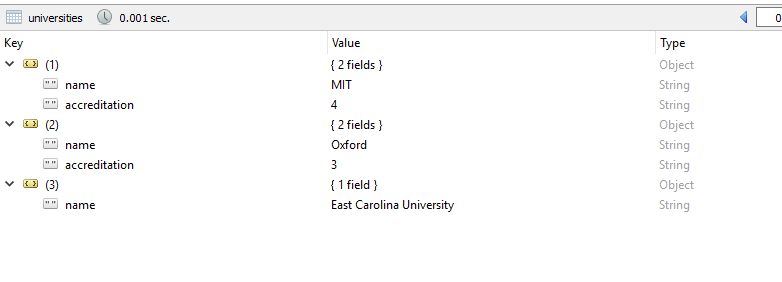
**Task**

1. Install MongoDB Server
2. Download MongoDB - <https://www.mongodb.com/download-center#community>
3. Install it
4. Create C:\data\db
5. Run mongod.exe via CMD or PowerShell (as administrator)
6. Download Robo 3T - <https://robomongo.org>
7. Connect to your local MongoDB
8. Install Robo 3T
9. Create three collections:
   1. “**universities**”;
   2. “**users**”;
   3. ” **courses**”



1. Insert data into the created collections (use scripts from InsertsToDB.docx)
2. Write next queries:
   1. Select all universities. Show name and accreditation

db.universities.find ({},{name:1, accreditation:1,\_id:0})



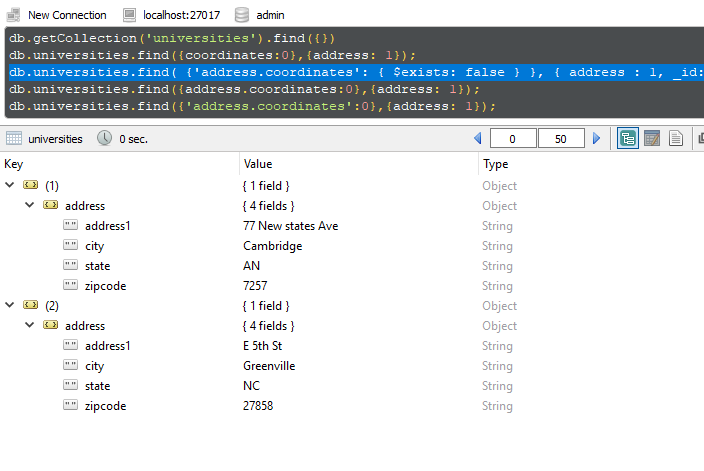
* 1. **Select university without coordinates. Show only Address information.**

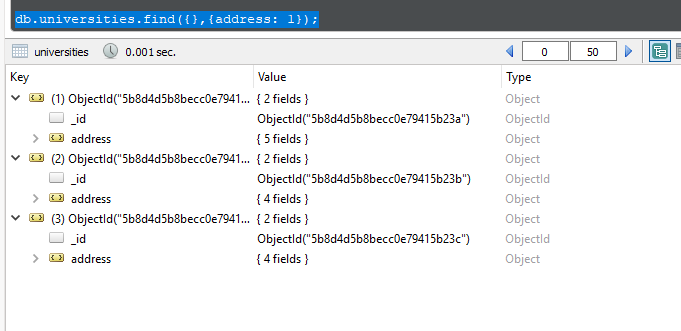
db.universities.find({},{address: 1});

b. Select university without coordinates. Show only Address information.

*не верно. выбирает так же университет у которого есть координаты*

db.universities.find( {'address.coordinates': { $exists: false } }, { address : 1, \_id: 0} );





c**. Select university with State = “MA” and zipcode not equal to “27897”. Show id, name, state, zipcode**

c. Select university with State = “MA” and zipcode not equal to “27897”. Show id, name, state, zipcode

*ошибка - выводит так же координаты*

db.universities.find(

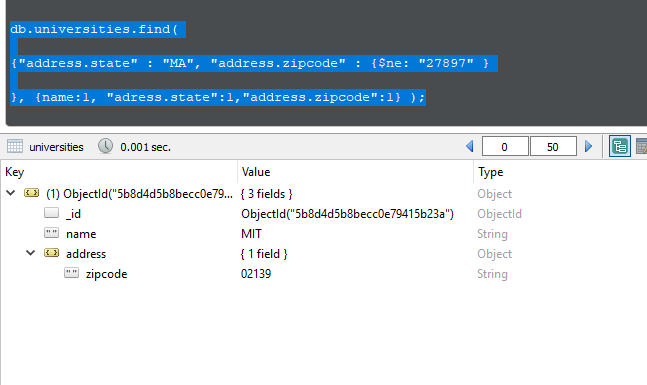
{"address.state" : "MA", "address.zipcode" : {$ne: "27897" }

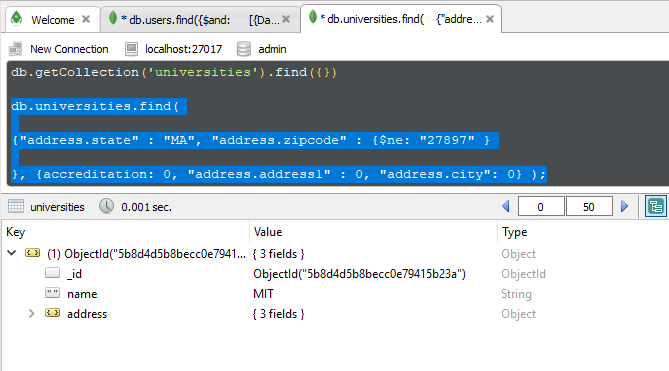
}, {accreditation: 0, "address.address1" : 0, "address.city": 0} );

db.universities.find(

{"address.state" : "MA", "address.zipcode" : {$ne: "27897" }

}, {name:1, "adress.state":1,"address.zipcode":1} );





**d. Select users with Date of Birth more than (>) 1980 year and less than current date. Show only Date of Birth**

d. Select users with Date of Birth more than (>) 1980 year and less than current date. Show only Date of Birth

*ошибка- выводит так же ID*

db.users.find({$and:

[{DateOfBirth: {$gt: ISODate("1980-12-31T00:00:00.000Z")}},

{DateOfBirth: {$lt:new Date()}}]},

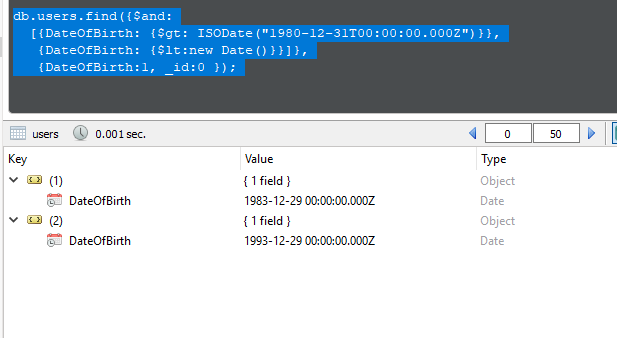
{DateOfBirth:1});

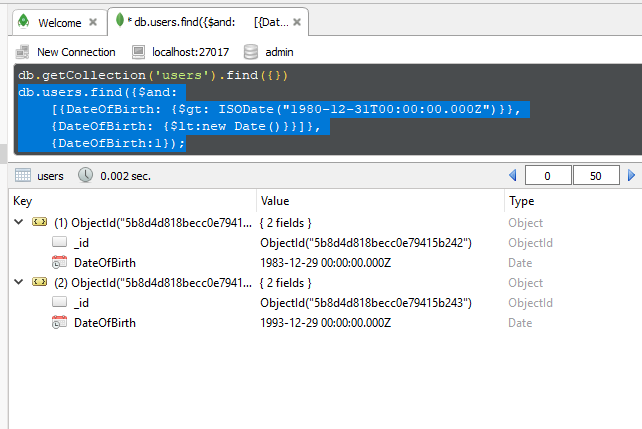
db.users.find({$and:

[{DateOfBirth: {$gt: ISODate("1980-12-31T00:00:00.000Z")}},

{DateOfBirth: {$lt:new Date()}}]},

{DateOfBirth:1, \_id:0 });





**e. Change course name and delete all other information (university\_id, users) for course, which contains only students.**

db.courses.updateMany({"users.role" : "student", "users.role" : {$eq: "student"}}, { $set : {name : "Updated Course"}}) e. Change course name and delete all other information (university\_id, users) for course, which contains only students

*не верно - изменило имя курсе везде где есть студенты, не удалило остальную информацию.*

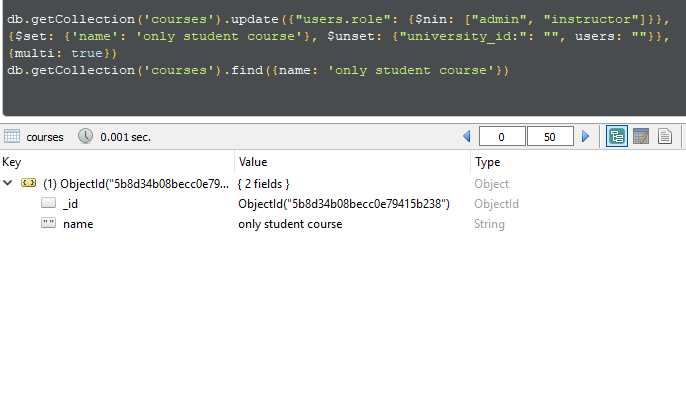
*(должно было изменить и удалить только там где  все users = students )*

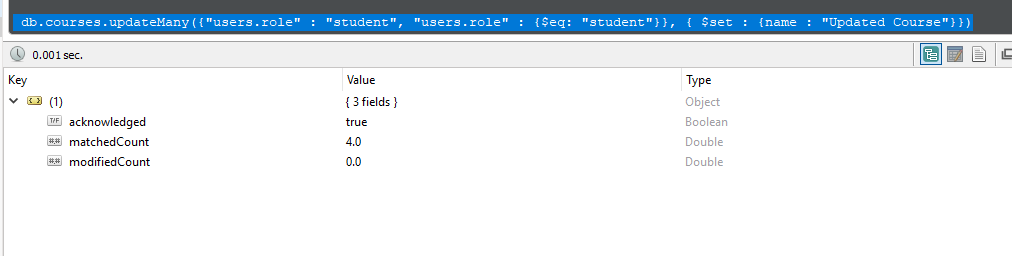
db.getCollection('courses').update({"users.role": {$nin: ["admin", "instructor"]}},

{$set: {'name': 'only student course'}, $unset: {"university\_id:": "", users: ""}},

{multi: true})

db.getCollection('courses').find({name: 'only student course'})

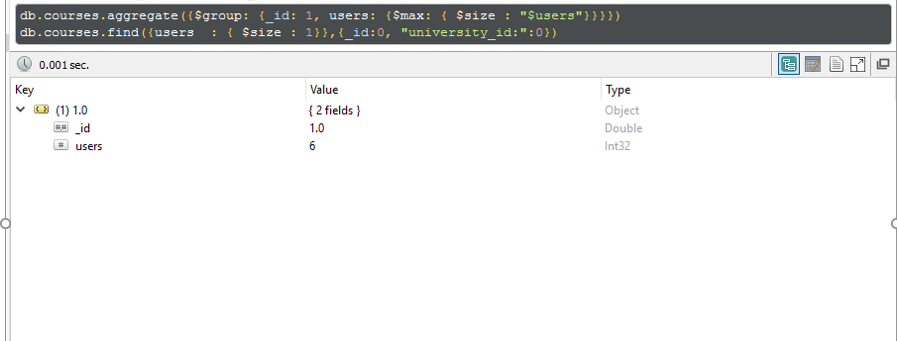




1. **Select courses with max number of users. Show course name, user roles, amount of users.**

db.courses.aggregate({$group: {\_id: 1, users: {$max: {$size : "$users"}}}})

db.courses.find({users : {$size : 1}},{\_id:0, "university\_id:":0})



**g. Select user with the longest MiddleName. Show \_id, MiddleName, length**.

db.users.aggregate({ $group : { \_id: "\_id", maxLenth: { $max : "$MiddleName"}}});

db.users.aggregate([{$project: {"length": { $strLenCP: "Polyakova" }}},{$limit: 1 },])

g. Select user with the longest MiddleName. Show \_id, MiddleName, length.

*не запускается*

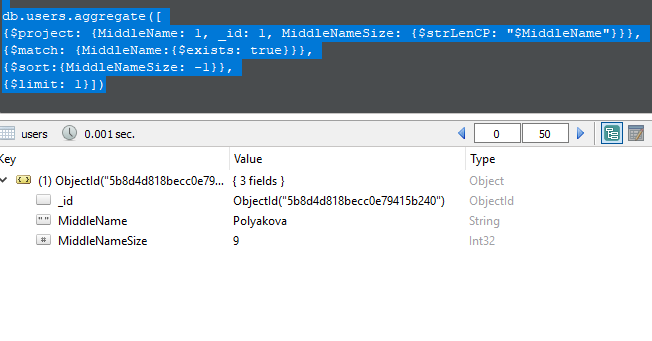
db.users.aggregate([

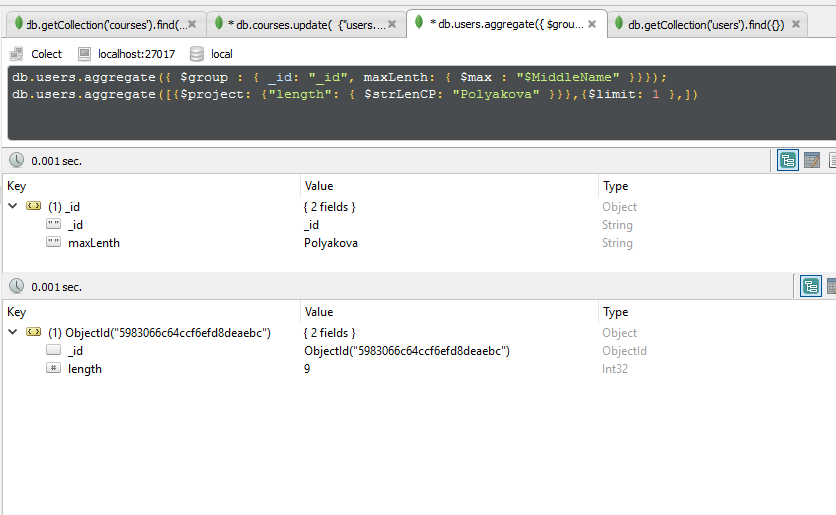
{$project: {MiddleName: 1, \_id: 1, MiddleNameSize: {$strLenCP: "$MiddleName"}}},

{$match: {MiddleName:{$exists: true}}},

{$sort:{MiddleNameSize: -1}},

{$limit: 1}])





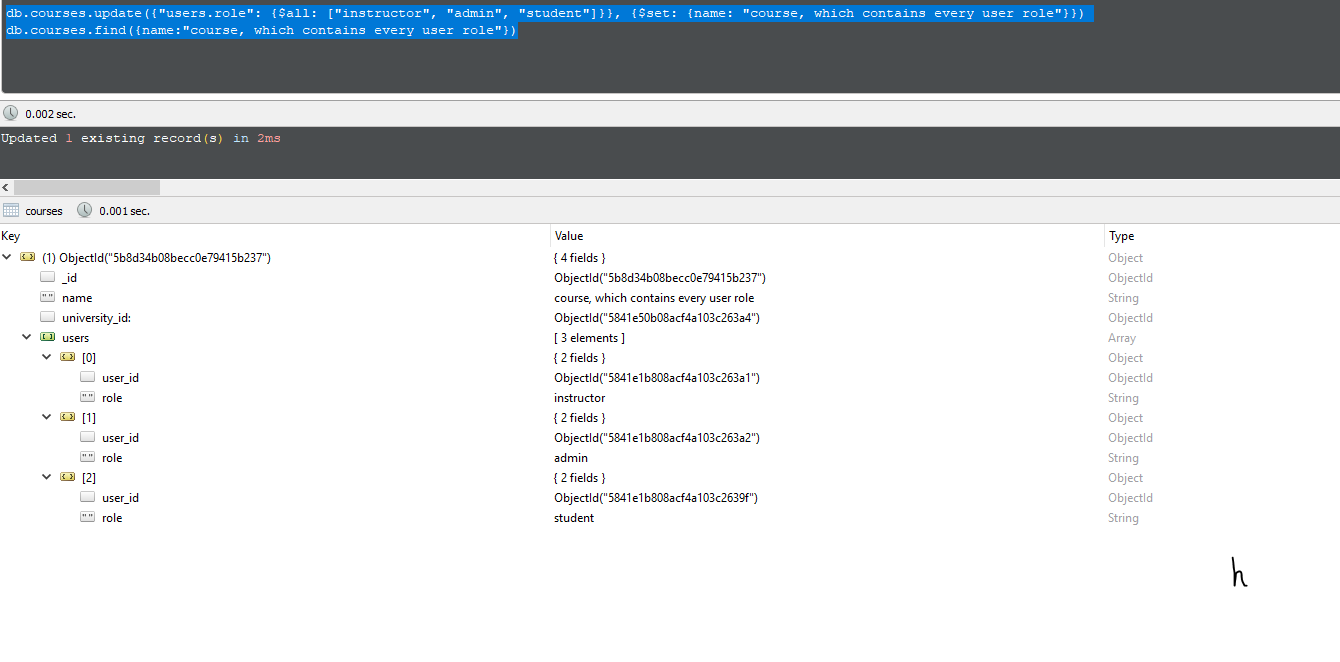
**h. Update *only* the course name, which contains every user role.**

h. Update only the course name, which contains every user role.

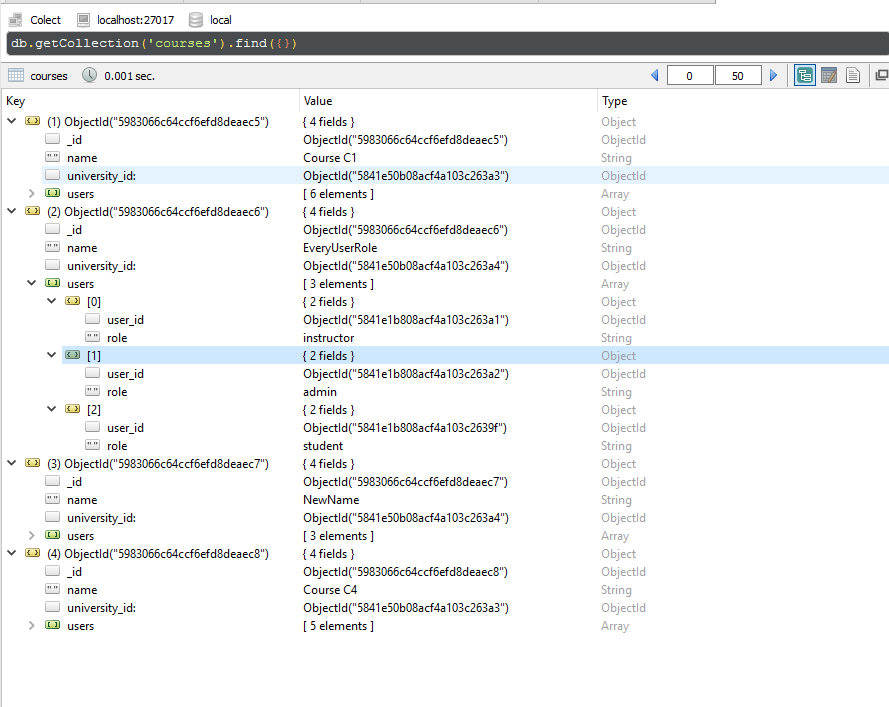
*не верно - Выводит курс №3, там где все только студенты   
(внимательнее чистай условие)*

db.courses.update({"users.role": {$all: ["instructor", "admin", "student"]}}, {$set: {name: "course, which contains every user role"}})

db.courses.find({name:"course, which contains every user role"})



* 1. db.getCollection('courses').find({})

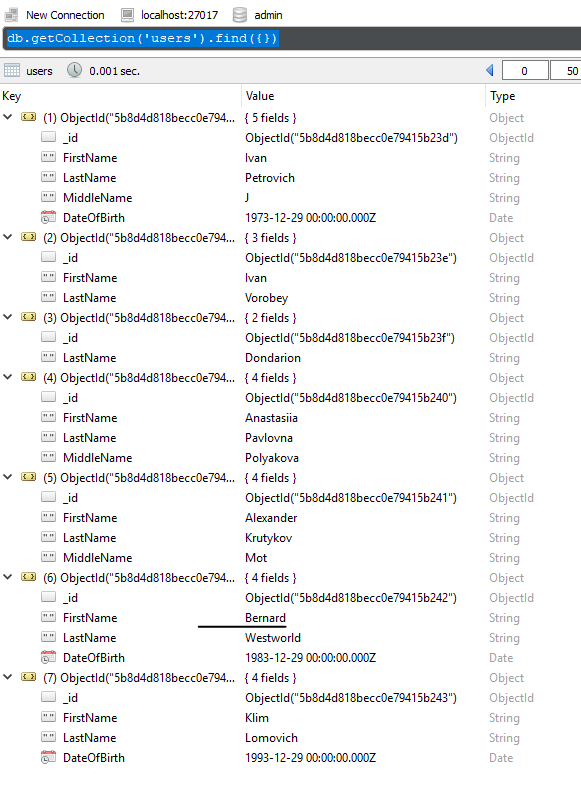
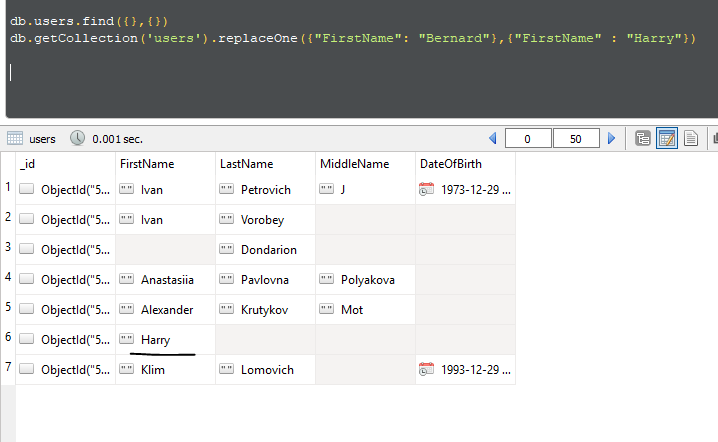


* + 1. Replace User document with Name = “Pavel” (insert your values and pay attention for the new structure. Use .find() before replacement and compare values after)

db.getCollection('users').find({})

db.users.find({},{})

db.getCollection('users').replaceOne({"FirstName": "Bernard"},{"FirstName" : "Harry"})



i. - *не вижу скрипта в таске*

1. Delete user which has only LastName by 2 ways (use delete()and remove() commands)(i.e. MiddleName, Date of Birth, First Name are null).

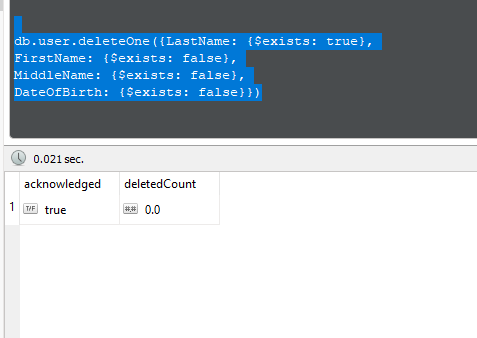
j. - *не вижу скрипта в таске, только скриншоты. не могу проверить*

db.user.deleteOne({LastName: {$exists: true},

FirstName: {$exists: false},

MiddleName: {$exists: false},

DateOfBirth: {$exists: false}})



db.user.remove({LastName: {$exists: true},

FirstName: null,

MiddleName: null,

DateOfBirth: {$exists: false}})

