**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

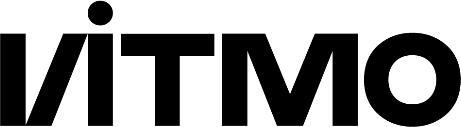
**Отчет**

по лабораторной работе №6 «Работа с БД в СУБД MongoDB» по дисциплине **«Проектирование и реализация баз данных»**

Автор: Андрей Федак

Факультет: ИКТ Группа: К3239

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

# Оглавление

[Практическое задание 2.1.1 4](#_bookmark0)

[Практическое задание 2.2.1: 5](#_bookmark1)

[Практическое задание 2.2.2: 7](#_bookmark2)

[Практическое задание 2.2.3: 7](#_bookmark3)

[Практическое задание 2.2.4: 8](#_bookmark4)

[Практическое задание 2.3.1: 8](#_bookmark5)

[Практическое задание 2.3.2: 9](#_bookmark6)

[Практическое задание 2.3.3: 9](#_bookmark7)

[Практическое задание 2.3.4: 10](#_bookmark8)

[Практическое задание 3.1.1: 11](#_bookmark9)

[Практическое задание 3.1.2: 13](#_bookmark10)

[Практическое задание 3.2.1: 14](#_bookmark11)

[Практическое задание 3.2.2: 14](#_bookmark12)

[Практическое задание 3.2.3: 14](#_bookmark13)

[Практическое задание 3.3.1: 15](#_bookmark14)

[Практическое задание 3.3.2: 16](#_bookmark15)

[Практическое задание 3.3.3: 17](#_bookmark16)

[Практическое задание 3.3.4: 18](#_bookmark17)

[Практическое задание 3.3.5: 19](#_bookmark18)

[Практическое задание 3.3.6: 20](#_bookmark19)

[Практическое задание 3.3.7: 21](#_bookmark20)

[Практическое задание 3.4.1: 22](#_bookmark21)

[Практическое задание 4.1.1: 23](#_bookmark22)

[Практическое задание 4.2.1: 24](#_bookmark23)

[Практическое задание 4.4.1: 26](#_bookmark24)

[Вывод 28](#_bookmark25)

**Цель:** овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

**Оборудование**: компьютерный класс.

**Программное обеспечение**: СУБД MongoDB 4+, 6.0.6 (текущая).

# Практическое задание 2.1.1:

1. *Создайте базу данных learn.*



1. *Заполните коллекцию единорогов unicorns:*

db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});

db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});

db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires:

99});

db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});

db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});

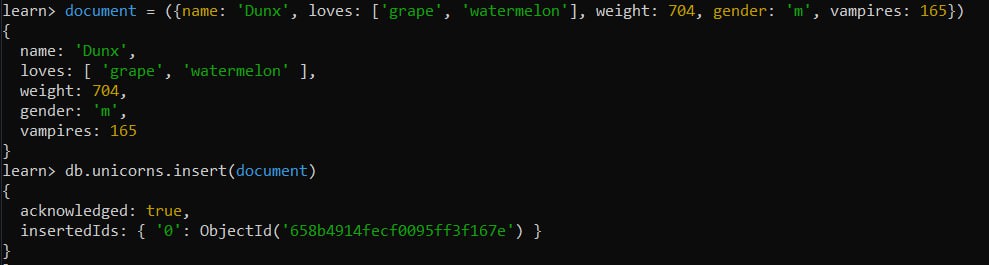
db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});

db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});

db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});

db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});

1. *Используя второй способ, вставьте в коллекцию единорогов документ:*



1. *Проверьте содержимое коллекции с помощью метода find.*



## Практическое задание 2.2.1:

* 1. *Сформируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.*

Самцы:

db.unicorns.find({gender: "m"}).limit(3).sort({name: 1})



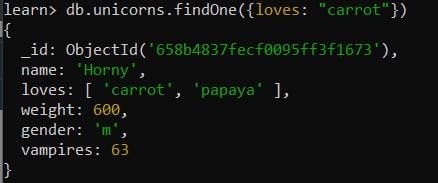
Самки:

db.unicorns.find({gender: "f"}).limit(3).sort({name: 1})



* 1. *Найдите всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций findOne и limit.*

db.unicorns.findOne({loves: "carrot"})



db.unicorns.find({loves: "carrot"}).limit(1)



## Практическое задание 2.2.2:

Модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпочтениях и поле.

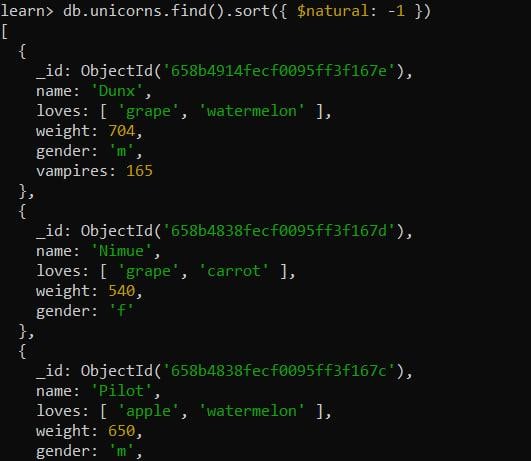
db.unicorns.find({gender : "m"}, {loves : 0, gender : 0}).sort({name: 1})



## Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

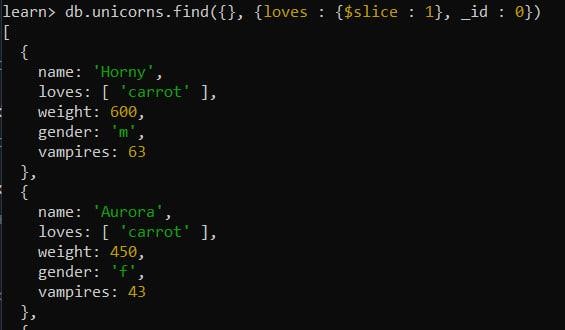
db.unicorns.find().sort({ $natural: -1 })



## Практическое задание 2.2.4:

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор.

db.unicorns.find({}, {loves : {$slice : 1}, \_id : 0})



## Практическое задание 2.3.1:

Вывести список самок единорогов весом от полутонны до 700 кг, исключив вывод идентификатора.

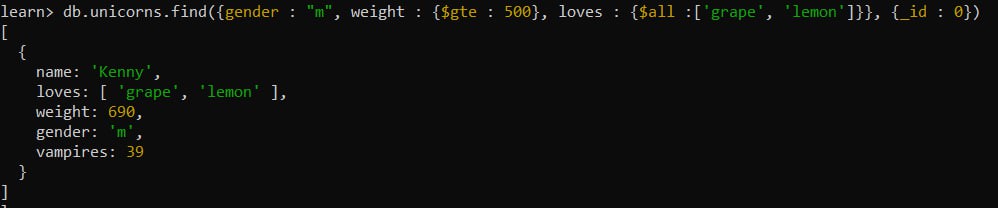
db.unicorns.find({gender : "f", weight : {$gte : 500, $lte : 700}}, {\_id: 0})



## Практическое задание 2.3.2:

Вывести список самцов единорогов весом от полутонны и предпочитающих grape и lemon, исключив вывод идентификатора.

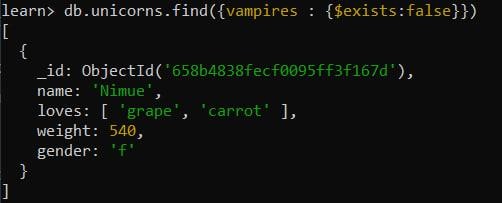
db.unicorns.find({gender : "m", weight : {$gte : 500}, loves : {$all :['grape', 'lemon']}}, {\_id : 0})



## Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ vampires.

db.unicorns.find({vampires : {$exists:false}})



## Практическое задание 2.3.4:

Вывести список упорядоченный список имен самцов единорогов с информацией об их первом предпочтении.

db.unicorns.find({gender : "m"}, {loves: {$slice : 1}})



## Практическое задание 3.1.1:

1. *Создайте коллекцию towns, включающую следующие документы:*

{name: "Punxsutawney ", populatiuon: 6200,

last\_sensus: ISODate("2008-01-31"), famous\_for: [""],

mayor: {

name: "Jim Wehrle"

}}

{name: "New York", populatiuon: 22200000,

last\_sensus: ISODate("2009-07-31"), famous\_for: ["status of liberty", "food"], mayor: {

name: "Michael Bloomberg", party: "I"}}

{name: "Portland", populatiuon: 528000,

last\_sensus: ISODate("2009-07-20"), famous\_for: ["beer", "food"], mayor: {

name: "Sam Adams", party: "D"}}

db.towns.insertMany([{name: "Punxsutawney ", populatiuon: 6200,

last\_sensus: ISODate("2008-01-31"), famous\_for: [""],

mayor: {

name: "Jim Wehrle"

}},

{name: "New York", populatiuon: 22200000,

last\_sensus: ISODate("2009-07-31"), famous\_for: ["status of liberty", "food"], mayor: {

name: "Michael Bloomberg", party: "I"}},

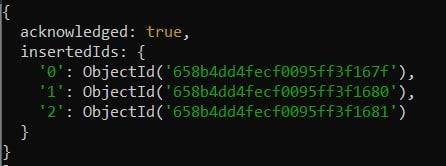
{name: "Portland", populatiuon: 528000,

last\_sensus: ISODate("2009-07-20"), famous\_for: ["beer", "food"],

mayor: {

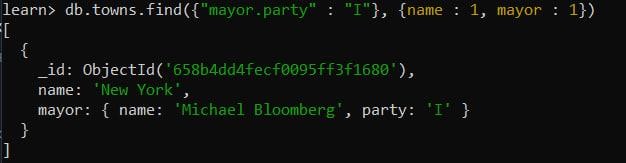
name: "Sam Adams", party: "D"}}

])



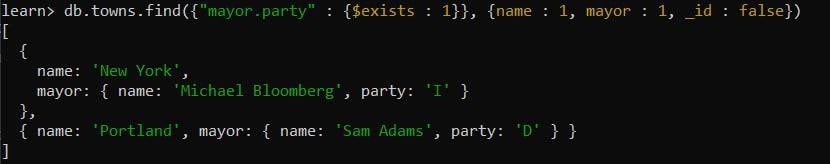
1. *Сформировать запрос, который возвращает список городов с независимыми мэрами (*party="I")*. Вывести только название города и информацию о мэре.*

db.towns.find({"mayor.party" : "I"}, {name : 1, mayor : 1})



1. *Сформировать запрос, который возвращает список беспартийных мэров (*party *отсутствует*)*. Вывести только название города и информацию о мэре.*

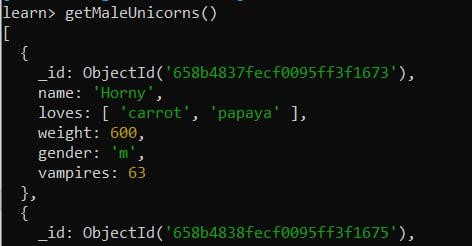
db.towns.find({"mayor.party" : {$exists : 1}}, {name : 1, mayor : 1, \_id : false})



## Практическое задание 3.1.2:

* 1. *Сформировать функцию для вывода списка самцов единорогов.*

function getMaleUnicorns() {return db.unicorns.find({gender:'m'})}



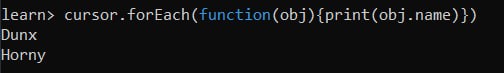
* 1. *Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.*

var cursor = getMaleUnicorns().sort({name:1}).limit(2);



* 1. *Вывести результат, используя forEach.*

cursor.forEach(function(obj){print(obj.name)})



## Практическое задание 3.2.1:

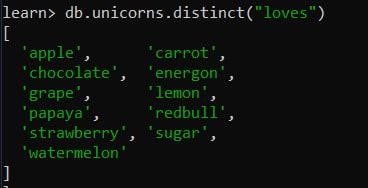
Вывести количество самок единорогов весом от полутонны до 600 кг.

db.unicorns.find({gender : "f", weight : {$gte : 500, $lte : 600}}).count()

## Практическое задание 3.2.2:

Вывести список предпочтений.

db.unicorns.distinct("loves")



## Практическое задание 3.2.3:

Посчитать количество особей единорогов обоих полов.

db.unicorns.aggregate({"$group":{\_id:"$gender", count : {$sum: 1}}})



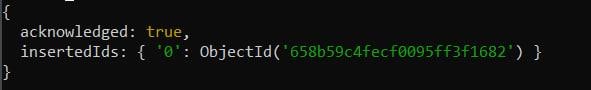
## Практическое задание 3.3.1:

1. *Выполнить команду:*

> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})

save не работает, так что:

db.unicorns.insert({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})



1. *Проверить содержимое коллекции unicorns.*



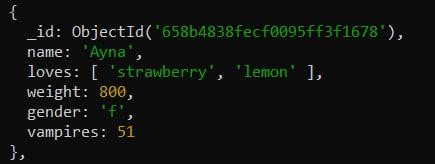
## Практическое задание 3.3.2:

* 1. *Для самки единорога* Ayna *внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.*

db.unicorns.updateOne({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})



* 1. *Проверить содержимое коллекции unicorns.*

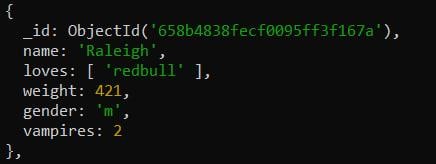


## Практическое задание 3.3.3:

1. *Для самца единорога* Raleigh *внести изменения в БД: теперь он любит рэдбул.*

db.unicorns.update({name : "Raleigh"}, {$set : {"loves" : ["redbull"]}})

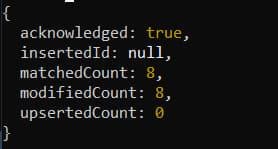
1. *Проверить содержимое коллекции unicorns.*



## Практическое задание 3.3.4:

* 1. *Всем самцам единорогов увеличить количество убитых вапмиров на 5.*

db.unicorns.updateMany({gender : "m"}, {$inc : {vampires : 5}})



* 1. *Проверить содержимое коллекции unicorns.*

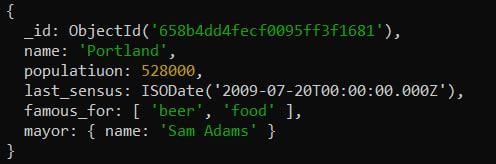


## Практическое задание 3.3.5:

1. *Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.*

db.towns.update({name : "Portland"}, {$unset: {"mayor.party" : 1}})

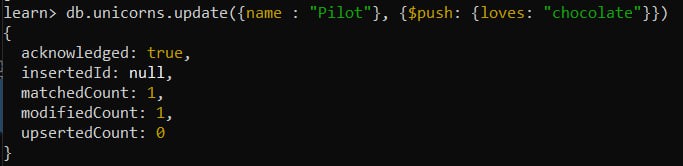
1. *Проверить содержимое коллекции towns.*



## Практическое задание 3.3.6:

1. *Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.*

db.unicorns.update({name : "Pilot"}, {$push: {loves: "chocolate"}})



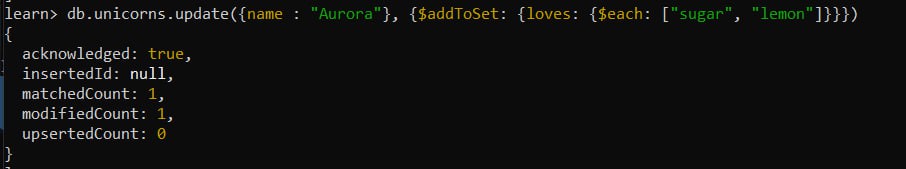
1. *Проверить содержимое коллекции unicorns.*



## Практическое задание 3.3.7:

1. *Изменить информацию о самке единорога* Aurora: *теперь она любит еще и сахар, и лимоны.*

db.unicorns.update({name : "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})



1. *Проверить содержимое коллекции unicorns.*



## Практическое задание 3.4.1:

* 1. *Создайте коллекцию towns, включающую следующие документы:*

{name: "Punxsutawney ", popujatiuon: 6200,

last\_sensus: ISODate("2008-01-31"), famous\_for: ["phil the groundhog"], mayor: {

name: "Jim Wehrle"

}}

{name: "New York", popujatiuon: 22200000,

last\_sensus: ISODate("2009-07-31"), famous\_for: ["status of liberty", "food"], mayor: {

name: "Michael Bloomberg", party: "I"}}

{name: "Portland", popujatiuon: 528000,

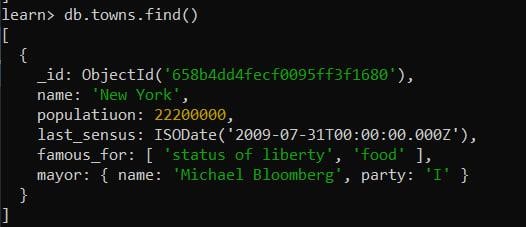
last\_sensus: ISODate("2009-07-20"), famous\_for: ["beer", "food"], mayor: {

name: "Sam Adams", party: "D"}}

* 1. *Удалите документы с беспартийными мэрами.*

db.towns.deleteMany({"mayor.party" : {$exists : false}})

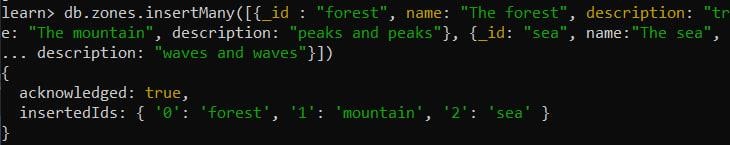
* 1. *Проверьте содержание коллекции.*



## Практическое задание 4.1.1:

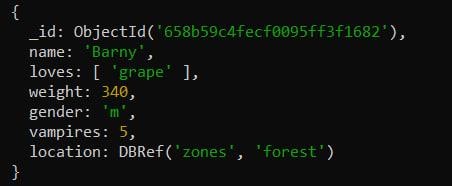
* + 1. *Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.*

db.zones.insertMany([{\_id : "forest", name: "The forest", description: "trees and trees"}, {\_id : "mountain", name: "The mountain", description: "peaks and peaks"}, {\_id: "sea", name:"The sea",description: "waves and waves"}])



* + 1. *Включите для нескольких единорогов в документы ссылку на зону обитания, использую второй способ автоматического связывания.*

db.unicorns.updateOne({\_id: ObjectId("658b59c4fecf0095ff3f1682")}, {$set: {location:{$ref : "zones", $id:"forest"}}})

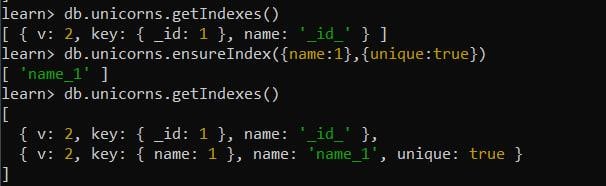


db.unicorns.updateOne({\_id: ObjectId("658b4838fecf0095ff3f167c")}, {$set: {location:{$ref : "zones", $id:"sea"}}})



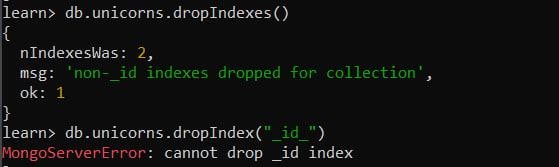
## Практическое задание 4.2.1:

1. *Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом* unique*.*



## Практическое задание 4.3.1:

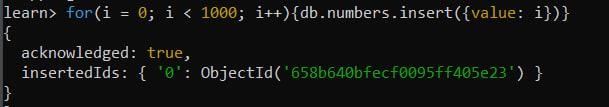
* 1. *Получите информацию о всех индексах коллекции unicorns .*
  2. *Удалите все индексы, кроме индекса для идентификатора.*
  3. *Попытайтесь удалить индекс для идентификатора.*



## Практическое задание 4.4.1:

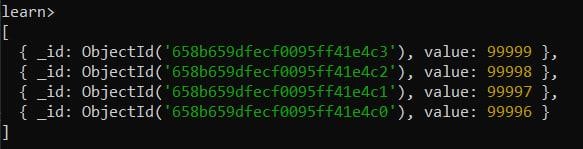
1. *Создайте объемную коллекцию numbers, задействовав курсор:*

for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}

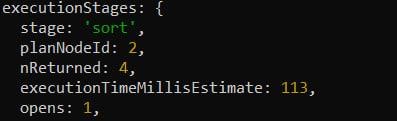


1. *Выберите последних четыре документа.*

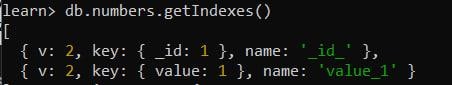
db.numbers.find().sort({value:-1}).limit(4)



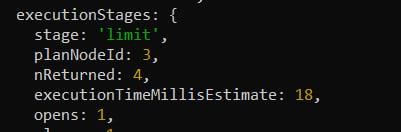
1. *Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis)*



1. *Создайте индекс для ключа value.*



1. *Получите информацию о всех индексах коллекции numbres.*
2. *Выполните запрос 2.*
3. *Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?*



1. *Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?*

Ускорение при запросе с индексами сильно заметно, следовательно запрос на выборку конкретных значений с индексами намного эффективнее такого же запроса, но без индексов.

# Вывод

В ходе лабораторной работы были использованы различные методы и освоены на практике NoSQL БД MongoDB.