

**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»**

Обучающийся Проскуряков Роман Владимирович
Факультет прикладной информатики
Группа К3239
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2024/2025

Цель работы: 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.

Практическое задание:

1. ВВЕДЕНИЕ В СУБД MONGODB. УСТАНОВКА MONGODB. НАЧАЛО РАБОТЫ С БД
2. CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ
3. АПРОСЫ К БАЗЕ ДАННЫХ MONGODB.ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ
4. ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

Выполнение:


1. ВВЕДЕНИЕ В СУБД MONGODB. УСТАНОВКА MONGODB. НАЧАЛО РАБОТЫ С БД

1.1. Установите MongoDB для обеих типов систем (32/64 бита).

```
C:\Program Files\MongoDB\Server\8.0\bin>. \mongod.exe
{"t":{"$date":"2025-06-23T05:47:33.615+03:00"},"s":"I", "c":
'none'}
{"t":{"$date":"2025-06-23T05:47:33.618+03:00"},"s":"I", "c":
{"t":{"$date":"2025-06-23T05:47:33.619+03:00"},"s":"I", "c":
'related parameters',"attr":{"relatedParameters":["tcpFastOpen
{"t":{"$date":"2025-06-23T05:47:33.622+03:00"},"s":"I", "c":
'on":0,"maxWireVersion":25},"incomingInternalClient":{"minWire
{"t":{"$date":"2025-06-23T05:47:33.624+03:00"},"s":"I", "c":
{"t":{"$date":"2025-06-23T05:47:33.624+03:00"},"s":"I", "c":
"64-bit","host":"DESKTOP-UC3TER5")}
```

1.2. Проверьте работоспособность системы запуском клиента mongo.

```

 mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Please enter a MongoDB connection string (Default: mongodb://localhost/):

Current Mongosh Log ID: 6858c8685fe5b092e9748a5e
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&se
Using MongoDB:          8.0.10
Using Mongosh:          2.5.3

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

-----
The server generated these startup warnings when booting
2025-06-23T05:53:20.296+03:00: Access control is not enabled for the dat
2025-06-23T05:53:20.297+03:00: This server is bound to localhost. Remote
it should serve responses from, or with --bind_ip_all to bind to all inter
-----

```

1.3. Выполните методы:

1.3.1. db.help()

```
learn> db.help()

Database Class:

  getMongo           Returns the current database connection
  getName            Returns the name of the DB
  getCollectionNames Returns an array containing the names of all coll
  getCollectionInfos Returns an array of documents with collection inf
  runCommand         Runs an arbitrary command on the database.
  adminCommand       Runs an arbitrary command against the admin datab
  aggregate          Runs a specified admin/diagnostic pipeline which
  getSiblingDB       Returns another database without modifying the db
  getCollection       Returns a collection or a view object that is fun
  dropDatabase       Removes the current database, deleting the associ
  createUser         Creates a new user for the database on which the
ase.
  updateUser         Updates the user's profile on the database on whi
cludes updates to the user's roles array.
  changeUserPassword Updates a user's password. Run the method in the
  logout             Ends the current authentication session. This fun
  dropUser           Removes the user from the current database.
  dropAllUsers       Removes all users from the current database.
  auth               Allows a user to authenticate to the database fro
  grantRolesToUser   Grants additional roles to a user.
```

1.3.2. db.help

```
learn> db.help

Database Class:

getMongo           Returns the current database
getName            Returns the name of the DB
getCollectionNames Returns an array containing t
getCollectionInfos  Returns an array of documents
runCommand          Runs an arbitrary command on
adminCommand        Runs an arbitrary command aga
aggregate           Runs a specified admin/diagno
getSiblingDB        Returns another database with
getCollection        Returns a collection or a vie
dropDatabase        Removes the current database,
createUser           Creates a new user for the da
ase.
updateUser          Updates the user's profile on
cludes updates to the user's roles array.
changeUserPassword   Updates a user's password. Ru
```

1.3.3. db.stats()

```
test> db.stats()
{
  db: 'test',
  collections: Long('0'),
  views: Long('0'),
  objects: Long('0'),
  avgObjSize: 0,
  dataSize: 0,
  storageSize: 0,
  indexes: Long('0'),
  indexSize: 0,
  totalSize: 0,
  scaleFactor: Long('1'),
  fsUsedSize: 0,
  fsTotalSize: 0,
  ok: 1
}
```

1.4. Создайте БД learn.

```
use learn
```

```
test> use learn
switched to db learn
```

1.5. Получите список доступных БД.

```
show dbs
```

```
learn> show dbs
admin   40.00 KiB
config  60.00 KiB
local   40.00 KiB
```

1.6. Создайте коллекцию unicorns, вставив в нее документ {name: 'Aurora', gender: 'f', weight: 450}.

```
db.unicorns.insertOne({ name: 'Aurora', gender: 'f', weight: 450 })
```

```
learn> db.unicorns.insertOne({ name: 'Aurora', gender: 'f', weight: 450 })
{
  acknowledged: true,
  insertedId: ObjectId('6858c36e7b54e1021b748a5f')
}
```

1.7. Просмотрите список текущих коллекций.

```
show collections
```

```
learn> show collections
unicorns
```

1.8. Переименуйте коллекцию unicorns.

```
db.unicorns.renameCollection('myUnicorns')
```

```
learn> db.unicorns.renameCollection('myUnicorns')
{ ok: 1 }
```

1.9. Просмотрите статистику коллекции.

```
db.myUnicorns.stats()
```

```
learn> db.myUnicorns.stats()
{
  ok: 1,
  capped: false,
  wiredTiger: {
    metadata: { formatVersion: 1 },
    creationString: 'access_pattern_hint=none,allocation_size=4KB,ap
allocation=best,block_compressor=snappy,cache_resident=false,checksum
ffman_value=,ignore_in_memory_cache_size=false,immutable=false,import
max=0,internal_key_max=0,internal_key_truncate=true,internal_page_ma
throttle=true,bloom=true,bloom_bit_count=16,bloom_config=,bloom_hash
erge_max=15,merge_min=0),memory_page_image_max=0,memory_page_max=10m
pen_per_child=0,split_pct=90,tiered_storage=(auth_token=,bucket=,bud
usage=none',
  type: 'file',
  uri: 'statistics:table:collection-7-11426270120820682950',
  LSM: {
    'bloom filter false positives': 0,
    'bloom filter hits': 0,
    'bloom filter misses': 0,
    'bloom filter pages evicted from cache': 0,
    'bloom filter pages read into cache': 0,
    'bloom filters in the LSM tree': 0,
    'chunks in the LSM tree': 0,
    'highest merge generation in the LSM tree': 0,
    'queries that could have benefited from a Bloom filter that di
    'sleep for LSM checkpoint throttle': 0,
```

1.10. Удалите коллекцию.

```
db.myUnicorns.drop()
```

```
learn> db.myUnicorns.drop()
true
```

1.11. Удалите БД learn.

```
db.dropDatabase()
```

```
learn> db.dropDatabase()
{ ok: 1, dropped: 'learn' }
```

2. 2 CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ

2.1. ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

2.1.1. Создайте базу данных learn и заполните коллекцию unicorns указанными документами.

```
use learn
```

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

... (остальные 10 insert)

```
var doc = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
db.unicorns.insert(doc)
```

```
learn> db.unicorns.find().pretty()
[
  {
    _id: ObjectId('6858cc695fe5b092e9748a5f'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a60'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a61'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a62'),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a63'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a64'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a65'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a66'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a67'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a68'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a69'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6858cc6b15fe5b092e9748a6a'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

2.2. ВЫБОРКА ДАННЫХ ИЗ БД

2.2.1. Сформируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени. Найдите всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций findOne и limit.

самцы

```
db.unicorns.find({gender:'m'}).sort({name:1})
```

самки, первые три

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

```
learn> db.unicorns.find({gender:'m'}).sort({name:1})
[
  {
    _id: ObjectId('6858ccb15fe5b092e9748a6a'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a5f'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a65'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a68'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a66'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a62'),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  }
]
```

```
learn> db.unicorns.find({gender:'f'}).sort({name:1}).limit(3)
[
  {
    _id: ObjectId('6858cc695fe5b092e9748a60'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6858cc695fe5b092e9748a64'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a67'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

```
db.unicorns.find({gender:'m'}, { id:0, loves:0})
```

```
learn> db.unicorns.find({gender:'m'}, {_id:0, loves:0})
[
  { name: 'Horny', weight: 600, gender: 'm', vampires: 63 },
  { name: 'Unicrom', weight: 984, gender: 'm', vampires: 182 },
  { name: 'Rooooooodles', weight: 575, gender: 'm', vampires: 99 },
  { name: 'Kenny', weight: 690, gender: 'm', vampires: 39 },
  { name: 'Raleigh', weight: 421, gender: 'm', vampires: 2 },
  { name: 'Pilot', weight: 650, gender: 'm', vampires: 54 },
  { name: 'Dunx', weight: 704, gender: 'm', vampires: 165 }
]
```

2.2.3. Вывести список единорогов в обратном порядке добавления.

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



```
learn> db.unicorns.find().sort({$natural:-1})
[
  {
    _id: ObjectId('6858ccb15fe5b092e9748a6a'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a69'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a68'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a67'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a66'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
]
```

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

`db.unicorns.find({}, { id:0, name:1, firstLove:{ $slice:['$loves',1] }})`

```
learn> db.unicorns.find({}, {_id:0, name:1, firstLove:{ $slice:['$loves',1] }})
[
  { name: 'Horny', firstLove: [ 'carrot' ] },
  { name: 'Aurora', firstLove: [ 'carrot' ] },
  { name: 'Unicrom', firstLove: [ 'energon' ] },
  { name: 'Rooooooodles', firstLove: [ 'apple' ] },
  { name: 'Solnara', firstLove: [ 'apple' ] },
  { name: 'Ayna', firstLove: [ 'strawberry' ] },
  { name: 'Kenny', firstLove: [ 'grape' ] },
  { name: 'Raleigh', firstLove: [ 'apple' ] },
  { name: 'Leia', firstLove: [ 'apple' ] },
  { name: 'Pilot', firstLove: [ 'apple' ] },
  { name: 'Nimue', firstLove: [ 'grape' ] },
  { name: 'Dunx', firstLove: [ 'grape' ] }
]
```

2.3. ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

2.3.1. Вывести список самок единорогов весом от полутонны до 700 кг (без _id)

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

```
learn> db.unicorns.find({gender:'f', weight:{$gte:500,$lte:700}}, {_id:0})
[
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

2.3.2. Вывести список самцов единорогов весом ≥ 500 кг, предпочитающих grape и lemon (без _id).

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

```
learn> db.unicorns.find({gender:'m', weight:{$gte:500}, loves:{$all:['grape','lemon']}}, {_id:0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

2.3.3. Найти всех единорогов, не имеющих ключ vampires.

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

```
learn> db.unicorns.find({vampires:{$exists:false}})
[
  {
    _id: ObjectId('6858cc6a5fe5b092e9748a69'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

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

db.unicorns.find({gender:'m'}, {_id:0, name:1, firstLove:{\$slice:['\$loves',1]}}).sort({name:1})

```
learn> db.unicorns.find({gender:'m'}, {_id:0, name:1, firstLove:{$slice:['$loves',1]}}).sort({name:1})
[
  { name: 'Dunx', firstLove: [ 'grape' ] },
  { name: 'Horny', firstLove: [ 'carrot' ] },
  { name: 'Kenny', firstLove: [ 'grape' ] },
  { name: 'Pilot', firstLove: [ 'apple' ] },
  { name: 'Raleigh', firstLove: [ 'apple' ] },
  { name: 'Rooooooodles', firstLove: [ 'apple' ] },
  { name: 'Unicrom', firstLove: [ 'energon' ] }
]
```

3. ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB. ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ

3.1. ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

3.1.1. Создайте коллекцию towns, включающую следующие документы:

3.1.1.1. 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"}, {_id:0, name:1, mayor:1 })
```

```
learn> db.towns.find({"mayor.party":"I"}, {_id:0, name:1, mayor:1 })
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

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

```
db.towns.find({"mayor.party":{"$exists:false"}}, {_id:0, name:1, mayor:1 })
```

```
learn> db.towns.find({"mayor.party":{"$exists:false"}}, {_id:0, name:1, mayor:1 })
[ { name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } } ]
```

3.1.2. Сформировать функцию для вывода списка самцов единорогов. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке. Вывести результат, используя forEach.

```
var male = function () { return this.gender === 'm'; }
```

```
var curs = db.unicorns.find({ $where: male }).sort({ name: 1 }).limit(2);
```

```
curs.forEach(printjson);
```

```
learn> var male = function () { return this.gender === 'm'; }
learn> var curs = db.unicorns.find({ $where: male }).sort({ name: 1 }).limit(2);
learn> curs.forEach(printjson);
{
  _id: ObjectId('6858da02faba195faf748a78'),
  name: 'Dunx',
  loves: [
    'grape',
    'watermelon'
  ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId('6858da01faba195faf748a6d'),
  name: 'Horny',
  loves: [
    'carrot',
    'papaya'
  ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

3.2. АГРЕГИРОВАННЫЕ ЗАПРОСЫ

3.2.1. Вывести количество самок единорогов весом от полутонны до 600 кг.

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

```
learn> db.unicorns.find({gender:'f', weight:{$gte:500,$lte:600}}).count()  
2
```

3.2.2. Вывести список предпочтений (loves).

```
db.unicorns.distinct('loves')
```

```
learn> db.unicorns.distinct('loves')  
[  
  'apple',      'carrot',  
  'chocolate', 'energon',  
  'grape',      'lemon',  
  'papaya',     'redbull',  
  'strawberry', 'sugar',  
  'watermelon'  
]
```

3.2.3. Посчитать количество единорогов обоих полов.

```
db.unicorns.aggregate([{$group:{_id:'$gender', count:{$sum:1}}}])
```

```
learn> db.unicorns.aggregate([{$group:{_id:'$gender', count:{$sum:1}}}])  
[ { _id: 'm', count: 7 }, { _id: 'f', count: 5 } ]
```

3.3. РЕДАКТИРОВАНИЕ ДАННЫХ

3.3.1. Сохранить документ Барну в коллекции unicorns и проверить содержимое.

```
db.unicorns.insertOne({name:'Barney', loves:['grape'], weight:340, gender:'m'})
```

```
learn> db.unicorns.insertOne({name:'Barney', loves:['grape'], weight:340, gender:'m'})  
{  
  acknowledged: true,  
  insertedId: ObjectId('6858df2dfaba195faf748a7b')  
}
```

```
db.unicorns.find({name:'Barney'})
```

```
learn> db.unicorns.find({name:'Barney'})  
[  
  {  
    _id: ObjectId('6858df2dfaba195faf748a7b'),  
    name: 'Barney',  
    loves: [ 'grape' ],  
    weight: 340,  
    gender: 'm'  
  }  
]
```

3.3.2. Изменить Айна: вес = 800, vampires = 51 и проверить коллекцию.

```
db.unicorns.update({name:'Ayna'}, {$set:{weight:800, vampires:51}})
```

```
db.unicorns.find({name:'Ayna'})
```

```
learn> db.unicorns.update({name:'Ayna'}, {$set:{weight:800, vampires:51}})  
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.  
{  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 1,  
  modifiedCount: 1,  
  upsertedCount: 0  
}  
learn> db.unicorns.find({name:'Ayna'})  
[  
  {  
    _id: ObjectId('6858da01faba195faf748a72'),  
    name: 'Ayna',  
    loves: [ 'strawberry', 'lemon' ],  
    weight: 800,  
    gender: 'f',  
    vampires: 51  
  }  
]
```

3.3.3. Изменить Raleigh: добавить предпочтение redbull и проверить коллекцию.

```
db.unicorns.update({name:'Raleigh'}, { $addToSet:{ loves:'redbull' } })
```

```
db.unicorns.find({name:'Raleigh'})
```

```
learn> db.unicorns.update({name: 'Raleigh'}, { $addToSet:{ loves:'redbull' } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Raleigh'})
[
  {
    _id: ObjectId('6858da01faba195faf748a74'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

3.3.4. Всем самцам увеличить vampires на 5 и проверить коллекцию.

```
db.unicorns.update({gender:'m'}, { $inc:{ vampires:5 } }, { multi:true })
```

```
db.unicorns.find({gender:'m'}, {name:1, vampires:1})
```

```
learn> db.unicorns.update({gender: 'm'}, { $inc:{ vampires:5 } }, { multi:true })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: 'm'}, {name:1, vampires:1})
[
  {
    _id: ObjectId('6858da01faba195faf748a6d'),
    name: 'Horny',
    vampires: 68
  },
  {
    _id: ObjectId('6858da01faba195faf748a6f'),
    name: 'Unicrom',
    vampires: 187
  },
  {
    _id: ObjectId('6858da01faba195faf748a70'),
    name: 'Rooooooodles',
    vampires: 104
  },
  {
    _id: ObjectId('6858da01faba195faf748a73'),
    name: 'Kenny',
    vampires: 44
  },
  {
    _id: ObjectId('6858da01faba195faf748a74'),
    name: 'Raleigh',
    vampires: 7
  },
  {
    _id: ObjectId('6858da01faba195faf748a76'),
    name: 'Pilot',
    vampires: 59
  },
  {
    _id: ObjectId('6858da02faba195faf748a78'),
    name: 'Dunx',
    vampires: 170
  },
  {
    _id: ObjectId('6858df2dfaba195faf748a7b'),
    name: 'Barny',
    vampires: 5
  }
]
```

3.3.5. Обновить город Portland: мэр беспартийный и проверить коллекцию towns.

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

```
db.towns.find({name:'Portland'})
```

```
learn> db.towns.find({name:'Portland'})
[
  {
    _id: ObjectId('6858d7b5faba195faf748a61'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

3.3.6. Изменить Pilot: теперь любит chocolate и проверить коллекцию.

```
db.unicorns.update({name:'Pilot'}, { $addToSet:{ loves:'chocolate' } })
```

```
db.unicorns.find({name:'Pilot'})
```

```
learn> db.unicorns.find({name:'Pilot'})
[
  {
    _id: ObjectId('6858da01faba195faf748a76'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

3.3.7. Изменить Aurora: теперь любит sugar и lemon и проверить коллекцию.

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

```
db.unicorns.find({name:'Aurora'})
```

```
learn> db.unicorns.find({name:'Aurora'})
[
  {
    _id: ObjectId('6858da01faba195faf748a6e'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

3.4. УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

3.4.1. Создайте коллекцию towns, включающую следующие документы:

3.4.1.1. Удалите документы с беспартийными мэрами. Проверьте содержание коллекции.

```
db.towns.remove({'mayor.party':{$exists:false}})
```

```
db.towns.find()
```

```
learn> db.towns.remove({'mayor.party':{$exists:false}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()
[
  {
    _id: ObjectId('6858e289faba195faf748a7d'),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('6858e289faba195faf748a7e'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

3.4.1.2. Очистите коллекцию. Просмотрите список доступных коллекций.

```
db.towns.remove({})
```

```
show collections
```

```
learn> db.towns.remove({})
{ acknowledged: true, deletedCount: 2 }
learn> show collections
towns
unicorns
learn> db.towns.find()
```

4. ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

4.1. ССЫЛКИ В БД

4.1.1. Создать коллекцию зон обитания, добавить ссылки у нескольких единорогов, проверить unicorns

```
db.habitats.insert({_id:'forest', full:'Enchanted Forest', descr:'Dense magical woods'})
db.habitats.insert({_id:'moon', full:'Moon (satellite)', descr:'A cold, airless wasteland'})
```

```
db.unicorns.update({name:'Horny'}, { $set: { habitat: { $ref:'habitats', $id:'forest' } } })
db.unicorns.update({name:'Dunx'}, { $set: { habitat: { $ref:'habitats', $id:'moon' } } })
db.unicorns.find({habitat:{$exists:true}})
```

```
learn> db.unicorns.find({habitat:{$exists:true}})
[
  {
    _id: ObjectId('6858e72ffaba195faf748aa1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63,
    habitat: DBRef('habitats', 'forest')
  },
  {
    _id: ObjectId('6858e730faba195faf748aab'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165,
    habitat: DBRef('habitats', 'moon')
  }
]
```

4.2. НАСТРОЙКА ИНДЕКСОВ

4.2.1. Проверить возможность установки unique-индекса на name в коллекции unicorns.

```
db.unicorns.createIndex({name:1}, {unique:true})
```

```
learn> db.unicorns.createIndex({name:1}, {unique:true})
name_1
```

4.3. УПРАВЛЕНИЕ ИНДЕКСАМИ

4.3.1. Получить все индексы коллекции unicorns, удалить все кроме _id_, попытаться удалить _id_

```
db.unicorns.getIndexes()
db.unicorns.dropIndexes()
db.unicorns.dropIndex('_id')
```

```
learn> db.unicorns.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
learn> db.unicorns.dropIndexes()
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
learn> db.unicorns.dropIndex('_id_')
MongoServerError[InvalidOptions]: cannot drop _id index
```


4.4. ПЛАН ЗАПРОСА

4.4.1. Создайте объемную коллекцию numbers, задействовав курсор:

4.4.1.1. Выберите последних четыре документа.

`db.numbers.find().sort({$natural:-1}).limit(4)`

```
learn> db.numbers.find().sort({$natural:-1}).limit(4)
[
  { _id: ObjectId('6858ecd1faba195faf76114b'), value: 99999 },
  { _id: ObjectId('6858ecd1faba195faf76114a'), value: 99998 },
  { _id: ObjectId('6858ecd1faba195faf761149'), value: 99997 },
  { _id: ObjectId('6858ecd1faba195faf761148'), value: 99996 }
]
```

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

`db.numbers.explain('executionStats').find().sort({$natural:-1}).limit(4)`

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 0,
  totalKeysExamined: 0,
  totalDocsExamined: 4,
}
```

4.4.1.3. Создайте индекс для ключа value.

`db.numbers.createIndex({value:1})`

```
learn> db.numbers.createIndex({value:1})
value_1
```

4.4.1.4. Получите информацию о всех индексах коллекции numbers.

`db.numbers.getIndexes()`

```
learn> db.numbers.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { value: 1 }, name: 'value_1' }
]
```

4.4.1.5. Выполните запрос 2.

`db.numbers.find().sort({$natural:-1}).limit(4)`

```
learn> db.numbers.find().sort({$natural:-1}).limit(4)
[
  { _id: ObjectId('6858ecd1faba195faf76114b'), value: 99999 },
  { _id: ObjectId('6858ecd1faba195faf76114a'), value: 99998 },
  { _id: ObjectId('6858ecd1faba195faf761149'), value: 99997 },
  { _id: ObjectId('6858ecd1faba195faf761148'), value: 99996 }
]
```

4.4.1.6. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

`db.numbers.explain('executionStats').find().sort({$natural:-1}).limit(4)`

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 0,
  totalKeysExamined: 0,
  totalDocsExamined: 4,
}
```

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

Время запросов одинаково и равно 0. Скорее всего данных слишком мало. Но у друга изменения были, так что я всё осознал.

Выводы

MongoDB очень похоже по функционалу и логике на sql. Многие функции называются так же. Он похож как два языка программирования с разными стилями. В ходе выполнения лабораторной работы были получены базовые навыки управления базами данных MongoDB, что будет полезно для дальнейшей работы с базами данных и разработки эффективных баз данных.