Отчет по лабораторной работе 6.1, 6.2

Выполнил: Полярус Павел Анатольевич

Тема: Работа с БД в СУБД MongoDB

Цель работы: Овладеть практическими навыками работы с CRUD-операциями, вложенными объектами, агрегацией, изменением данных, ссылками и индексами в MongoDB.

Выполнение:

Лабораторная работа 6.1

Задание 3

Выполните методы:

- db.help()
- db.help
- db.stats()

a)

```
db.help()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] test> 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 collections in the current database.
    getCollectionInfos                      Returns an array of documents with collection information, i.e. collection name and options, for the current database.
    runCommand                              Runs an arbitrary command on the database.
    adminCommand                            Runs an arbitrary command against the admin database.
    aggregate                               Runs a specified admin/diagnostic pipeline which does not require an underlying collection.
    getSiblingDB                            Returns another database without modifying the db variable in the shell environment.
    getCollection                           Returns a collection or a view object that is functionally equivalent to using the db.<collectionName>.
    dropDatabase                            Removes the current database, deleting the associated data files.
    createUser                              Creates a new user for the database on which the method is run. db.createUser() returns a duplicate user error if the user already exists on the
 database.
    updateUser                              Updates the user's profile on the database on which you run the method. An update to a field completely replaces the previous field's values. Th
is includes updates to the user's roles array.
    changeUserPassword                      Updates a user's password. Run the method in the database where the user is defined, i.e. the database you created the user.
    logout                                  Ends the current authentication session. This function has no effect if the current session is not authenticated.
    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 from within the shell.
    grantRolesToUser                        Grants additional roles to a user.
    revokeRolesFromUser                     Removes a one or more roles from a user on the current database.
    getUser                                 Returns user information for a specified user. Run this method on the user's database. The user must exist on the database on which the method r
uns.
    getUsers                                Returns information for all the users in the database.
    createCollection                        Create new collection
    createEncryptedCollection               Creates a new collection with a list of encrypted fields each with unique and auto-created data encryption keys (DEKs). This is a utility functi
on that internally utilises ClientEnryption.createEncryptedCollection.
    createView                              Create new view
    createRole                              Creates a new role.
    updateRole                              Updates the role's profile on the database on which you run the method. An update to a field completely replaces the previous field's values.
    dropRole                                Removes the role from the current database.
    dropAllRoles                            Removes all roles from the current database.
    grantRolesToRole                        Grants additional roles to a role.
    revokeRolesFromRole                     Removes a one or more roles from a role on the current database.
    grantPrivilegesToRole                   Grants additional privileges to a role.
    revokePrivilegesFromRole                Removes a one or more privileges from a role on the current database.
    getRole                                 Returns role information for a specified role. Run this method on the role's database. The role must exist on the database on which the method r
uns.
    getRoles                                Returns information for all the roles in the database.
    currentOp                               Runs an aggregation using $currentOp operator. Returns a document that contains information on in-progress operations for the database instance.
 For further information, see $currentOp.
    killOp                                  Calls the killOp command. Terminates an operation as specified by the operation ID. To find operations and their corresponding IDs, see $current
Op or db.currentOp().
    shutdownServer                          Calls the shutdown command. Shuts down the current mongod or mongos process cleanly and safely. You must issue the db.shutdownServer() operation
```

b)

```
db.help
```

```
Atlas atlas-f8hxqs-shard-0 [primary] test> 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 collections in the current database.
    getCollectionInfos                      Returns an array of documents with collection information, i.e. collection name and options, for the current database.
    runCommand                              Runs an arbitrary command on the database.
    adminCommand                            Runs an arbitrary command against the admin database.
    aggregate                               Runs a specified admin/diagnostic pipeline which does not require an underlying collection.
    getSiblingDB                            Returns another database without modifying the db variable in the shell environment.
    getCollection                           Returns a collection or a view object that is functionally equivalent to using the db.<collectionName>.
    dropDatabase                            Removes the current database, deleting the associated data files.
    createUser                              Creates a new user for the database on which the method is run. db.createUser() returns a duplicate user error if the user already exists on the
 database.
    updateUser                              Updates the user's profile on the database on which you run the method. An update to a field completely replaces the previous field's values. Th
is includes updates to the user's roles array.
    changeUserPassword                      Updates a user's password. Run the method in the database where the user is defined, i.e. the database you created the user.
    logout                                  Ends the current authentication session. This function has no effect if the current session is not authenticated.
    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 from within the shell.
    grantRolesToUser                        Grants additional roles to a user.
    revokeRolesFromUser                     Removes a one or more roles from a user on the current database.
    getUser                                 Returns user information for a specified user. Run this method on the user's database. The user must exist on the database on which the method r
uns.
    getUsers                                Returns information for all the users in the database.
    createCollection                        Create new collection
    createEncryptedCollection               Creates a new collection with a list of encrypted fields each with unique and auto-created data encryption keys (DEKs). This is a utility functi
on that internally utilises ClientEnryption.createEncryptedCollection.
    createView                              Create new view
    createRole                              Creates a new role.
    updateRole                              Updates the role's profile on the database on which you run the method. An update to a field completely replaces the previous field's values.
    dropRole                                Removes the role from the current database.
    dropAllRoles                            Removes all roles from the current database.
    grantRolesToRole                        Grants additional roles to a role.
    revokeRolesFromRole                     Removes a one or more roles from a role on the current database.
    grantPrivilegesToRole                   Grants additional privileges to a role.
    revokePrivilegesFromRole                Removes a one or more privileges from a role on the current database.
    getRole                                 Returns role information for a specified role. Run this method on the role's database. The role must exist on the database on which the method r
uns.
    getRoles                                Returns information for all the roles in the database.
    currentOp                               Runs an aggregation using $currentOp operator. Returns a document that contains information on in-progress operations for the database instance.
 For further information, see $currentOp.
    killOp                                  Calls the killOp command. Terminates an operation as specified by the operation ID. To find operations and their corresponding IDs, see $current
Op or db.currentOp().
    shutdownServer                          Calls the shutdown command. Shuts down the current mongod or mongos process cleanly and safely. You must issue the db.shutdownServer() operation
```

c)

```
db.stats()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] test> db.stats()
{
  db: 'test',
  collections: 0,
  views: 0,
  objects: Long('0'),
  avgObjSize: 0,
  dataSize: Long('0'),
  storageSize: Long('0'),
  totalFreeStorageSize: Long('0'),
  numExtents: Long('0'),
  indexes: 0,
  indexSize: Long('0'),
  indexFreeStorageSize: Long('0'),
  fileSize: Long('0'),
  nsSizeMB: 0,
  ok: 1
}
Atlas atlas-f8hxqs-shard-0 [primary] test>
```

1. Создайте БД learn.

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

3. Получите список доступных БД.

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.insertOne({name: 'Aurora', gender: 'f', weight: 450})
{
  acknowledged: true,
  insertedId: ObjectId('6837583f8b3a1dd66beb5415')
}
Atlas atlas-f8hxqs-shard-0 [primary] learn> show dbs
learn                8.00 KiB
sample_airbnb       77.18 MiB
sample_analytics     9.56 MiB
sample_geospatial    1.20 MiB
sample_guides       40.00 KiB
sample_mflix       120.29 MiB
sample_restaurants   7.93 MiB
sample_supplies      1.07 MiB
sample_training     51.78 MiB
sample_weatherdata   2.63 MiB
admin              348.00 KiB
local               21.51 GiB
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

Задание 7

Просмотрите список текущих коллекций.

```
db.getCollectionNames()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.getCollectionNames()
[ 'unicorns' ]
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

Задание 8

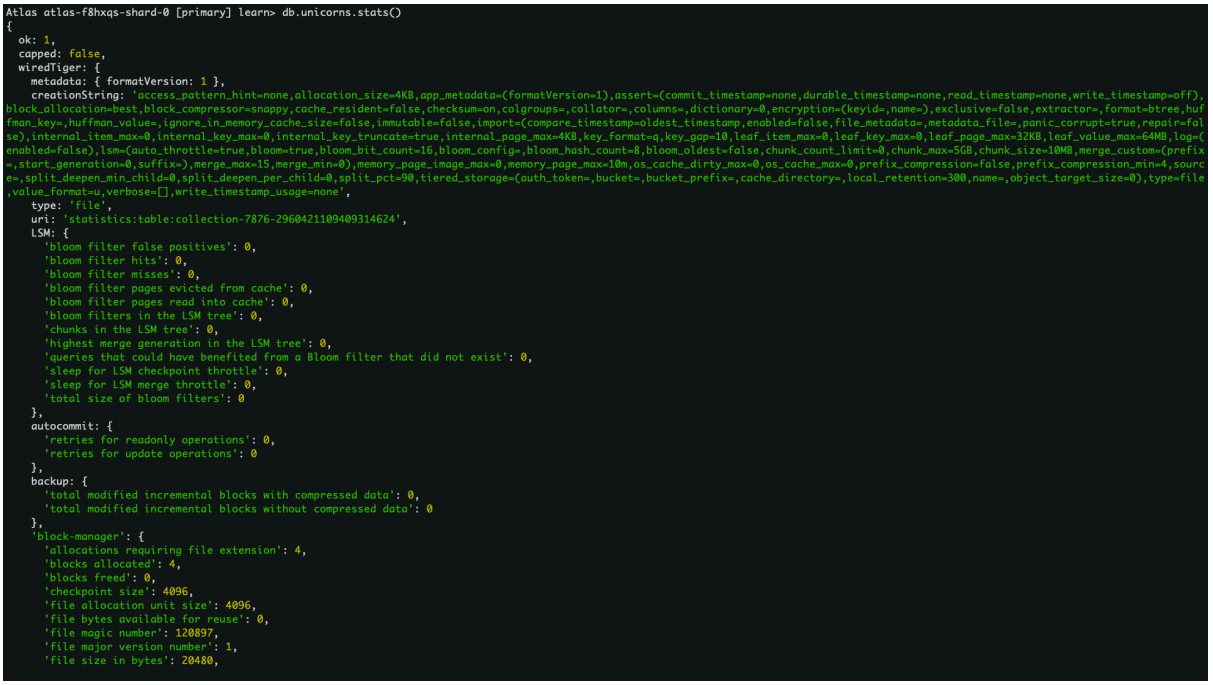
Переименуйте коллекцию unicorns.

```
db.unicorns.renameCollection("new_unicorns")
```

Задание 9

Просмотрите статистику коллекции.

```
db.new_unicorns.stats()
```



Задание 10

Удалите коллекцию

```
db.new_unicorns_name.drop()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.new_unicorns_name.drop()
true
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

Задание 11

Удалите БД learn.

```
db.dropDataBase()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.dropDatabase()
{ ok: 1, dropped: 'learn' }
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

Задание 2.1.1

1) Создайте базу данных learn.
2) Заполните коллекцию единорогов unicorns:
3) *Используя второй способ, вставьте в коллекцию единорогов документ:*
`{name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}`

4) Проверьте содержимое коллекции с помощью метода find.

1)

```
use learn
```

2)

```
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'});
```

3)

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> document = ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
... db.unicorns.insert(document);
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('68377a638b3a1dd66beb5421') }
}
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

4)

```
db.unicorns.find()
```

Результат:

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find()
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5416'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5417'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5418'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5419'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb541a'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb541b'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
```

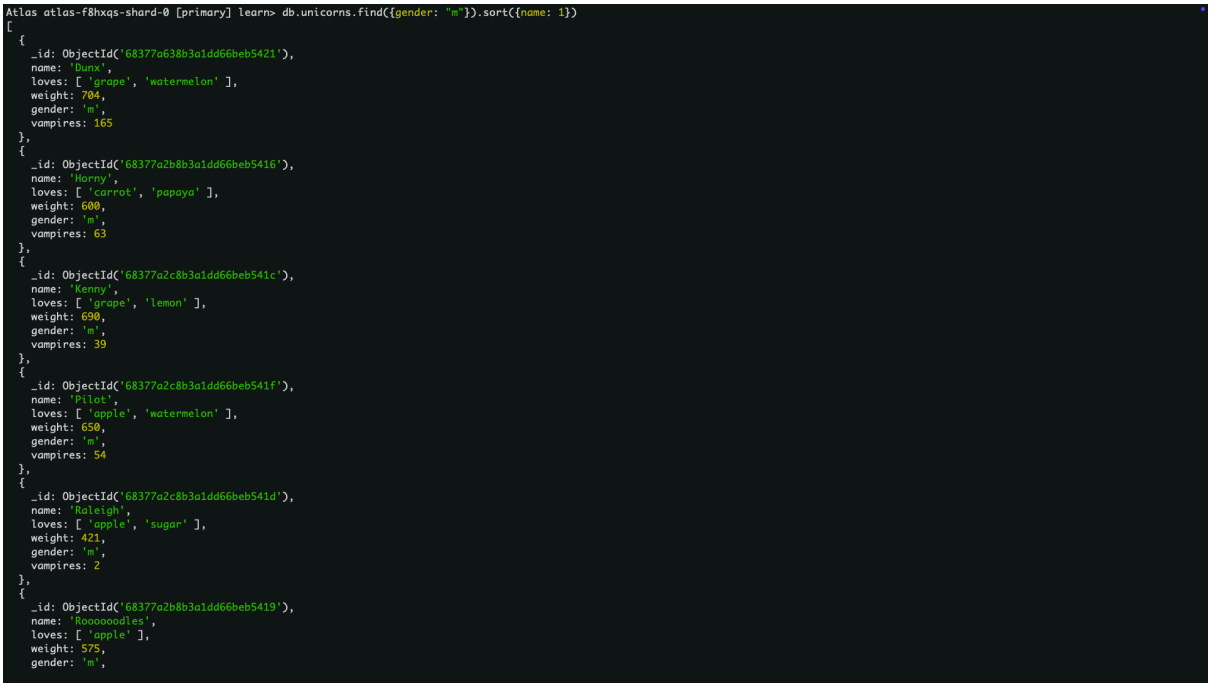
Задание 2.2.1

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

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

Самцы, отсортированные по имени:

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



```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({gender: "m"}).sort({name: 1})
[
  {
    _id: ObjectId('68377a638b3a1dd66beb5421'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5416'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541c'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541f'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541d'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5419'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
```

Первые три самки по имени:

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5417'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb541b'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541e'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

Все самки, любящие carrot:

```
db.unicorns.find({gender: "f", loves: "carrot"})
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({gender: "f", loves: "carrot"})
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5417'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb541a'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb5420'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Вывод первой самки из предыдущего запроса:

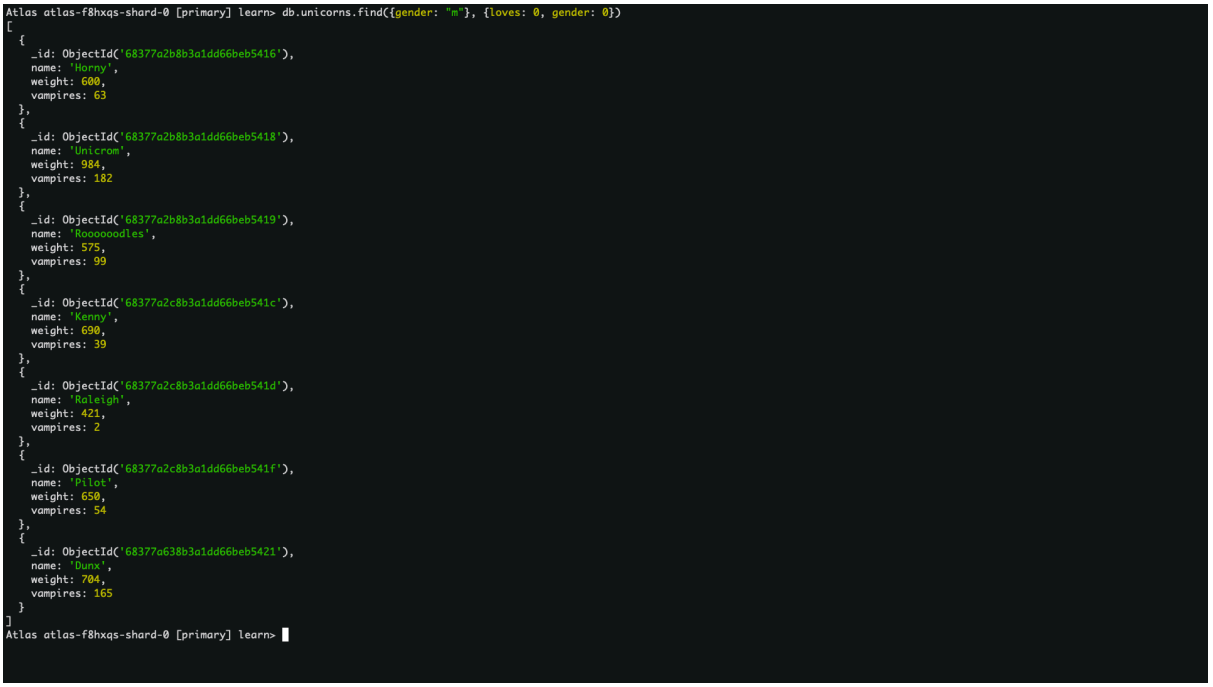
```
db.unicorns.findOne({gender: "f", loves: "carrot"})
db.unicorns.find({gender: "f", loves: "carrot"}).limit(1)
```

```
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5417'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Задание 2.2.2 — Исключение полей

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

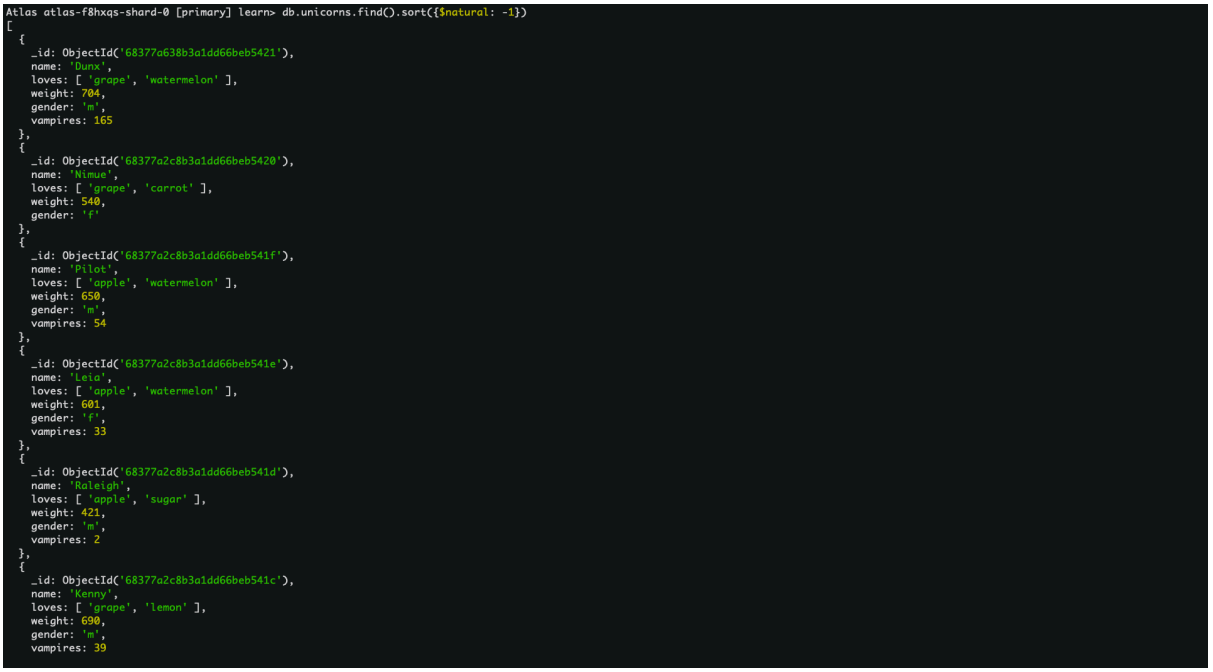
```
db.unicorns.find({gender: "m"}, {loves: 0, gender: 0})
```



Задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

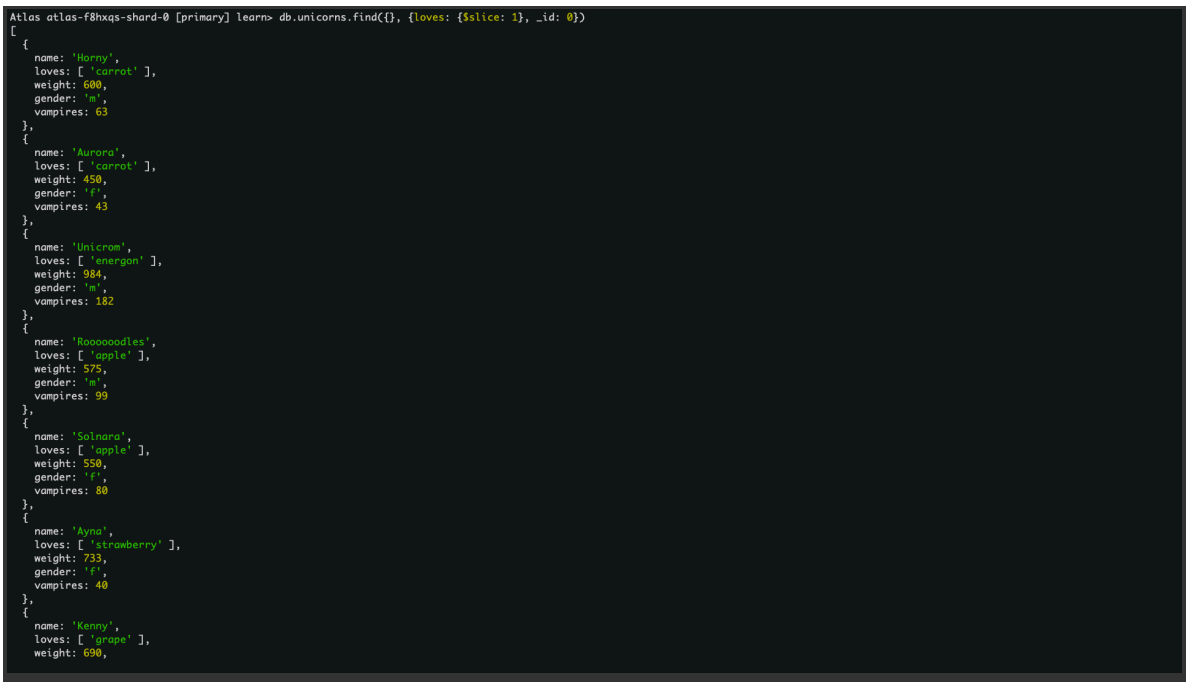
```
db.unicorns.find().sort({$natural: -1})
```



Задание 2.1.4

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

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



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})
```

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})
```

Задание 2.3.3

Найти всех единорогов, не имеющих ключ `vampires`

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

Задание 2.3.4

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

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({gender: "m"}, {name: 1, loves: {$slice: 1}}).sort({name: 1})
[
  {
    _id: ObjectId('68377a638b3a1dd66beb5421'),
    name: 'Dunx',
    loves: [ 'grape' ]
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5416'),
    name: 'Horny',
    loves: [ 'carrot' ]
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541c'),
    name: 'Kenny',
    loves: [ 'grape' ]
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541f'),
    name: 'Pilot',
    loves: [ 'apple' ]
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541d'),
    name: 'Raleigh',
    loves: [ 'apple' ]
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5419'),
    name: 'Roooooodles',
    loves: [ 'apple' ]
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5418'),
    name: 'Unicrom',
    loves: [ 'energon' ]
  }
]
```

Задание 3.1.1

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

1)

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.towns.find({"mayor.party": "I"}, {name: 1, mayor: 1, _id: 0})
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

2)

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

```
1, mayor: 1, _id: 0})
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.towns.find({"mayor.party": {$exists: false}}, {name: 1, mayor: 1, _id: 0})
[ { name: 'Punxsutawney', mayor: { name: 'Jim Wehrle' } } ]
```

Задание 3.1.2

1) Сформировать функцию для вывода списка самцов единорогов.

2) Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.

3) Вывести результат, используя forEach.

1)

```
db.unicorns.find({gender: "m"})
```

2)

```
var cursor = db.unicorns.find({gender: "m"}).sort({name:
1}).limit(2);null;
```

3)

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> var cursor = db.unicorns.find({gender: "m"});null;
... cursor.forEach(function(obj) {
...     print(obj.name);
...     })
...
Horny
Unicrom
Roooooodles
Kenny
Raleigh
Pilot
Dunx
```

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()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({gender: "f", weight: {$gte: 500, $lte: 600}}).count()
2
```

Задание 3.2.2

Вывести список предпочтений

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.distinct("loves")
[
  'apple',      'carrot',
  'chocolate',  'energon',
  'grape',      'lemon',
  'papaya',     'redbull',
  'strawberry', 'sugar',
  'watermelon'
]
```

Задание 3.2.3

Посчитать количество особей единорогов обоих полов.

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.aggregate([{$group: {_id: "$gender", count: {$sum: 1}}}])
[ { _id: 'm', count: 7 }, { _id: 'f', count: 5 } ]
```

Задание 3.3.1

1. Выполнить команду:

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

2. Проверить содержимое коллекции `unicorns`.

Вывод: в версии Монго, начиная с 3.2, метод save более не поддерживается

```
TypeError: db.unicorns.save is not a function
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

Задание 3.3.2

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

2. Проверить содержимое коллекции `unicorns`*.*

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb541b'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

Задание 3.3.3

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

2. Проверить содержимое коллекции `unicorns`*.*

```
db.unicorns.update({ name: "Raleigh", gender: "m" },
{$push: {loves: "рэдбул"}})
```

```
  weight: 550,
  gender: 'f',
  vampires: 80
},
{
  _id: ObjectId('68377a2b8b3a1dd66beb541b'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
{
  _id: ObjectId('68377a2c8b3a1dd66beb541c'),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 39
},
{
  _id: ObjectId('68377a2c8b3a1dd66beb541d'),
  name: 'Raleigh',
  loves: [ 'apple', 'sugar', 'рэдбул' ],
  weight: 421,
  gender: 'm',
  vampires: 2
},
{
  _id: ObjectId('68377a2c8b3a1dd66beb541e'),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
```

Задание 3.3.4

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

2. Проверить содержимое коллекции `unicorns`.

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({gender: 'm'})
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5416'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5418'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5419'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541c'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541d'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'рэдбул' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541f'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
```

Задание 3.3.5

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

2. Проверить содержимое коллекции `towns`.

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.towns.find()
[
  {
    _id: ObjectId('68386d358b3a1dd66beb5425'),
    name: 'Punxsutawney',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('68386d358b3a1dd66beb5426'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'statue of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('68386d358b3a1dd66beb5427'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

Задание 3.3.6

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

2. Проверить содержимое коллекции `unicorns`

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

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find({ name: "Pilot", gender: "m" })
[
  {
    _id: ObjectId('68377a2c8b3a1dd66beb541f'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

Задание 3.3.7

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

2. Проверить содержимое коллекции `unicorns`.

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

```
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5417'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
```

Задание 3.4.1

1. Удалите документы с беспартийными мэрами.

2. Проверьте содержание коллекции.

3. Очистите коллекцию.

4. Просмотрите список доступных коллекций.

```
db.towns.remove({"mayor.party": {$exists: false}},
{multi: true});
db.towns.remove({});
db.getCollectionNames();
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.towns.remove({"mayor.party": {$exists: false}}, {multi: true});
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 1 }
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.towns.find()
[
  {
    _id: ObjectId('68386d358b3a1dd66beb5426'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'statue of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('68386d358b3a1dd66beb5427'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.towns.remove({});
{ acknowledged: true, deletedCount: 2 }
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.getCollectionNames();
[ 'unicorns', 'users', 'towns' ]
```

Вывод: так как в условии было сказано очистить, а не удалить коллекцию, все верно

Задание 4.1.1

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

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

3. Проверьте содержание коллекции едиорогов.

4. Содержание коллекции единорогов `unicorns`:

1) Отправляем Хорни единорога в леса Сибири:

```
> db.habitats.insert({_id: "forest", name: "Siberia Forest", desc: "severe russian forest"})

> db.unicorns.update({name: 'Horny'}, {$set: {habitat: {$ref: "habitats", $id: "forest"}}})
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.habitats.insert({_id: "forest", name: "Siberia Forest", desc: "severe russian forest"})
{ acknowledged: true, insertedIds: { '0': 'forest' } }
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.update({name: 'Horny'}, {$set: {habitat: {$ref: "habitats", $id: "forest"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2) Аврору поселим в горах Кавказа:

```
> db.habitats.insert({_id: "mount", name: "Kavkaz mountains", desc: "beautiful fr"})
> db.unicorns.update({name: 'Aurora'}, {$set: {habitat: {$ref: "habitats", $id: "mount"}}})
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.habitats.insert({_id: "mount", name: "Kavkaz mountains", desc: "beautiful fr"})
{ acknowledged: true, insertedIds: { '0': 'mount' } }
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.update({name: 'Aurora'}, {$set: {habitat: {$ref: "habitats", $id: "mount"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.find()
[
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5416'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habitat: DBRef('habitats', 'forest')
  },
  {
    _id: ObjectId('68377a2b8b3a1dd66beb5417'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat: DBRef('habitats', 'mount')
  },
```

Задание 4.2.1

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

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

Вывод: можно

Задание 4.3.1 — Работа с индексами

1. Получите информацию о всех индексах коллекции `unicorns` .

2. Удалите все индексы, кроме индекса для идентификатора.

3. Попытайтесь удалить индекс для идентификатора.

1)

```
db.unicorns.getIndexes()
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
```

2)

```
db.unicorns.dropIndexes()
```

Не затрагивает системный индекс *id*.

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.dropIndexes()
...
...
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1748555289, i: 6 }),
    signature: {
      hash: Binary.createFromBase64('8E63YD/1p1eDbBrf+Cm6oAbAl1w=', 0),
      keyId: Long('7465773399587946512')
    }
  },
  operationTime: Timestamp({ t: 1748555289, i: 6 })
}
```

3)

```
db.unicorns.dropIndex("_id_")
```

```
Atlas atlas-f8hxqs-shard-0 [primary] learn> db.unicorns.dropIndex("_id_")
...
...
MongoServerError[InvalidOptions]: cannot drop _id index
Atlas atlas-f8hxqs-shard-0 [primary] learn>
```

Вывод - так нельзя

Задание 4.4.1

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

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

2. Выберите последних четыре документа.

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

4. Создайте индекс для ключа `value`.

5. Получите информацию о всех индексах коллекции `numbres`.

6. Выполните запрос 2.

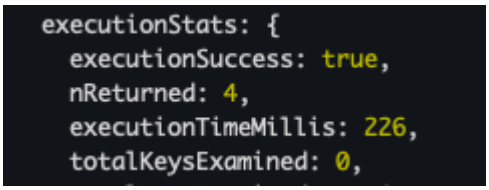
7. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

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

2)

```
db.numbers.find().sort({value:
-1}).limit(4).explain("executionStats")
```

3)



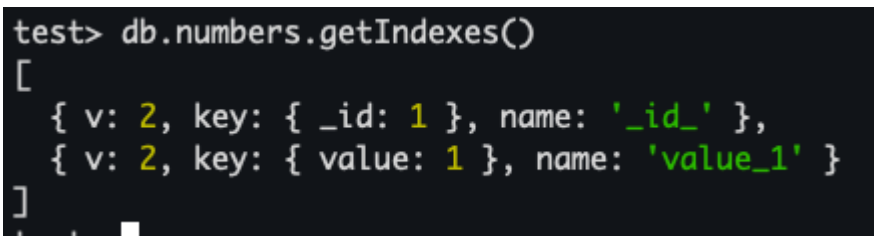
Запрос отработал за 226 мс

4)

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

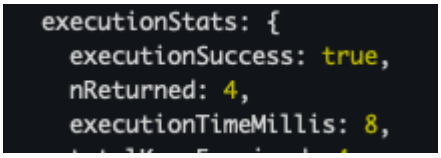
5)

```
db.numbers.getIndexes()
```



6)

```
db.numbers.find().sort({value:
-1}).limit(4).explain("executionStats")
```



Потребовалось - 10мс, разница существенная

Вывод: при работе с большими коллекциями, индекс заметно оптимизирует время отрабатывания запросов.

Выводы

В ходе лабораторной работы были изучены:

- CRUD-операции в MongoDB;
- Выборка и фильтрация данных;
- Использование агрегатов и курсоров;
- Работа со вложенными объектами и массивами;
- Использование индексов и связей между коллекциями.