Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное
учреждение высшего образования
“Национальный исследовательский университет ИТМО”

Факультет инфокоммуникационных технологий

ЛАБОРАТОРНАЯ РАБОТА №5

Работа с БД в СУБД MongoDB по дисциплине: «Проектирование и реализация баз данных»

Выполнил студент:
Зайцев Кирилл Дмитриевич
Группа №K33402
Преподаватель:
Говорова Марина Михайловна

Санкт-Петербург
2023

Цель работы:

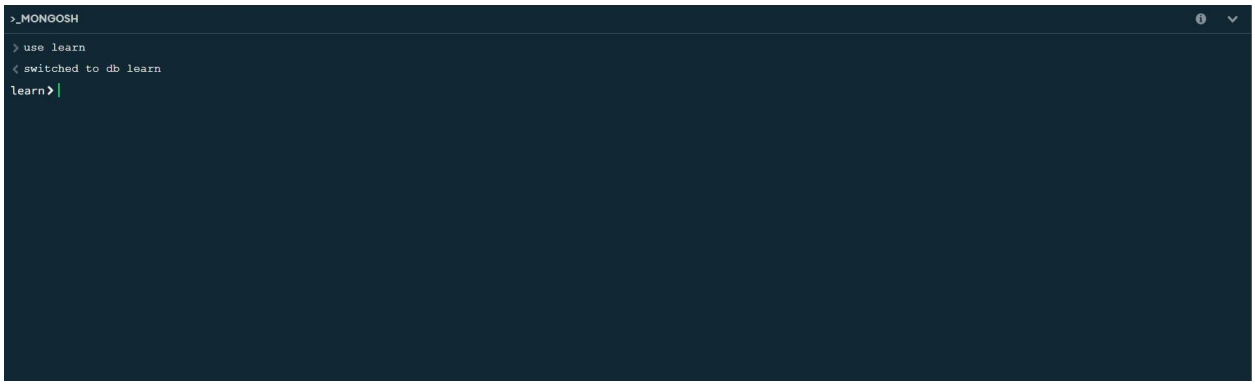
Овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

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

Ход работы:

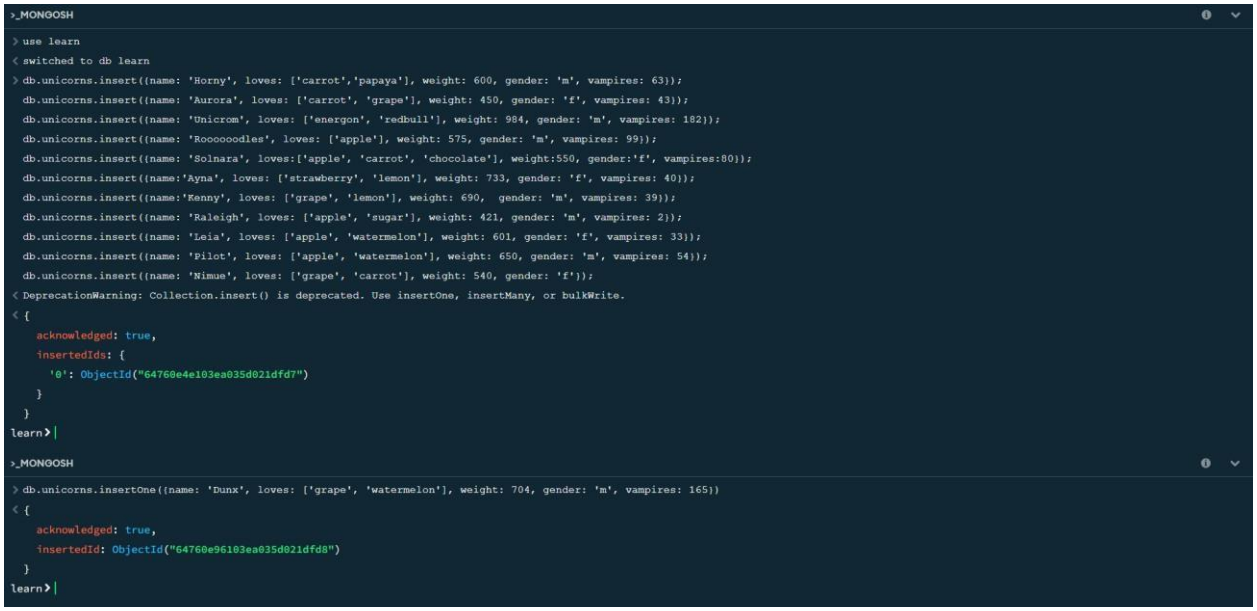
Практическое задание 8.1.1

1. Создайте базу данных learn.



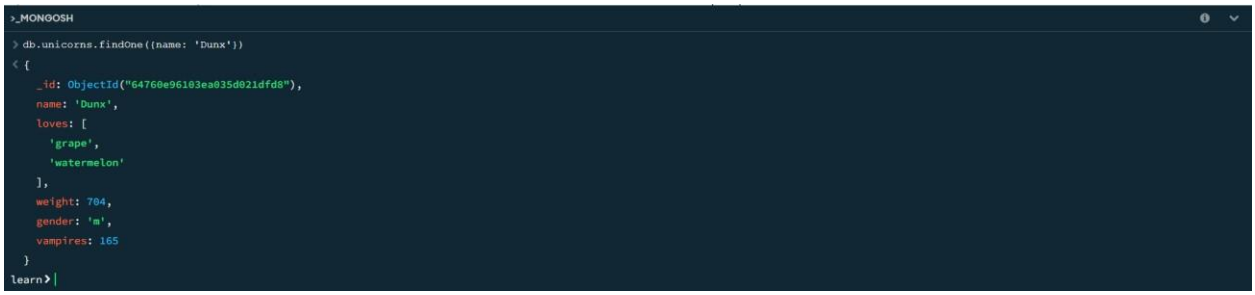

```
use learn
```

2. Заполните коллекцию единорогов unicorns:




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

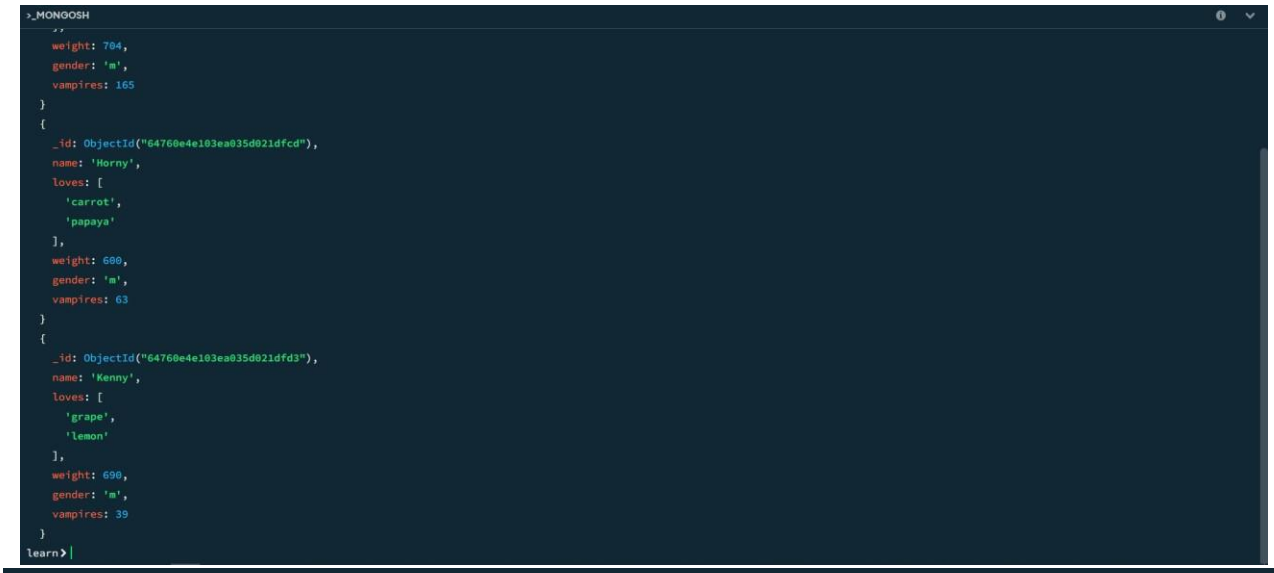
3. Проверьте содержимое коллекции с помощью метода find.



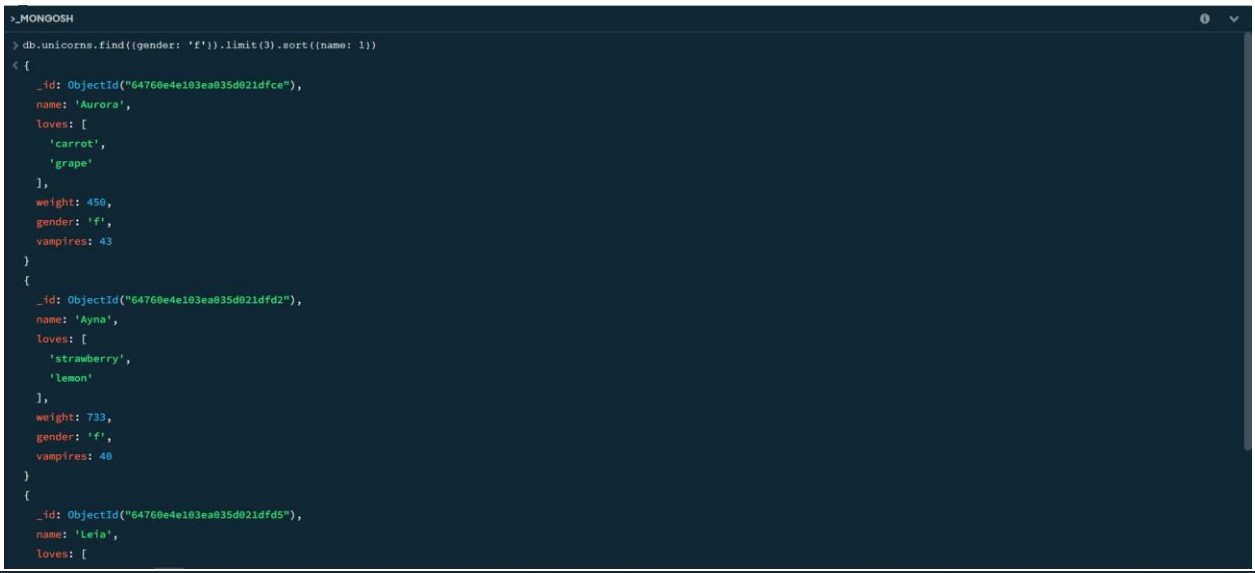

```
{name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
```

Практическое задание 8.1.2

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




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



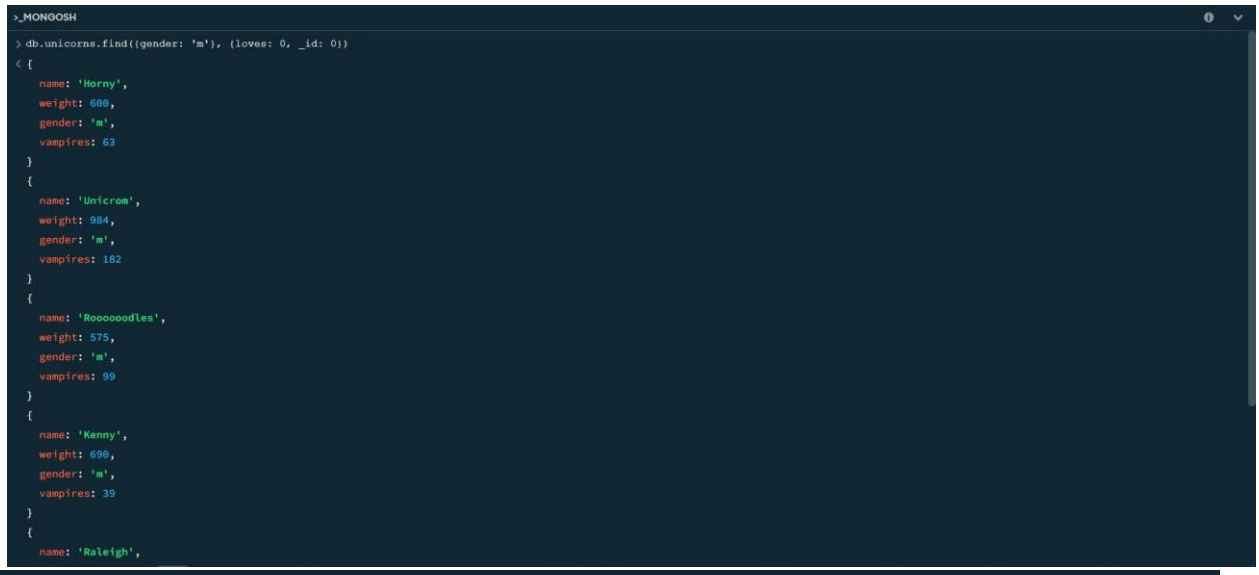

```
db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
```

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


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

Практическое задание 8.1.3

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




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

Практическое задание 8.1.4

Вывести список единорогов в обратном порядке добавления.


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

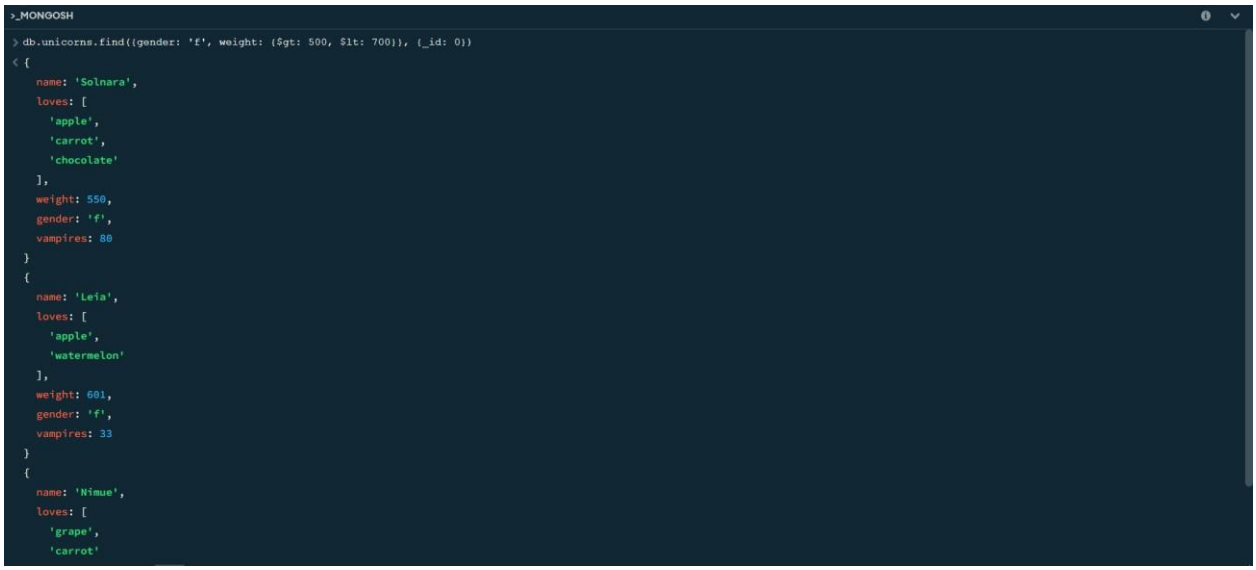
Практическое задание 8.1.5

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


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

Практическое задание 8.1.6

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




```
db.unicorns.find({gender: 'f', weight: {$gt: 500, $lt: 700}}, {_id: 0})
```

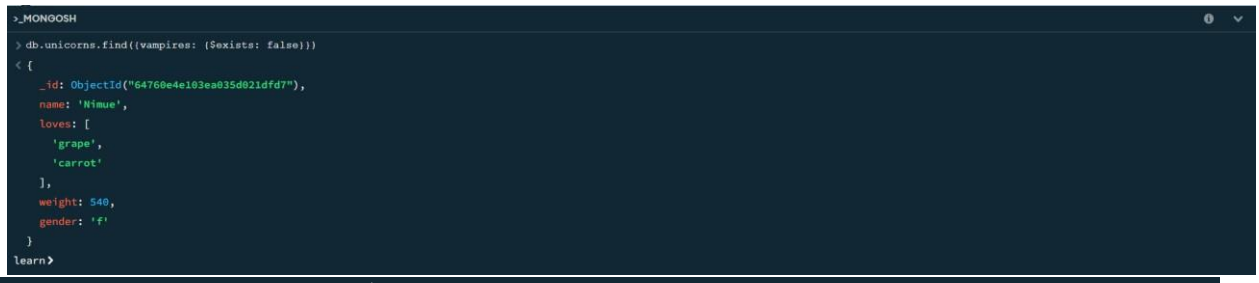
Практическое задание 8.1.7

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


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

Практическое задание 8.1.8

Найти всех единорогов, не имеющих ключ vampires.




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

Практическое задание 8.1.9

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

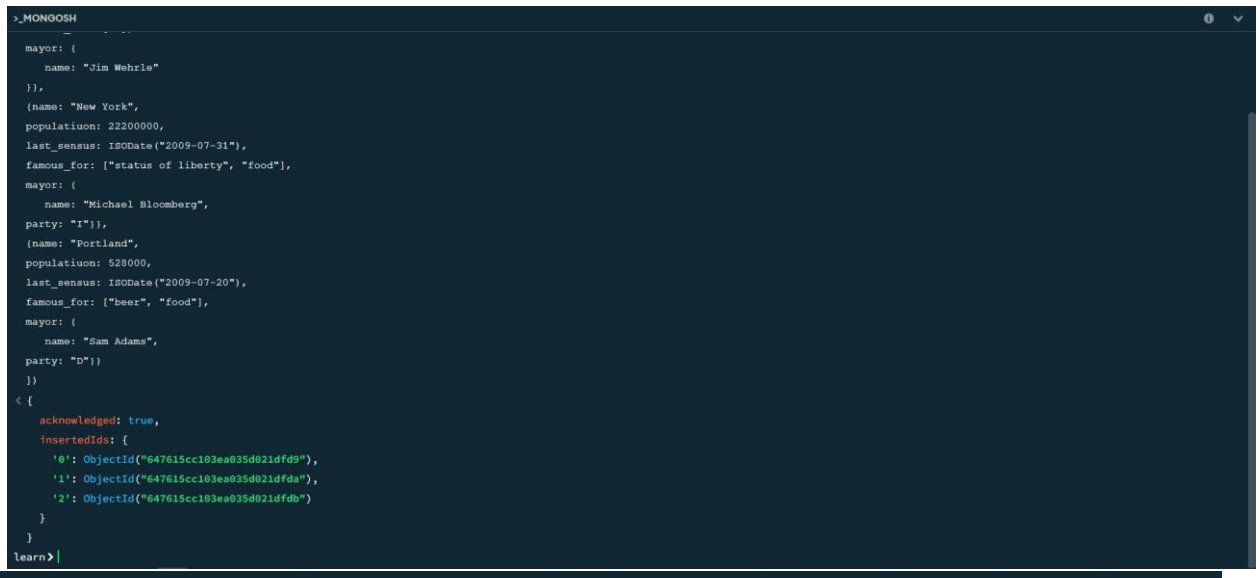

```
db.unicorns.find({gender: 'm'}, {_id: 0, name: 1, loves: {$slice:
1}}).sort({name: 1})
```

Практическое задание 8.2.1

1. Создайте коллекцию towns, включающую следующие документы:

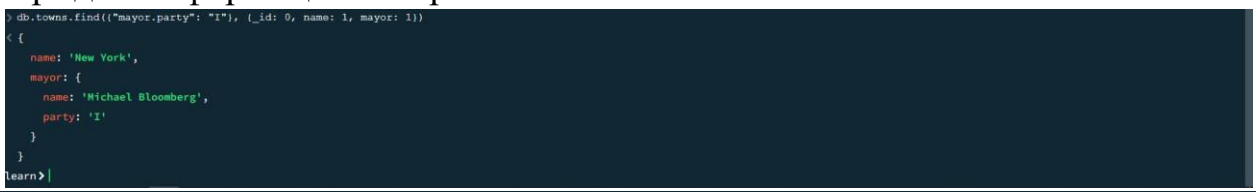
```
{name: "Punxsutawney ",
populatiuon: 6200,  last_sensus:
```

```
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"}}
```



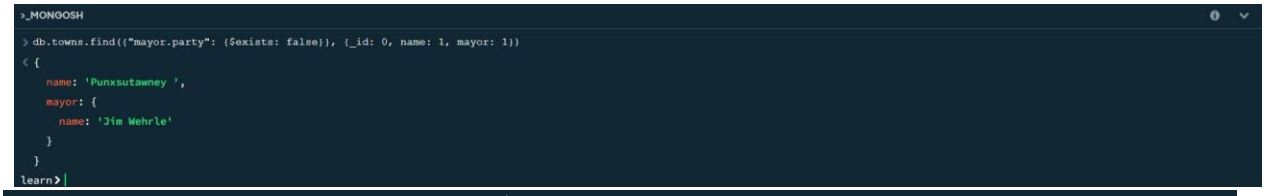

```
db.towns.insertMany([ {name:
"Punxsutawney ",
populatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: [""], mayor: {
   name: "Jim Wehrle"
}},
{name: "New York", populatiuon:
22200000,
last_sensus: ISODate("2009-07-31"), famous_for:
["status of liberty", "food"], mayor: {
   name: "Michael Bloomberg",
party: "I"}}, {name:
"Portland", populatiuon:
528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"], mayor:
{
   name: "Sam Adams", party:
"D"}}
])
```

2. 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})
```

3. 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})
```

Практическое задание 8.2.2

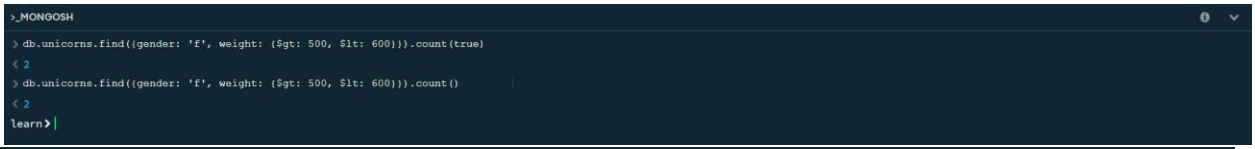
1. Сформировать функцию для вывода списка самцов единорогов.


```
let logUnicorns = function(cursor) {
cursor.forEach((el) => {
          print(el.name)
 })
};
let cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2); null;

logUnicorns(cursor)
```

2. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.
3. Вывести результат, используя forEach.

Практическое задание 8.2.3

Вывести количество самок единорогов весом от полутонны до 600 кг.




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

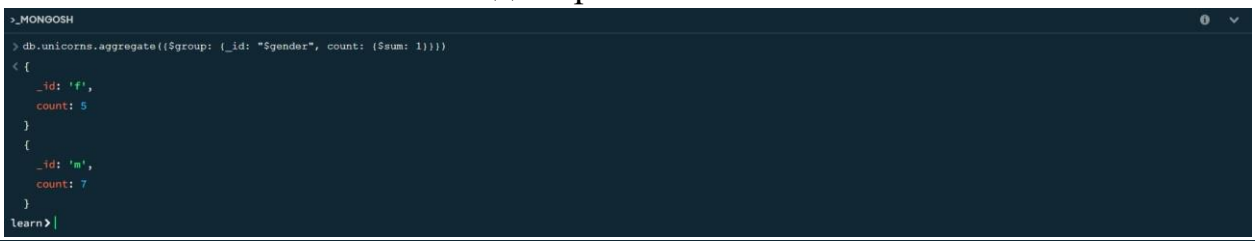
Практическое задание 8.2.4

Вывести список предпочтений.


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

Практическое задание 8.2.5

Посчитать количество особей единорогов обоих полов.



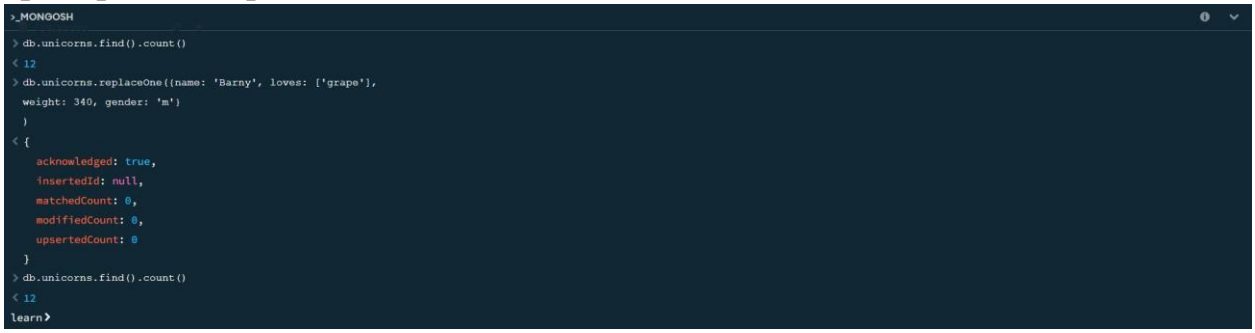

```
db.unicorns.aggregate({$group: {_id: "$gender", count: {$sum: 1}}})
```

Практическое задание 8.2.6

1. Выполнить команду:

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

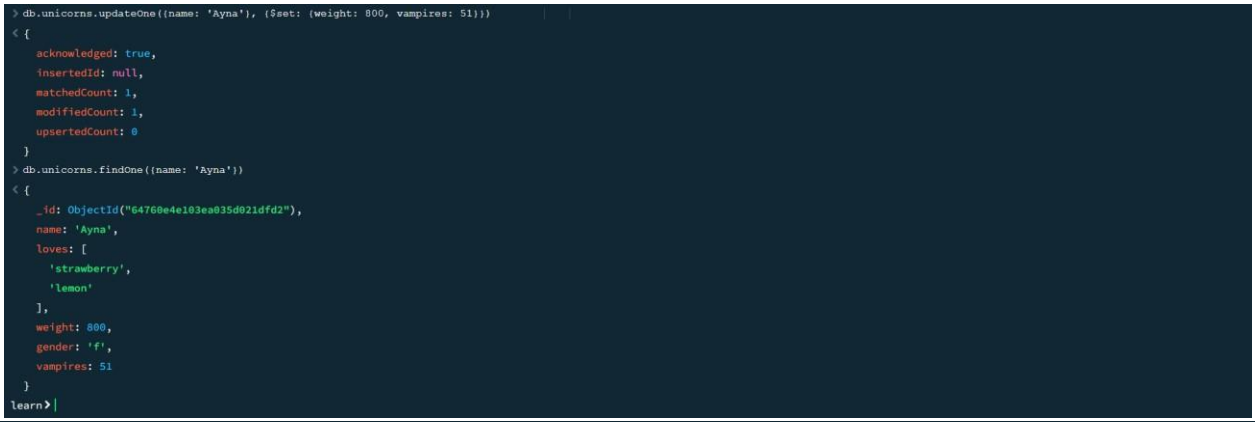
Проверить содержимое коллекции unicorns.




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

Практическое задание 8.2.7

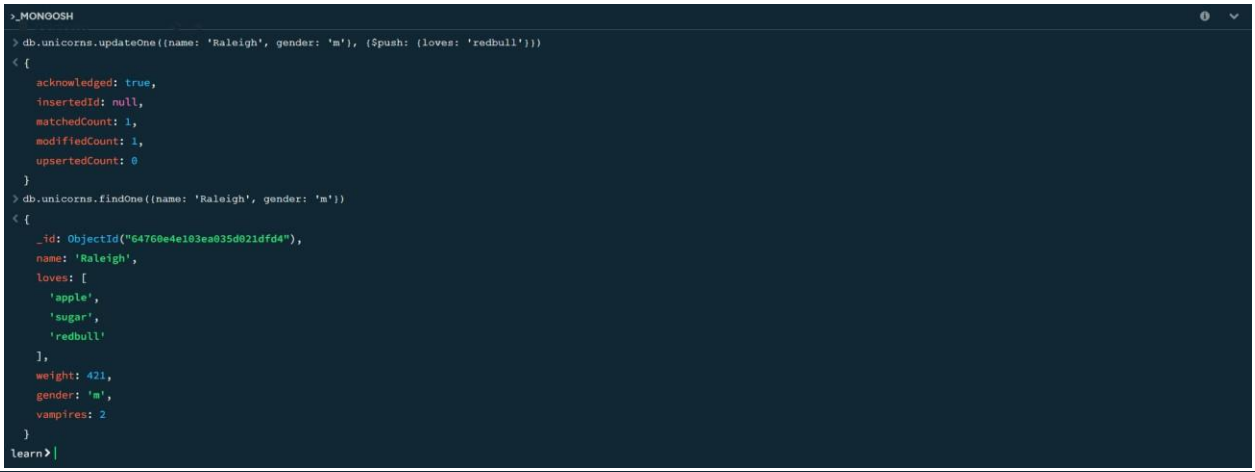
Для самки единорога Ayna внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.




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

Практическое задание 8.2.8

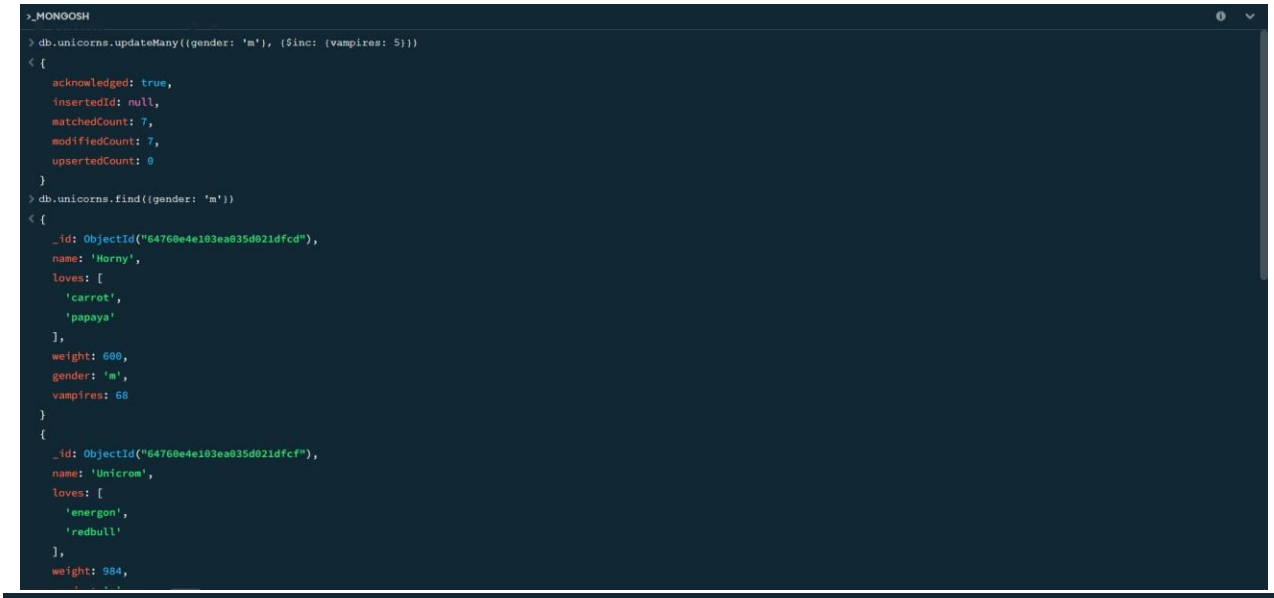
1. Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.




```
db.unicorns.updateOne({name: 'Raleigh', gender: 'm'}, {$push: {loves: 'redbull'}})
```

Практическое задание 8.2.9

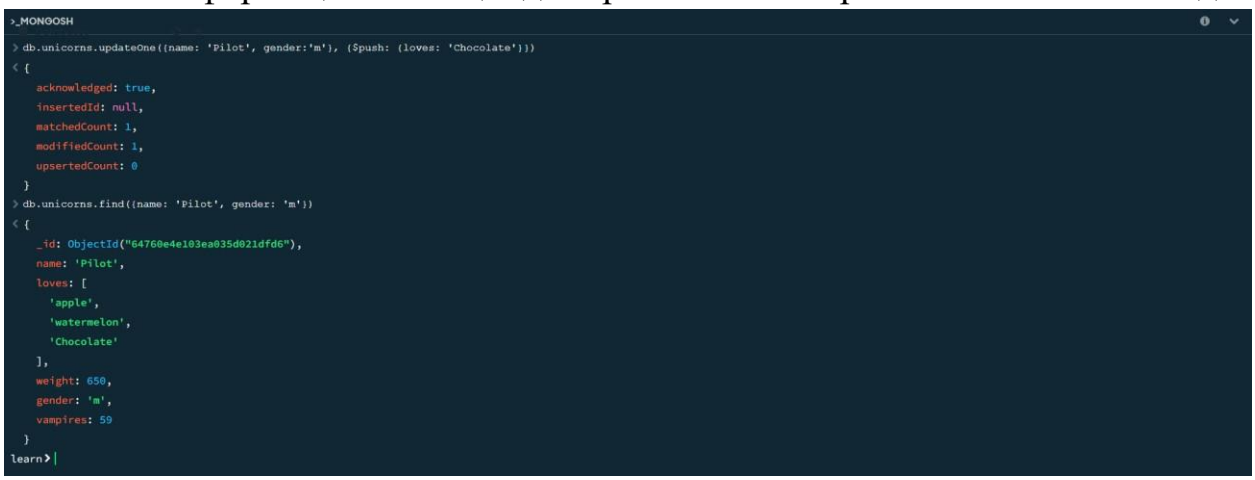
Всем самцам единорогов увеличить количество убитых вапмиров на 5.




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

Практическое задание 8.2.11

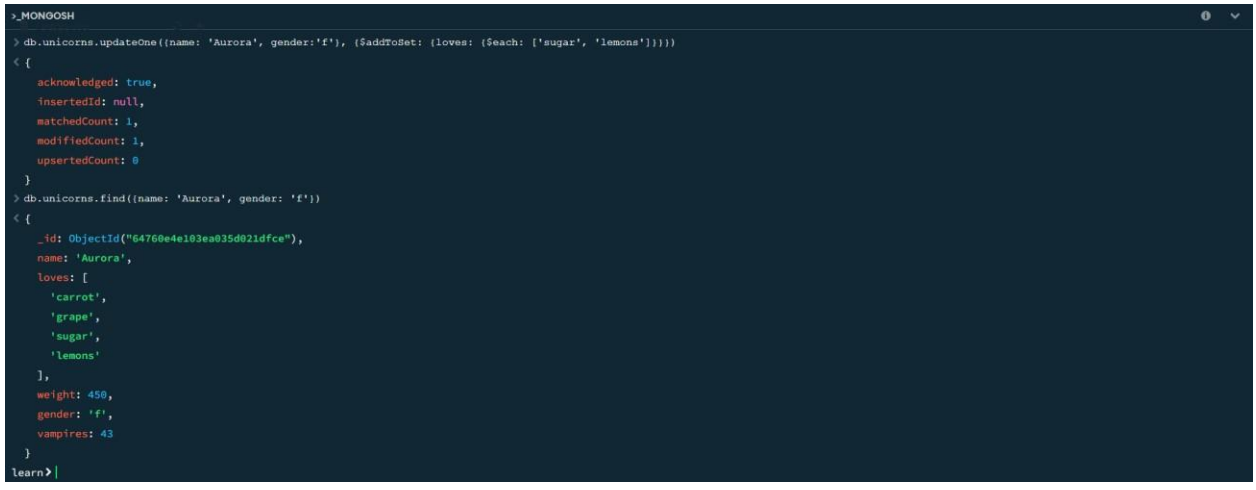
Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.




```
db.unicorns.updateOne({name: 'Pilot', gender:'m'}, {$push: {loves:
'Chocolate'}})
```

Практическое задание 8.1.12

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



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

Практическое задание 8.1.13

1. Удалите документы с беспартийными мэрами.

```
>_MONGOSH
> db.towns.deleteMany({"mayor.party": {$exists: false}})
< {
    acknowledged: true,
    deletedCount: 1
  }
> db.towns.find({})
< {
    _id: ObjectId("64772c310997c99920fab103"),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: 2009-07-31T00:00:00.000Z,
    famous_for: [
      'status of liberty',
      'food'
    ],
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
  {
    _id: ObjectId("64772c310997c99920fab104"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
```

2.

3.
```
db.towns.deleteMany({"mayor.party": {$exists: false}})
```

4. Проверьте содержание коллекции.
5. Очистите коллекцию.

```
>_MONGOSH
> db.towns.deleteMany({})
< {
    acknowledged: true,
    deletedCount: 2
  }
> show collections
< towns
  unicorns
learn>
```

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

6. Просмотрите список доступных коллекций.

Практическое задание 8.3.1

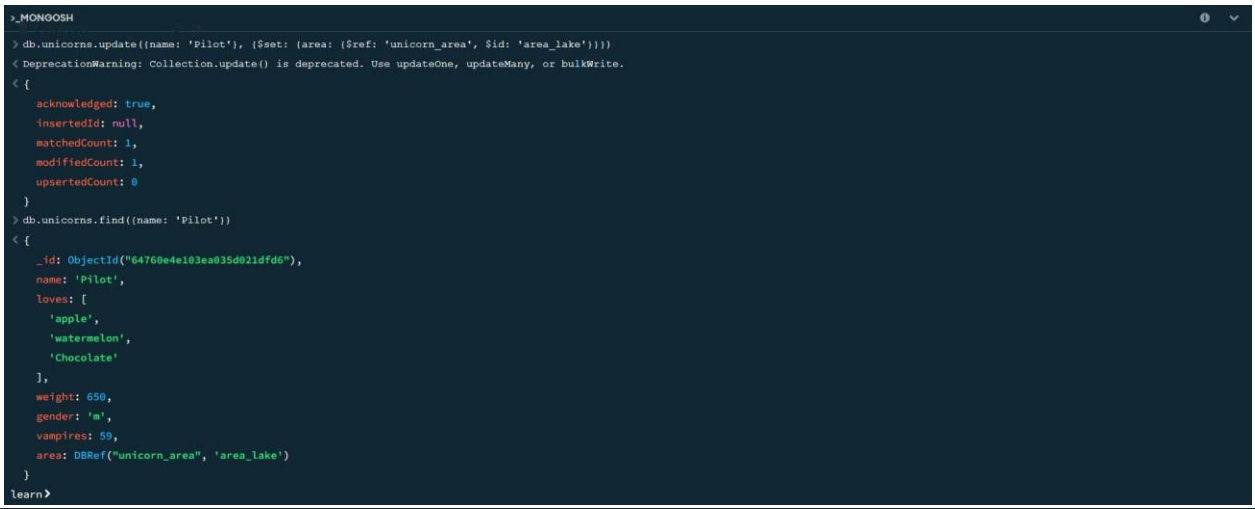
1. Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.

```
>_MONGOSH
> db.unicorn_area.insertMany([
    {_id: 'areal_field', name: 'Field', description: 'description.........'},
    {_id: 'areal_forest', name: 'Forest', description: 'description.........'},
    {_id: 'areal_lake', name: 'Lake', description: 'description.........'},
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': 'areal_field',
      '1': 'areal_forest',
      '2': 'areal_lake'
    }
  }
learn>
```

```
db.unicorn_area.insertMany([
 {_id: 'areal_field', name: 'Field', description: 'description.........'},
```

```
{_id: 'areal_forest', name: 'Forest', description: 'description.........'},
{_id: 'areal_lake', name: 'Lake', description: 'description.........'}, ])
```

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




```
db.unicorns.update({name: 'Pilot'}, {$set: {area: {$ref: 'unicorn_area', $id:
'area_lake'}}})
```

3. Проверьте содержание коллекции едиорогов.

Практическое задание 8.3.2

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


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

Практическое задание 8.3.3

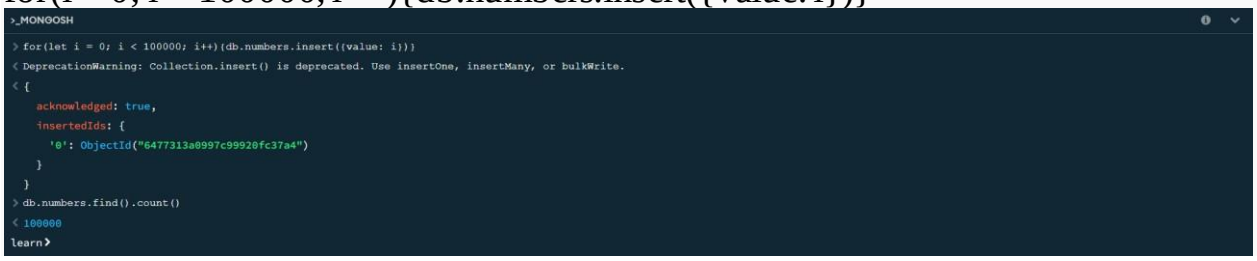
1. Получите информацию о всех индексах коллекции unicorns .
2. Удалите все индексы, кроме индекса для идентификатора.
3. Попытайтесь удалить индекс для идентификатора.


```
db.unicorns.getIndexes() db.unicorns.dropIndex('name_1')
```

Практическое задание 8.3.4

1. Создайте объемную коллекцию numbers, задействовав курсор:

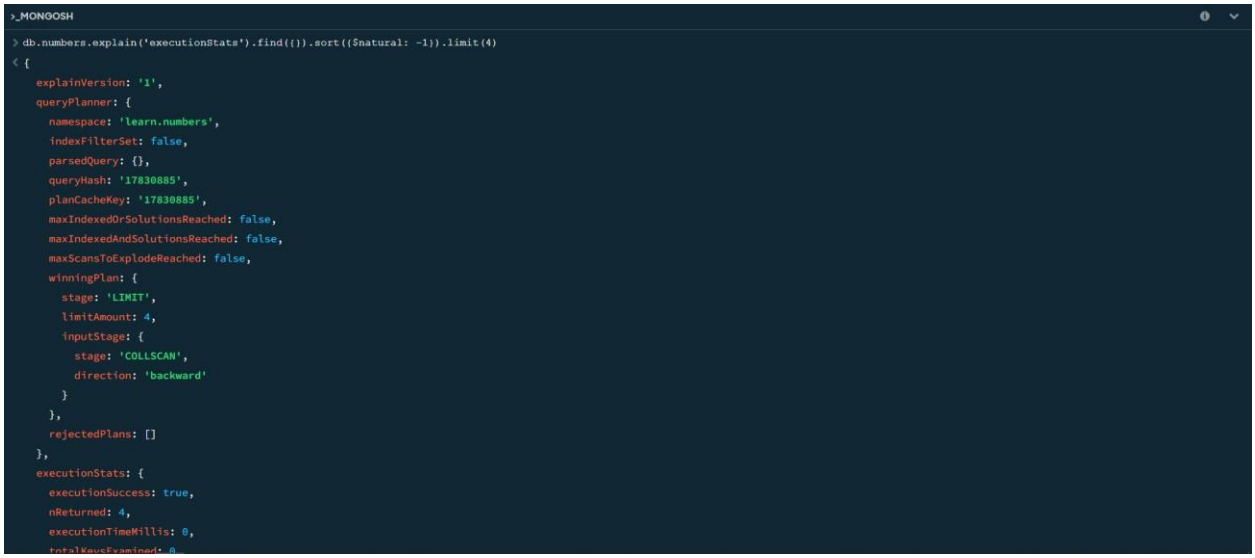
```
for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
```



2. Выберите последних четыре документа.
3. Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis) Запрос занял 0мс


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

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?

Запрос также занял 0 мс

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

Время выполнения запроса не изменилось. Возможно в этом случае индекс не ускоряет выполнение запроса.

Выводы:

1. Были произведены операции вставки, удаления, обновления, выборки данных.
2. Были созданы связи между объектами разных коллекций с помощью DBRef.
3. Были созданы индексы, сравнены запросы до и после применения индексов.