Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

По лабораторной работе №5

«**РАБОТА С БД В СУБД MONGODB**»

Автор: Ле Хоанг Чыонг

Факультет: ИКТ

Группа: К32392

Преподаватель: Говорова М. М.

Санкт-Петербург, 2023

1. Описание работы

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).

2. Выполнение работы

Практическое задание 8.1.1:

1. *Создайте базу данных learn.*
2. *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'});
```

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.*

```
> use learn
< switched to db learn
> db.createCollection("unicorns")
< { ok: 1 }
> db.unicorns.insertMany([
    {name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63},
    {name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43},
    {name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182},
    {name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99},
    {name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80},
    {name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40},
    {name:'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39},
    {name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2},
    {name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33},
    {name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54},
    {name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'}
  ]);
```

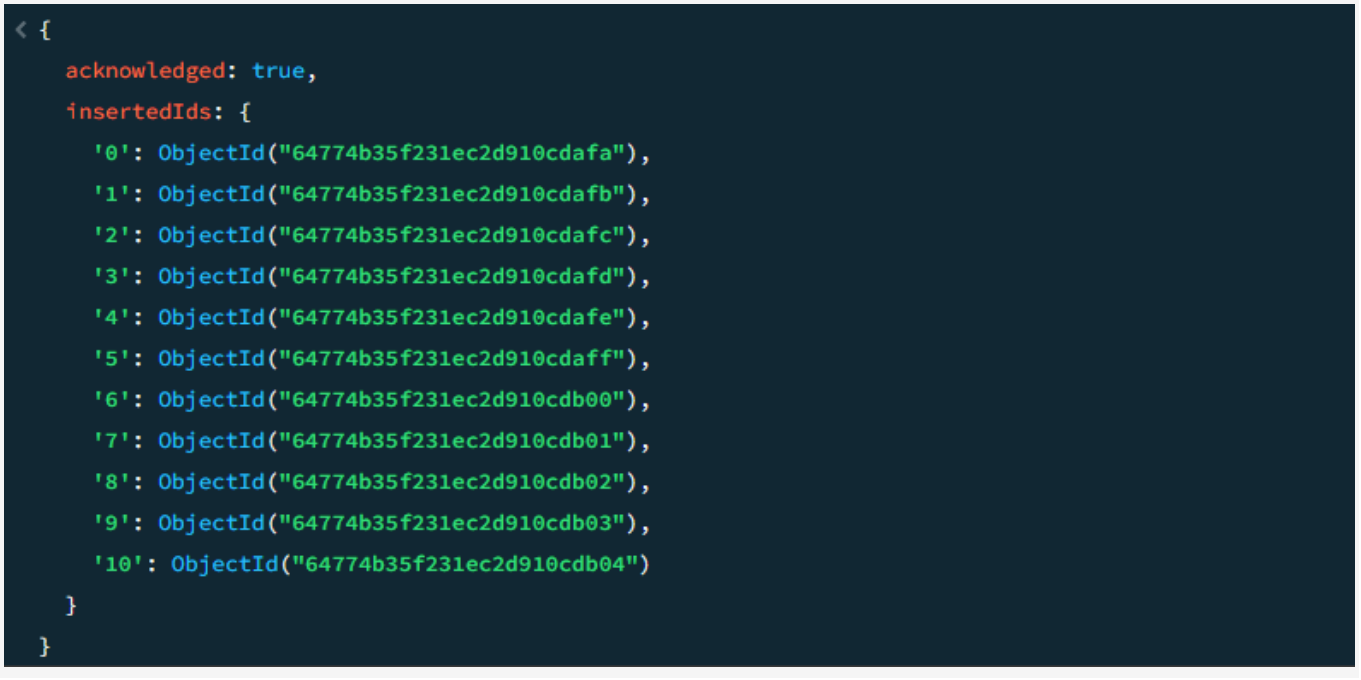

Рисунок 2 – Создание коллекции

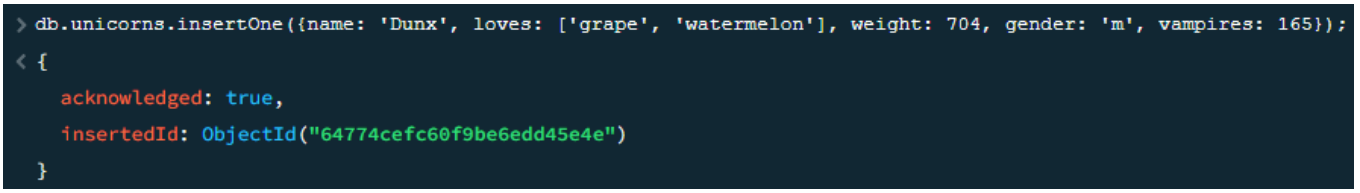

Рисунок 3 – Вставьте документ в коллекцию единорогов:

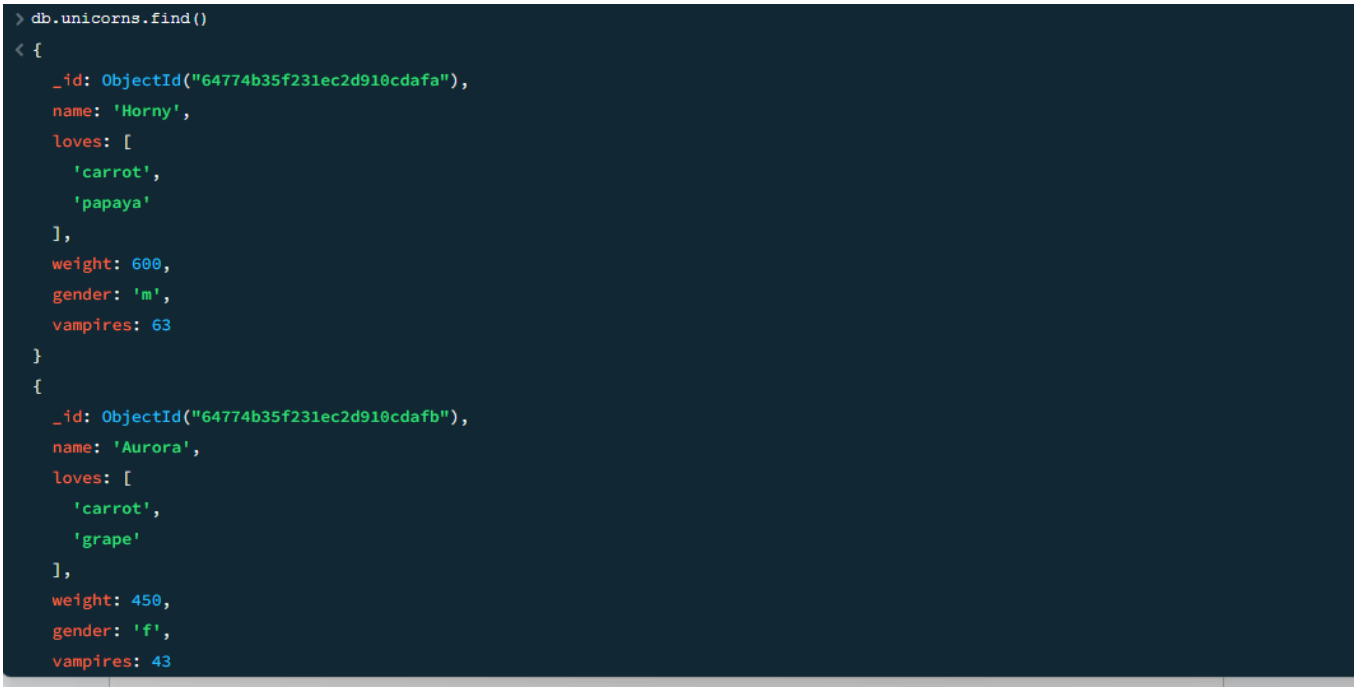

Рисунок 4 – Проверьте содержимое коллекции с помощью метода find

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

- *Запрос для вывода списка самцов единорогов, отсортированного по имени:*

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

```
> db.unicorns.find({gender: 'm'}).sort({name: 1})
< {
    _id: ObjectId("64774cefc60f9be6edd45e4e"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("64774b35f231ec2d910cdafa"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
```

Рисунок 4 – Запрос для вывода списка самцов единорогов, отсортированного по имени

- *Запрос для вывода списка самок единорогов, отсортированного по имени и ограниченного первыми тремя особями:*

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

```
> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
< {
    _id: ObjectId("64774b35f231ec2d910cdafb"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("64774b35f231ec2d910cdaff"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
```

Рисунок 5 – Запрос для вывода списка самок единорогов, отсортированного по имени и ограниченного первыми тремя особями

- *Запрос для вывода списка всех самок, которые любят carrot:*

```
db.unicorns.find({loves: 'carrot', gender: 'f'}).sort({name: 1})
```

```
> db.unicorns.find({loves: 'carrot', gender: 'f'}).sort({name: 1})
< {
    _id: ObjectId("64774b35f231ec2d910cdafb"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("64774b35f231ec2d910cdb04"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("64774b35f231ec2d910cdafe"),
    name: 'Solnara',
```

Рисунок 6 – Запрос для вывода списка всех самок, которые любят carrot

- *Запрос для вывода первой особи из списка всех самок, которые любят carrot:*

```
db.unicorns.findOne({gender: 'f', loves: 'carrot'})
```

```
> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
< {
    _id: ObjectId("64774b35f231ec2d910cdafb"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Рисунок 7 – Запрос для вывода первой особи из списка всех самок, которые любят carrot

Практическое задание 8.1.3:

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

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

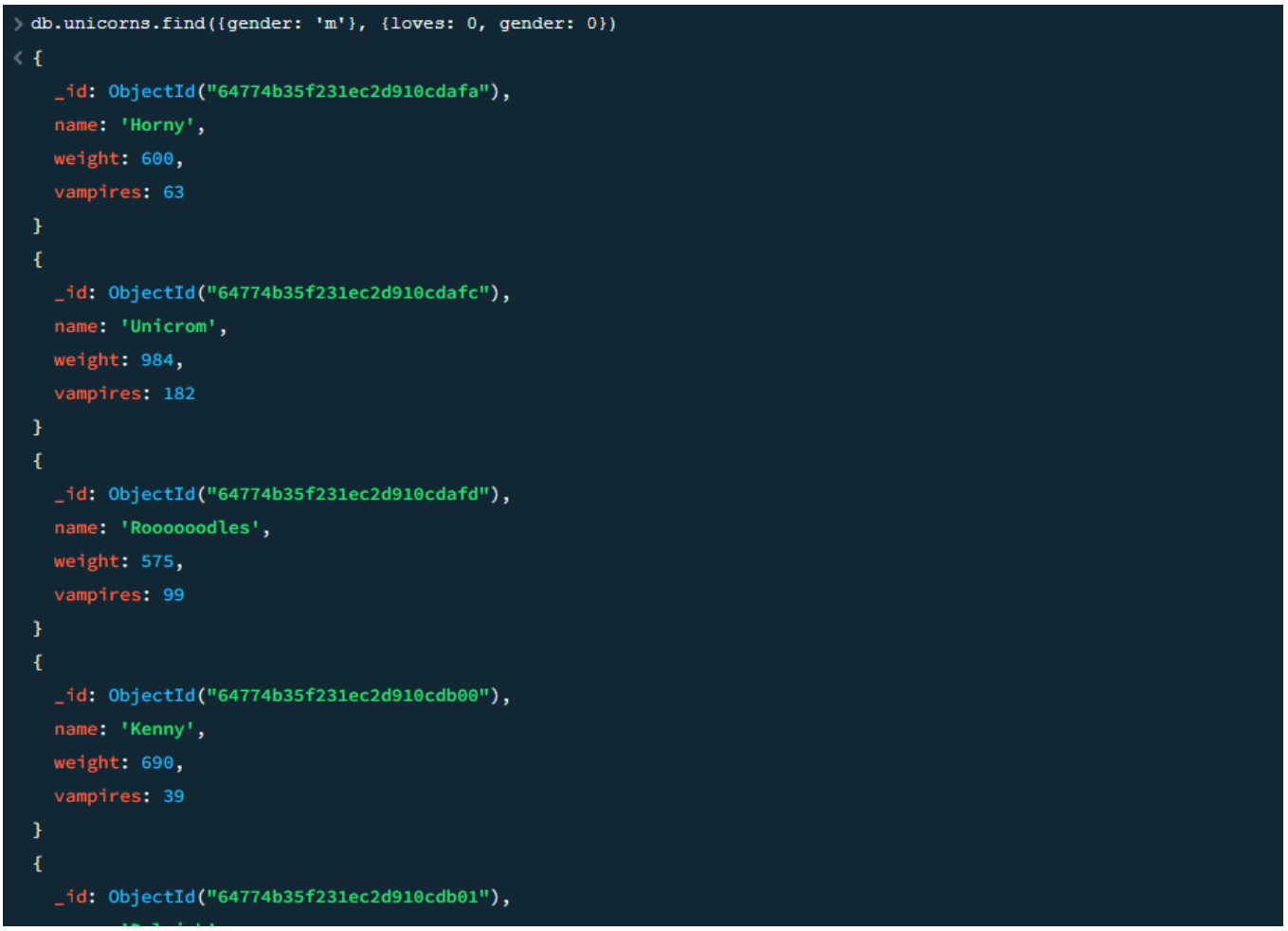

Рисунок 8 – Выполнение запроса 8.1.3

Практическое задание 8.1.4:

1. *Вывести список единорогов в обратном порядке добавления.*

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

```
> db.unicorns.find().sort({$natural: -1})
< {
    _id: ObjectId("64774cefc60f9be6edd45e4e"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("64774b35f231ec2d910cdb04"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("64774b35f231ec2d910cdb03"),
```

Рисунок 9 – Выполнение запроса 8.1.4

Практическое задание 8.1.5:

1. *Вывести список единорогов в обратном порядке добавления.*

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

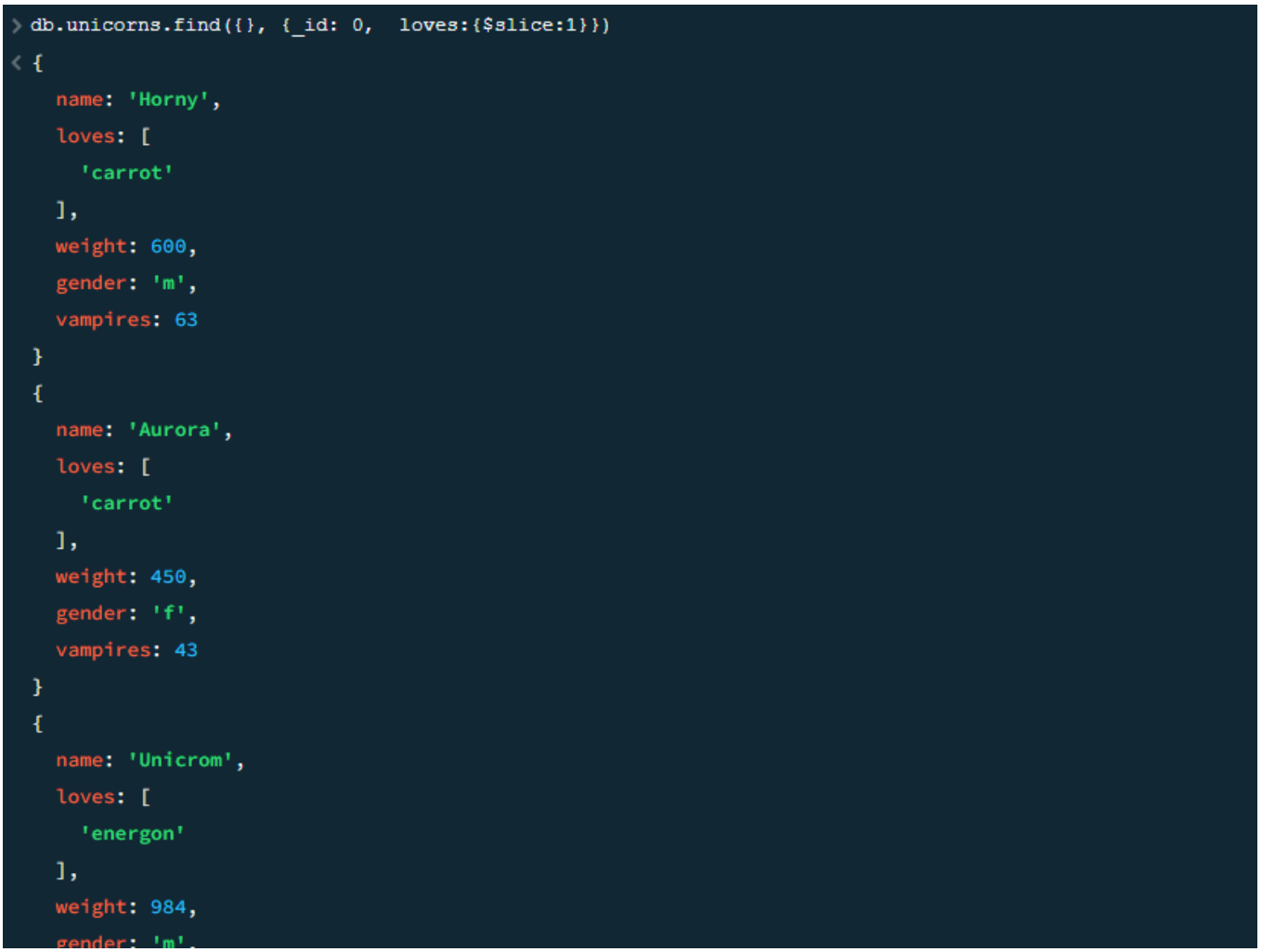

Рисунок 10 – Выполнение запроса 8.1.5

Практическое задание 8.1.6:

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})
```

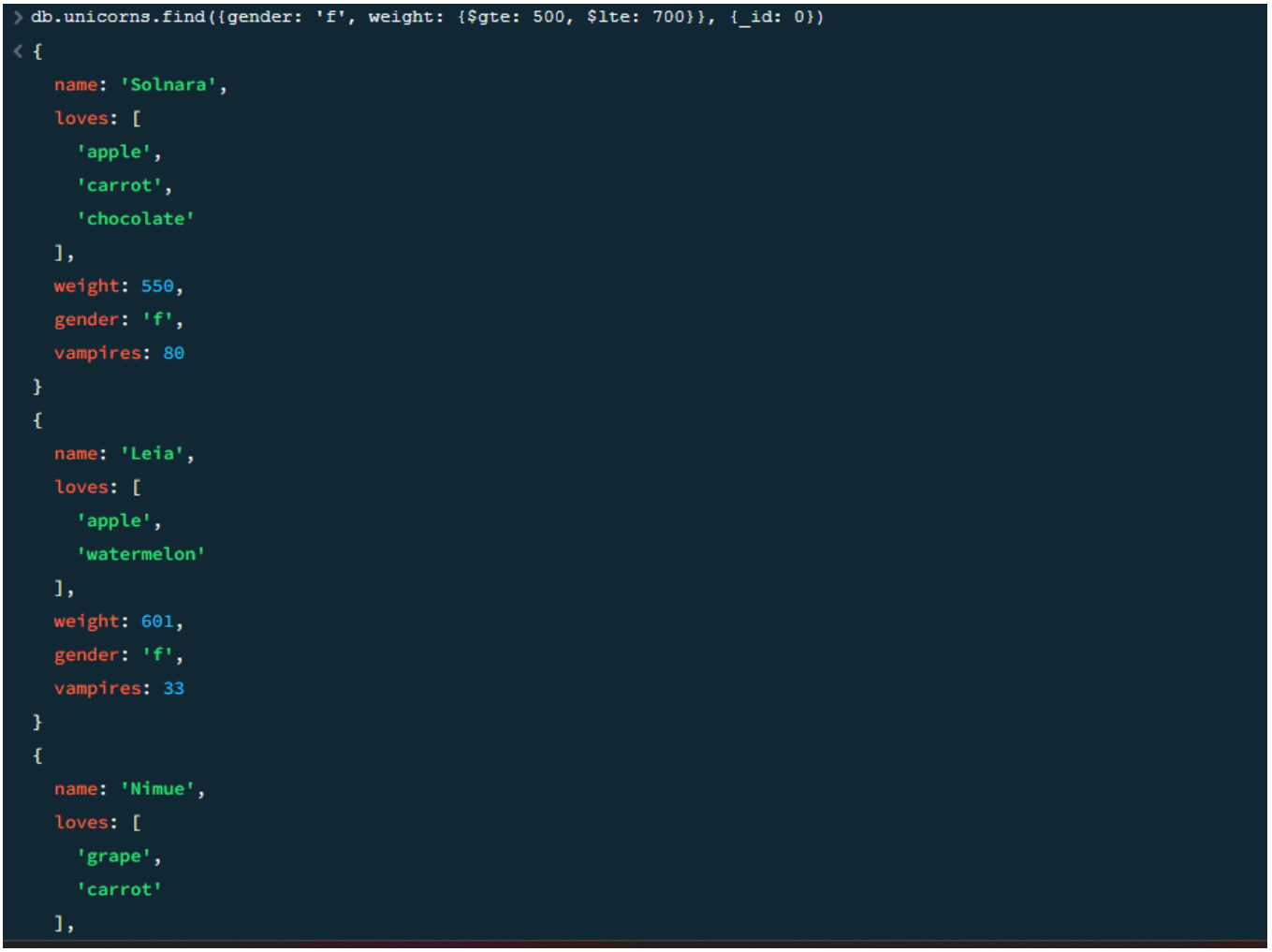

Рисунок 11 – Выполнение запроса 8.1.6

Практическое задание 8.1.7:

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

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

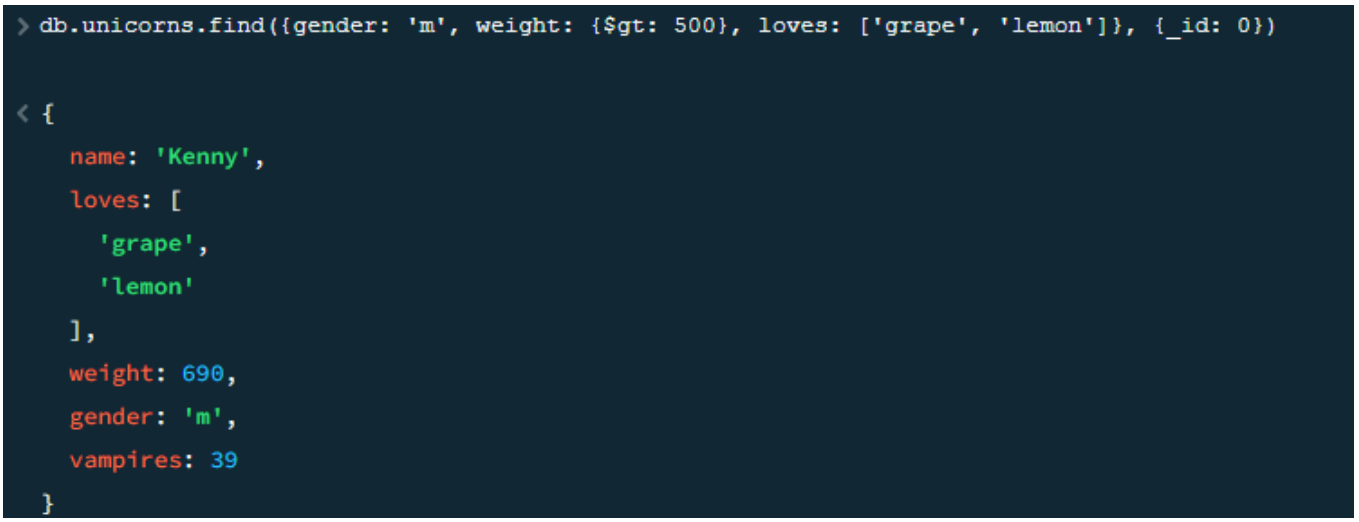

Рисунок 12 – Выполнение запроса 8.1.7

Практическое задание 8.1.8:

1. *Найти всех единорогов, не имеющих ключ `vampires`.*

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

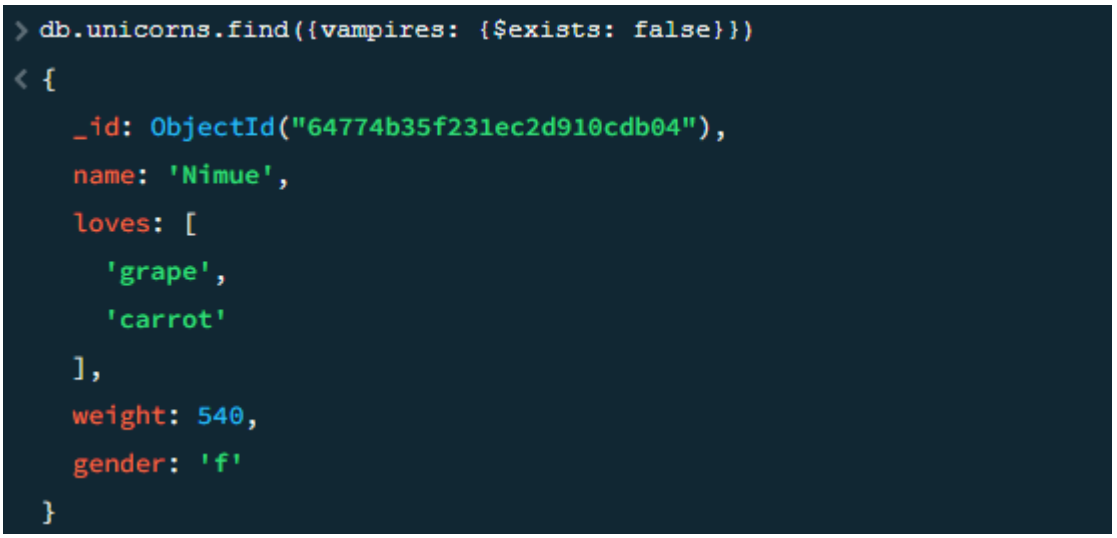

Рисунок 13 – Выполнение запроса 8.1.8

Практическое задание 8.1.9:

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

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

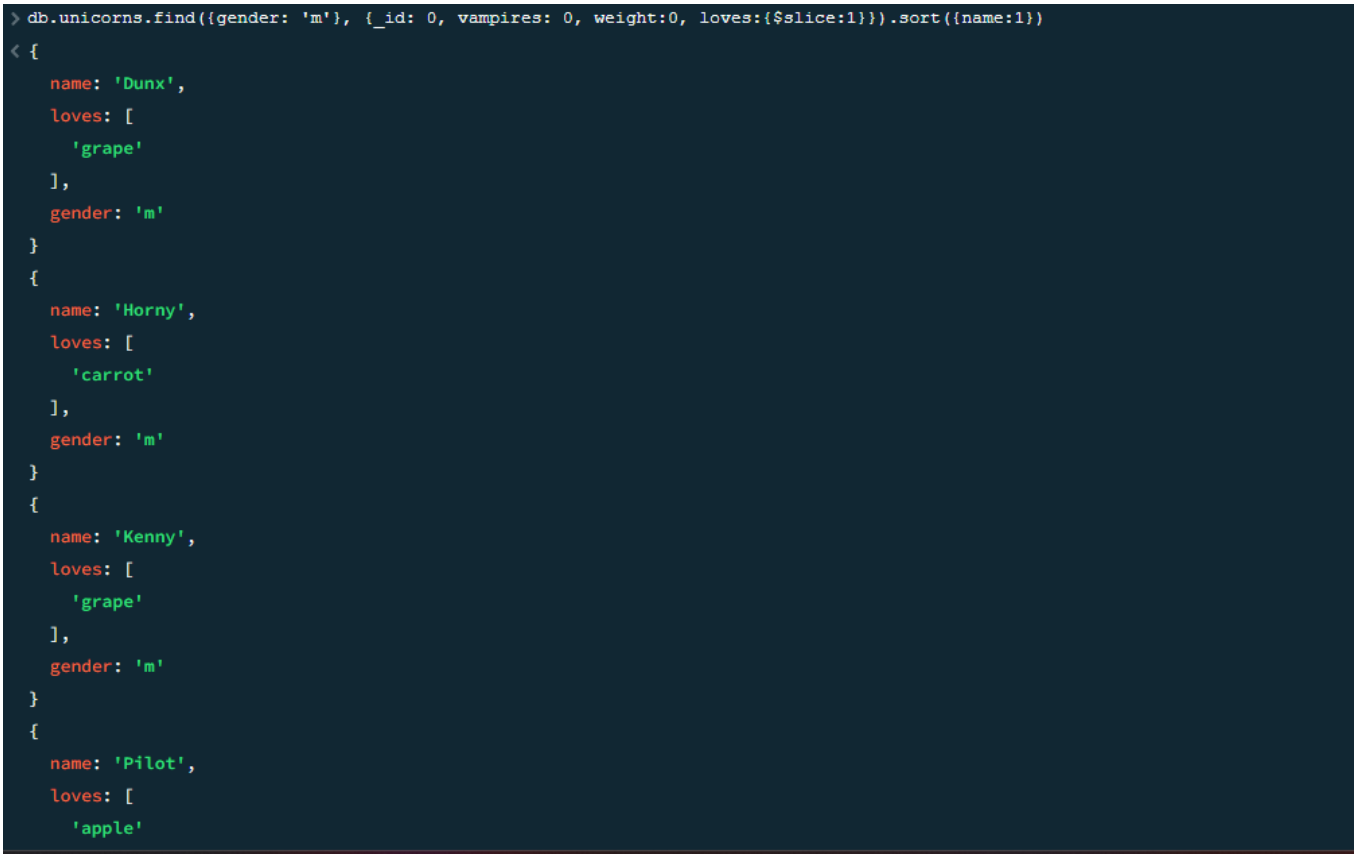

Рисунок 14 – Выполнение запроса 8.1.9

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"}}
```

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

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

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

```
> db.towns.find(
    {"mayor.party": "I"},
    {name: 1, "mayor.name": 1, _id: 0}
  );
< {
    name: 'New York',
    mayor: {
      name: 'Michael Bloomberg'
    }
  }
> db.towns.find(
    {"mayor.party": {$exists: false}},
    {name: 1, "mayor.name": 1, _id: 0}
  );
< {
    name: 'Punxsutawney',
    mayor: {
      name: 'Jim Wehrle'
    }
  }
learn>
```

Рисунок 14 – Выполнение задания 8.2.1

Практическое задание 8.2.2:

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.*

```
func = function() {return this.gender=='m';}
const cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2);
cursor.forEach(function(obj) {print(obj)})
```

```
> func = function() {return this.gender=='m';}
< [Function: func]
> const cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2);
> cursor.forEach(function(obj) {print(obj)})
< {
    _id: ObjectId("64774cefc60f9be6edd45e4e"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
< {
    _id: ObjectId("64774b35f231ec2d910cdafa"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
```

Рисунок 15 – Выполнение задания 8.2.2

Практическое задание 8.2.3:

1. *Вывести количество самок единорогов весом от полутонны до 600 кг.*

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

```
> db.unicorns.find({gender: 'f', weight: {$gt:500, $lt:600}}).count()
< 2
```

Рисунок 16 – Выполнение задания 8.2.3

Практическое задание 8.2.4:

1. *Вывести список предпочтений.*

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

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

Рисунок 17 – Выполнение задания 8.2.4

Практическое задание 8.2.5:

1. *Посчитать количество особей единорогов обоих полов.*

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

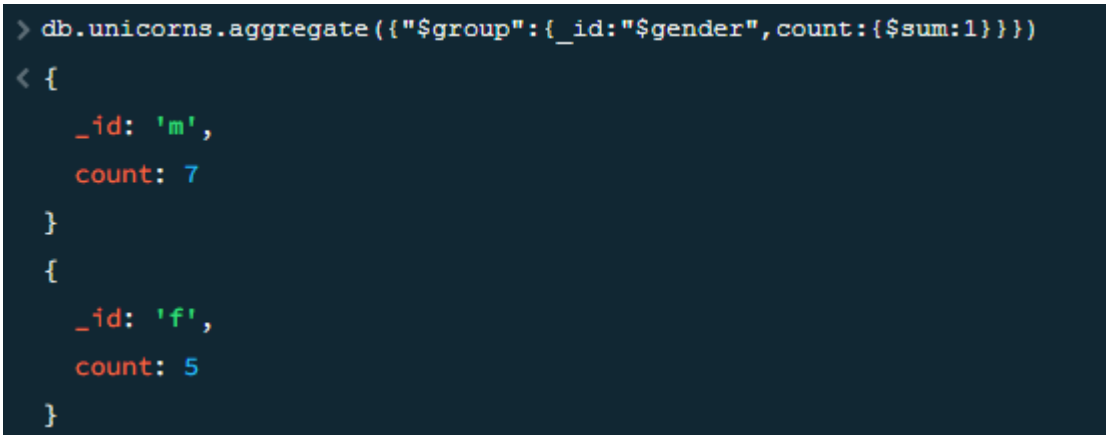

Рисунок 18 – Выполнение задания 8.2.5

Практическое задание 8.2.6:

1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции* `unicorns`.

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

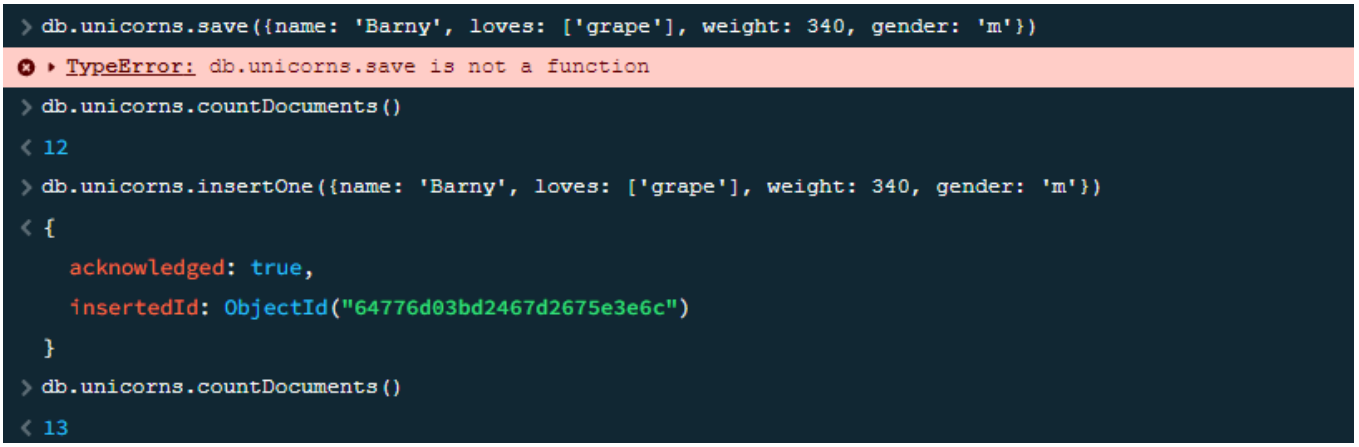

Рисунок 19 – Выполнение задания 8.2.6

Практическое задание 8.2.7:

1. *Для самки единорога* `Ayna` *внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.*
2. *Проверить содержимое коллекции* `unicorns`.

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

```
> db.unicorns.updateOne({name : "Ayna"},  {$set : {weight: 800, vampires: 51}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name : "Ayna"})
< {
    _id: ObjectId("64774b35f231ec2d910cdaff"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
```

Рисунок 20 – Выполнение задания 8.2.7

Практическое задание 8.2.8:

1. *Для самца единорога* `Raleigh` *внести изменения в БД: теперь он любит рэдбул.*
2. *Проверить содержимое коллекции* `unicorns`*.*

```
db.unicorns.updateOne({name: 'Raleigh'}, {$set: {loves: ['redbull']}})
```

```
> db.unicorns.find({name : "Raleigh"})
< {
    _id: ObjectId("64774b35f231ec2d910cdb01"),
    name: 'Raleigh',
    loves: [
      'redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```

Рисунок 21 – Выполнение задания 8.2.8

Практическое задание 8.2.9

1. *Всем самцам единорогов увеличить количество убитых вапмиров на 5.*
2. *Проверить содержимое коллекции* `unicorns`.

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

Рисунок 22 – Выполнение задания 8.2.9

Практическое задание 8.2.10

1. *Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.*
2. *Проверить содержимое коллекции* `towns`.

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

```
> db.towns.updateOne({name: 'Portland'}, {$unset: {"mayor.party": 1}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Рисунок 23 – Выполнение задания 8.2.10

Практическое задание 8.2.11:

1. *Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.*
2. *Проверить содержимое коллекции* `unicorns`.

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

Рисунок 24 – Выполнение задания 8.2.11

Практическое задание 8.2.12:

1. *Изменить информацию о самке единорога* `Aurora`: *теперь она любит еще и сахар, и лимоны.*
2. *Проверить содержимое коллекции* `unicorns`.

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

```
> db.unicorns.updateOne({name: 'Aurora'}, {$addToSet: {loves: {$each: ['sugar', 'lemon']}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name : "Aurora"})
< {
    _id: ObjectId("64774b35f231ec2d910cdafb"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
learn>
```

Рисунок 25 – Выполнение задания 8.2.12

Практическое задание 8.2.13:

1. *Удалите документы с беспартийными мэрами.*
2. *Проверьте содержание коллекции.*
3. *Очистите коллекцию.*
4. *Просмотрите список доступных коллекций.*

```
db.towns.deleteMany({"mayor.party": {$exists:false}})
db.towns.find({}, {name:1, mayor:1})
db.towns.drop()
show collections
```

Рисунок 26 – Выполнение задания 8.2.13

Практическое задание 8.3.1:

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`*:*

```
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', 44), 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'});
db.unicorns.insert {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender:
'm', vampires: 165}
```

```
db.zones.insert({_id: 'z1', name: 'зона 1', description: 'description 1'});
```

```
db.zones.insert({_id: 'z2', name: 'зона 2', description: 'description 2'});
```

```
> db.zones.insert({_id: 'z1', name: 'зона 1', description: 'description 1'});
< {
    acknowledged: true,
    insertedIds: {
      '0': 'z1'
    }
  }
> db.zones.insert({_id: 'z2', name: 'зона 2', description: 'description 2'});
< {
    acknowledged: true,
    insertedIds: {
      '0': 'z2'
    }
  }
> db.zones.find()
< {
    _id: 'z1',
    name: 'зона 1',
    description: 'description 1'
  }
  {
    _id: 'z2',
    name: 'зона 2',
    description: 'description 2'
  }
```

Рисунок 28 - Создать зону

```
> 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'});
  db.unicorns.insert({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("647784a6bd2467d2675e3e7a")
    }
  }
```

Рисунок 29 - Вставить данные

```
> db.unicorns.updateMany(
    { name: { $regex: /^[^A-M]/ } },
    { $set: { zone: { $ref: "zones", $id: "z1" } } }
  );
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 6,
    modifiedCount: 6,
    upsertedCount: 0
  }
> db.unicorns.updateMany(
    { name: { $regex: /^[A-M]/ } },
    { $set: { zone: { $ref: "zones", $id: "z1" } } }
  );
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 6,
    modifiedCount: 6,
    upsertedCount: 0
  }
```

```
> db.unicorns.find()
< {
    _id: ObjectId("64778c0dbd2467d2675e3e7b"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63,
    zone: DBRef("zones", 'z1')
  }
  {
    _id: ObjectId("64778c0dbd2467d2675e3e7c"),
    name: 'Aurora',
    loves: [
      'carrot',
```

Рисунок 30 - Выполнение задания 8.3.1

(Обновите все единороги с учетом их местоположения в зависимости от их имени (единороги с именами, начинающимися на буквы от A до M, будут в зоне 1, а остальные единороги - в зоне 2).

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 })
< [ 'name_1' ]
```

Рисунок 31 – Создание индексов

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

Рисунок 32 – Выполнение задания 8.3.3

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`*)*
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?*

```
> for (var i = 0; i < 100; i++) {
    db.numbers.insert({ value: i });}
< DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("647794c66a8049fbce10c80f")
    }
  }
> db.numbers.find().sort({ _id: -1 }).limit(4);
< {
    _id: ObjectId("647794c66a8049fbce10c80f"),
    value: 99
  }
  {
    _id: ObjectId("647794c66a8049fbce10c80e"),
    value: 98
  }
  {
    _id: ObjectId("647794c66a8049fbce10c80d"),
    value: 97
  }
  {
    _id: ObjectId("647794c66a8049fbce10c80c"),
    value: 96
  }
```

Рисунок 33 – Выполнение задания 8.3.4

Без индекса запрос db.numbers.find().sort({ value: 1 }).limit(4) выполнялся 2 миллисекунды, с индексом – ноль.

3. *Выводы*

В этой лаборатории я освоил практические навыки работы с CRUD-операциями, ссылками и индексами в базе данных MongoDB, узнал больше об операциях Javascript в MongoDB. Я считаю, что это действительно полезные знания, которые помогут программистам в будущей работе.