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

Отчет

по лабораторной работе №6 «РАБОТА С БД В СУБД MONGODB»

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Залетов А.Д.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

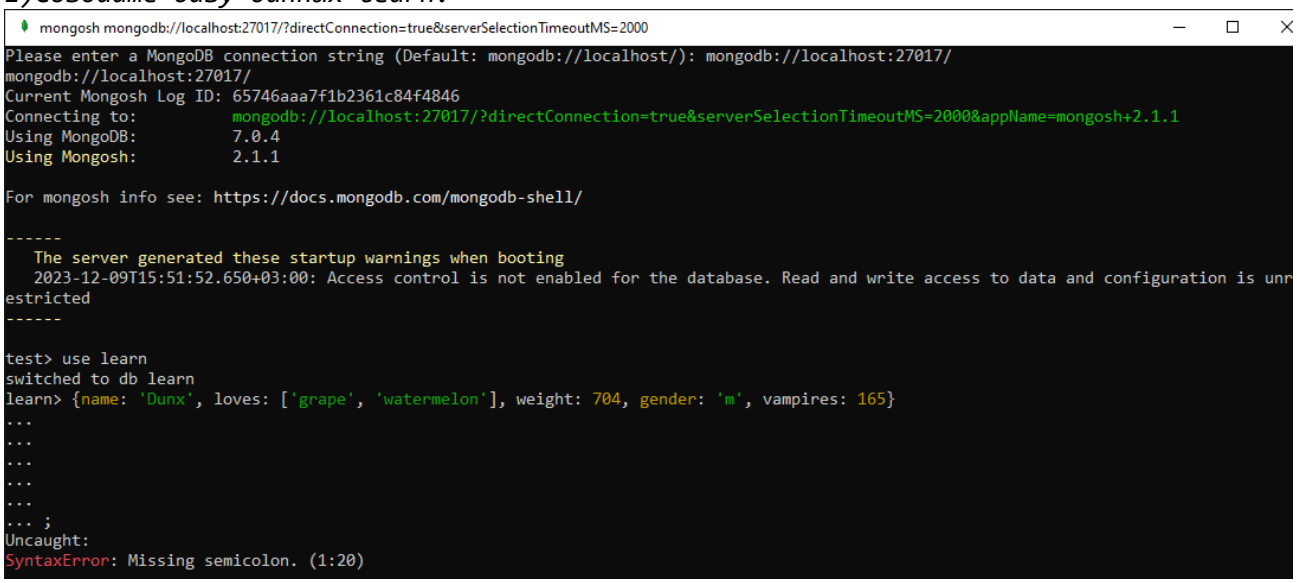
Цель: овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

Оборудование: компьютерный класс.

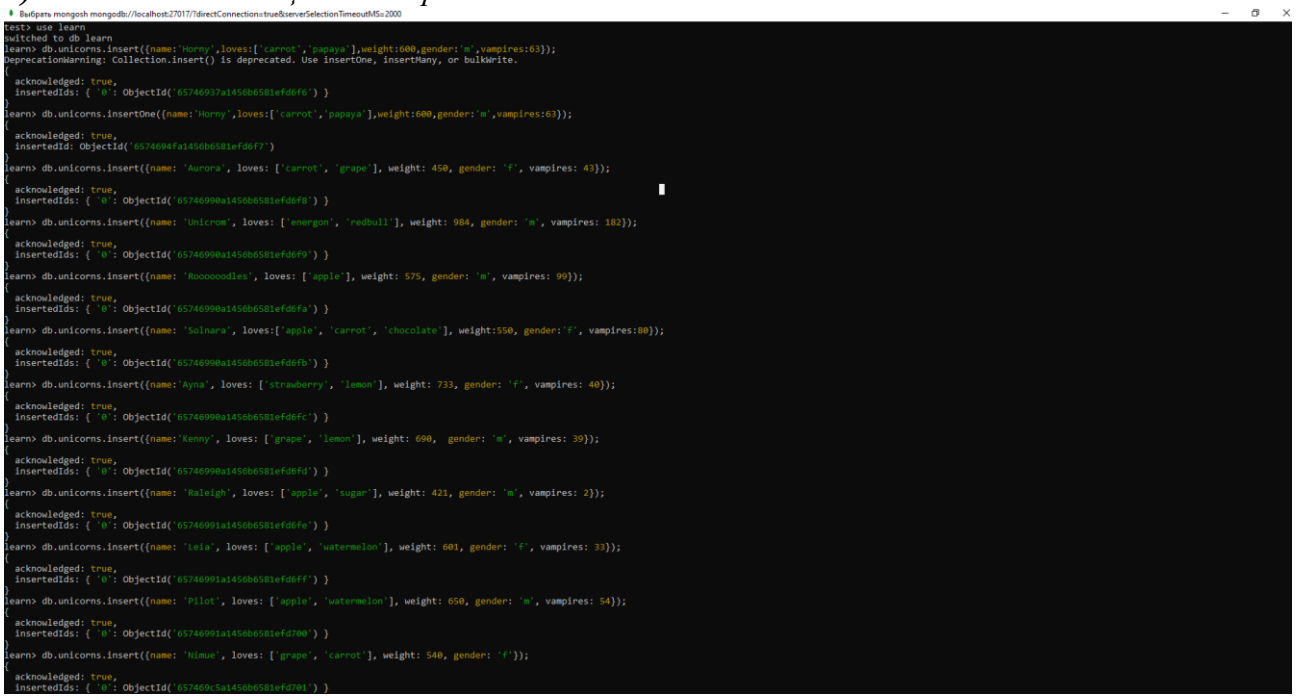
Программное обеспечение: СУБД MongoDB 4+, 6.0.6 (текущая).

Выполнение работы

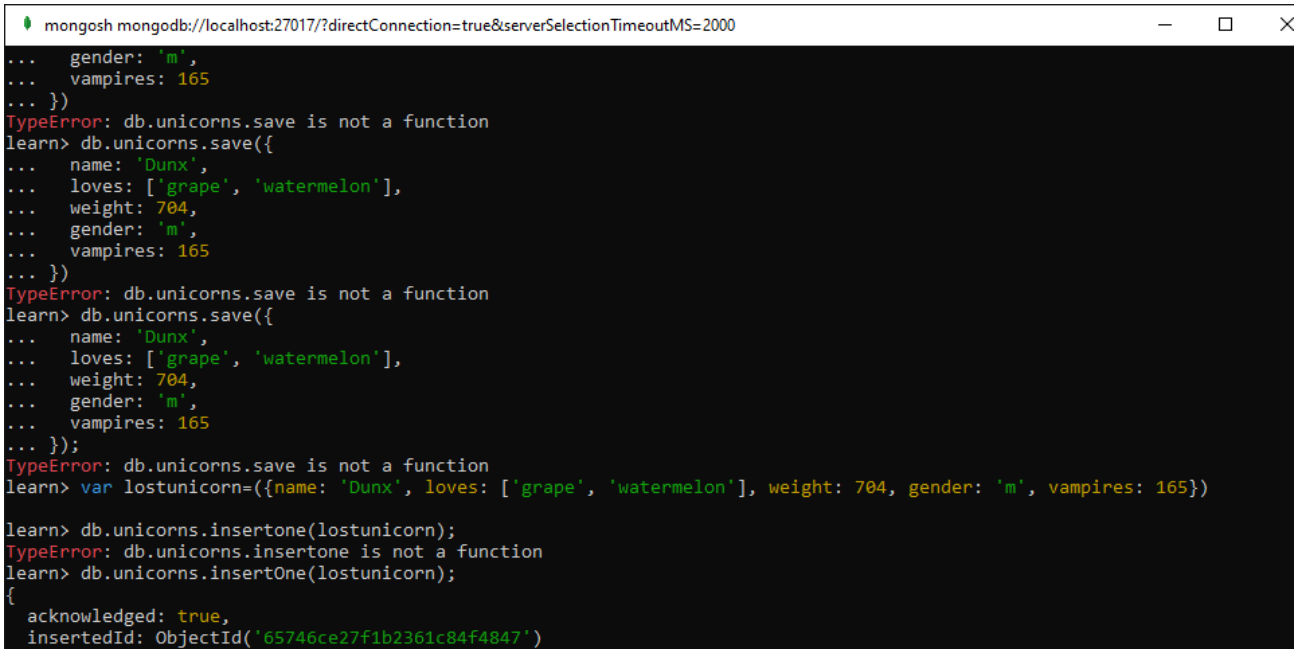
1)Создайте базу данных learn.



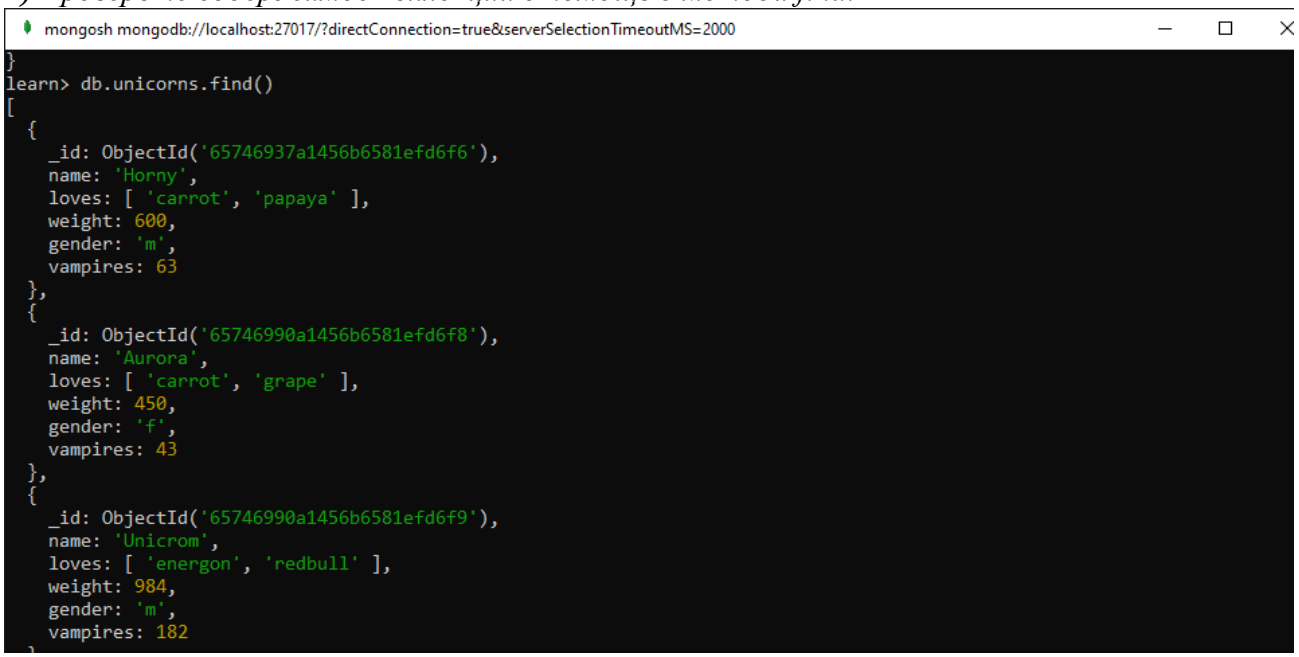
2) Заполните коллекцию единорогов unicorns:



3)Используя второй способ, вставьте в коллекцию единорогов документ:



4) Проверьте содержимое коллекции с помощью метода find.

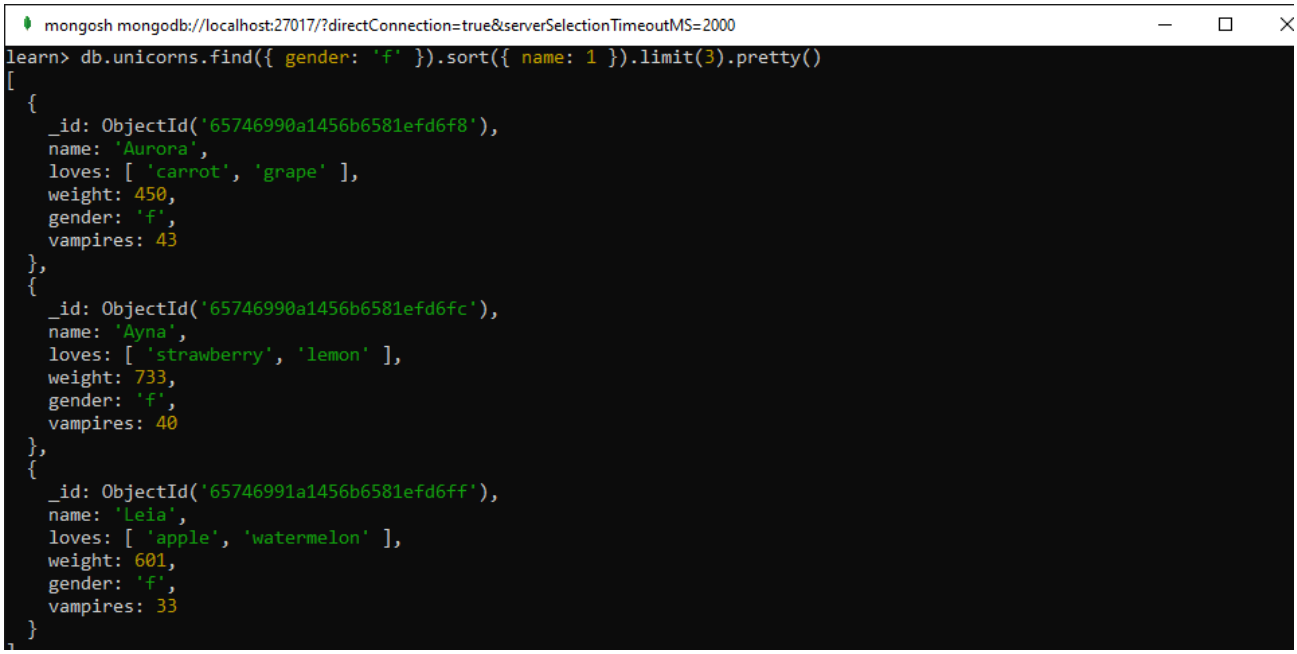


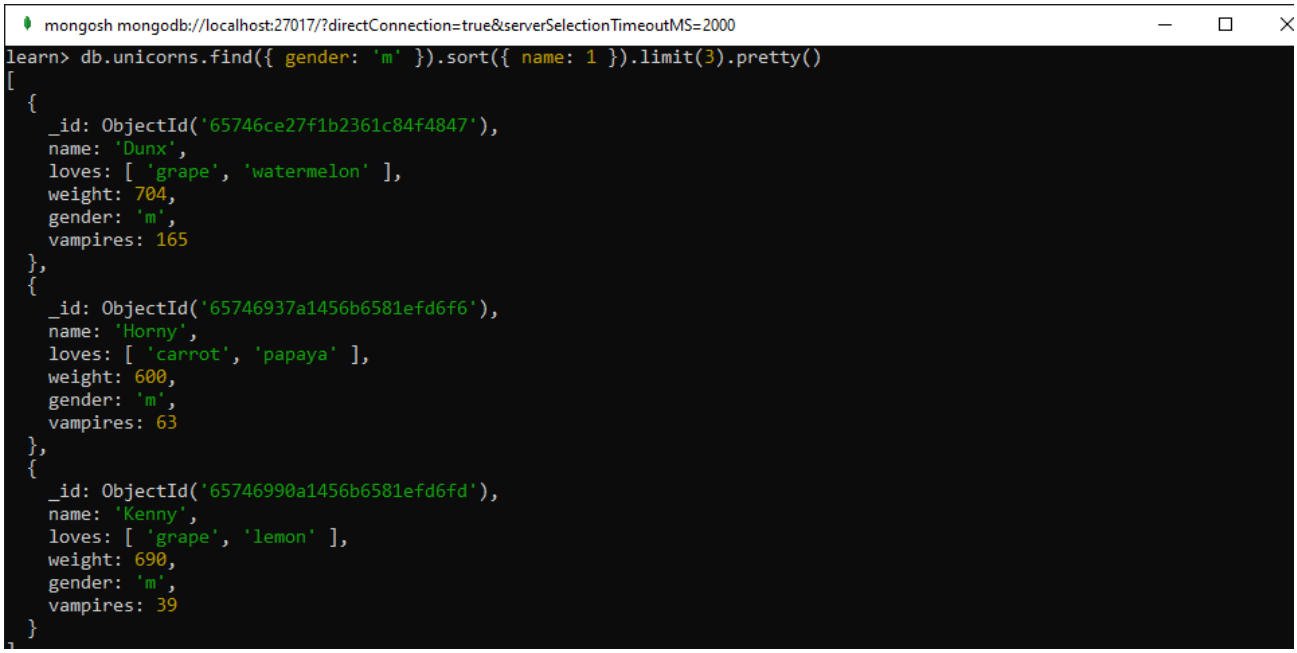

mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
{
_id: ObjectId('65746990a1456b6581efd6fa'),
name: 'Roooooodles',
loves: ['apple'],
weight: 575,
gender: 'm',
vampires: 99
},
{
_id: ObjectId('65746990a1456b6581efd6fb'),
name: 'Solnara',
loves: ['apple', 'carrot', 'chocolate'],
weight: 550,
gender: 'f',
vampires: 80
},
{
_id: ObjectId('65746990a1456b6581efd6fc'),
name: 'Ayna',
loves: ['strawberry', 'lemon'],
weight: 733,
gender: 'f',
vampires: 40
},
{
_id: ObjectId('65746990a1456b6581efd6fd'),


mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
loves: ['grape', 'lemon'],
weight: 690,
gender: 'm',
vampires: 39
},
{
_id: ObjectId('65746991a1456b6581efd6fe'),
name: 'Raleigh',
loves: ['apple', 'sugar'],
weight: 421,
gender: 'm',
vampires: 2
},
{
_id: ObjectId('65746991a1456b6581efd6ff'),
name: 'Leia',
loves: ['apple', 'watermelon'],
weight: 601,
gender: 'f',
vampires: 33
},
{
_id: ObjectId('65746991a1456b6581efd700'),
name: 'Pilot',
loves: ['apple', 'watermelon'],
weight: 650,

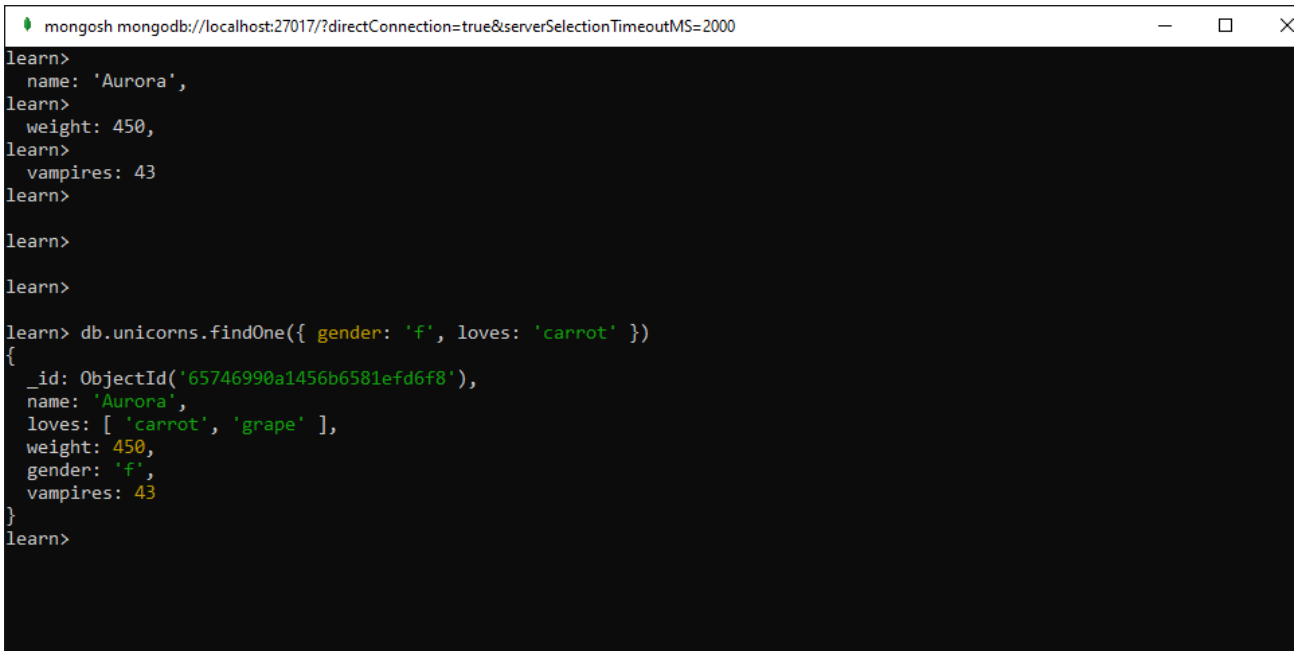
Практическое задание 2.2.1:

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



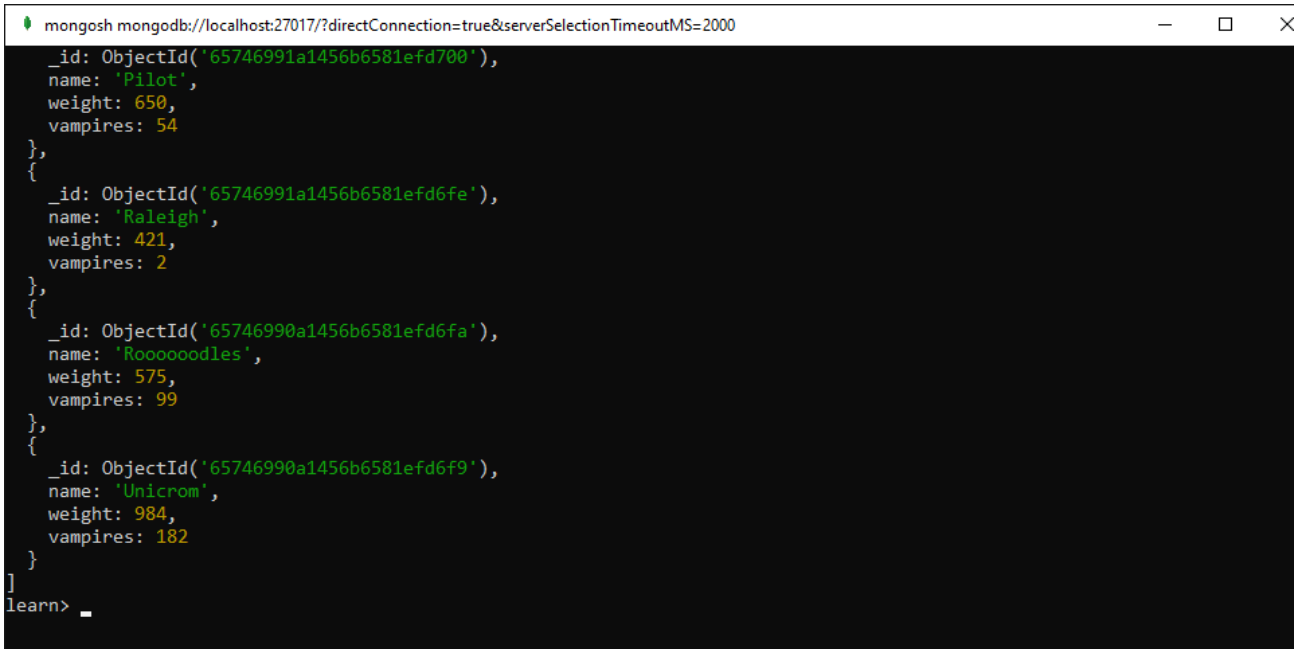


2)Найдите всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций findOne и limit.



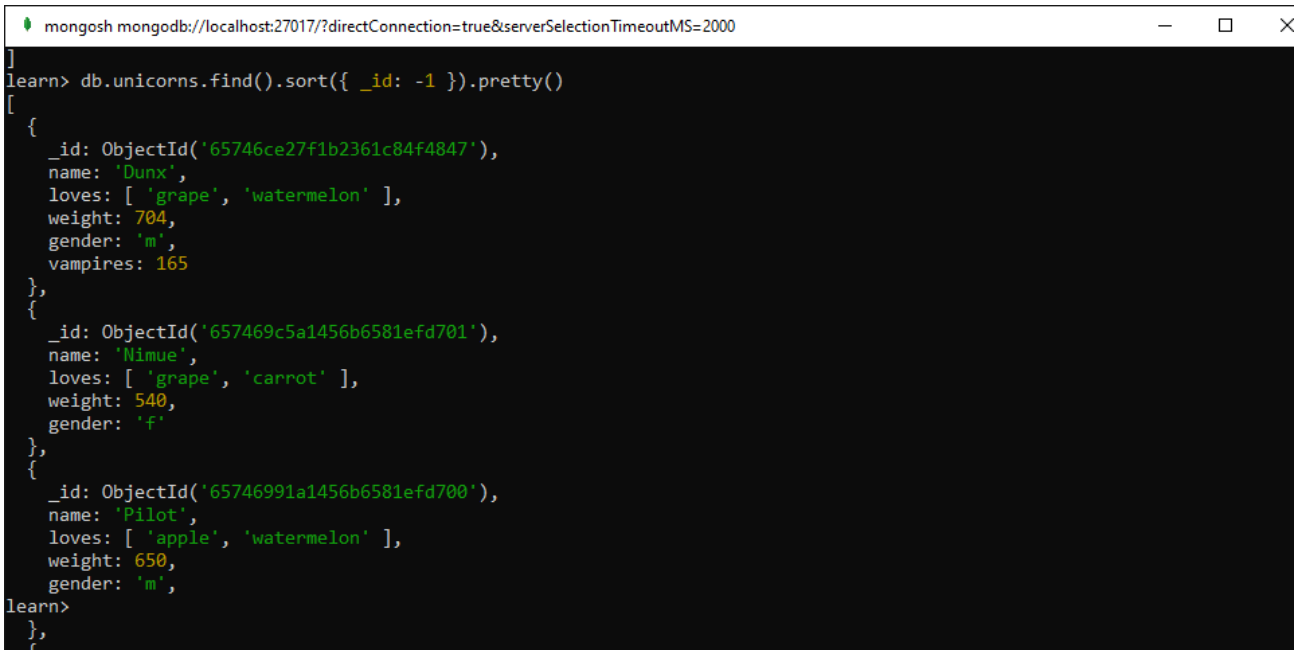
Практическое задание 2.2.2:

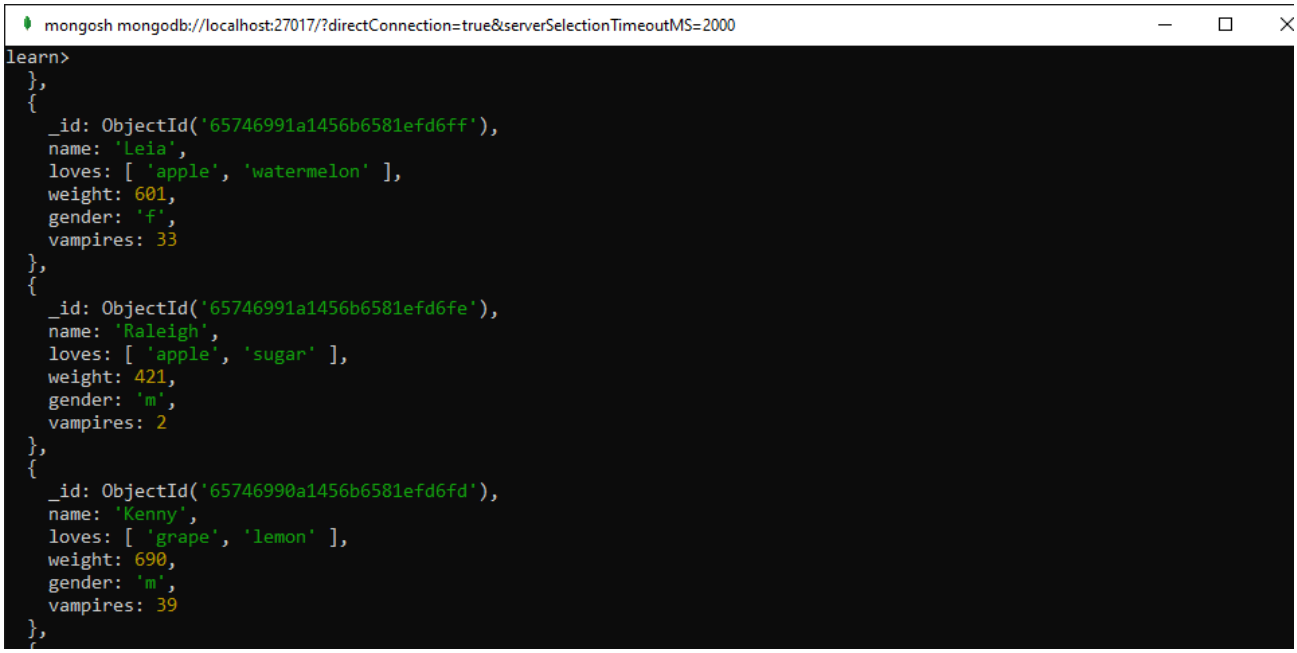
Модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпотениях и поле.

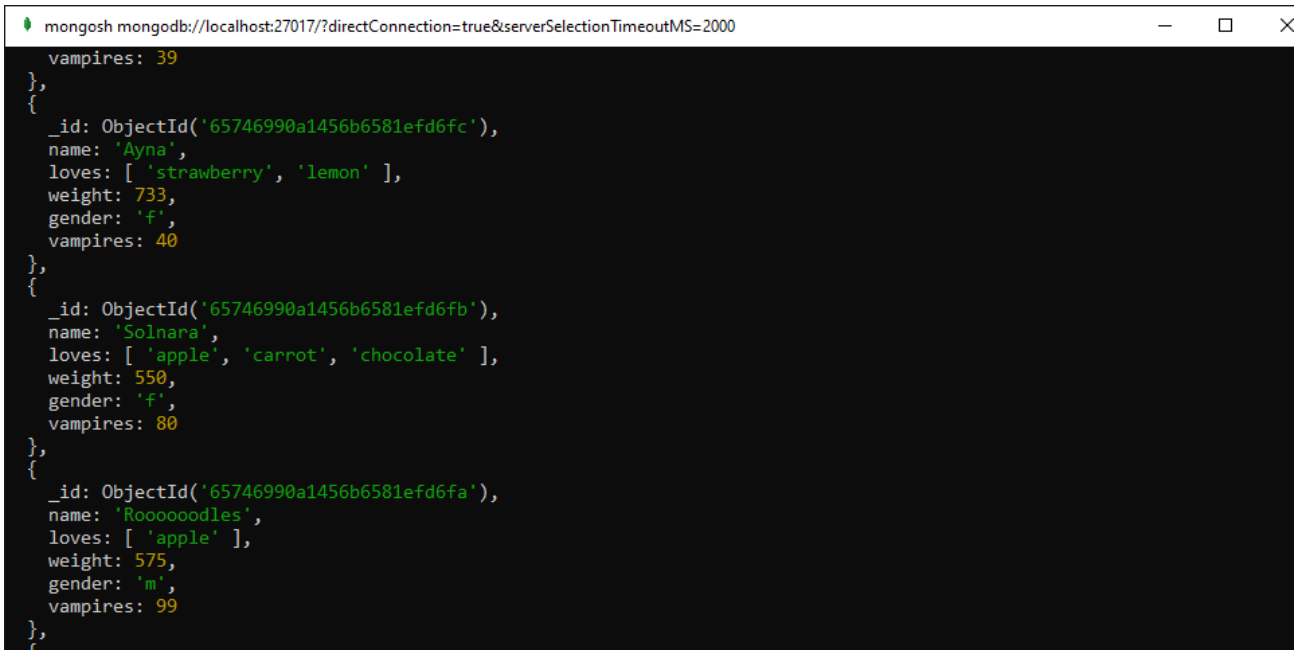


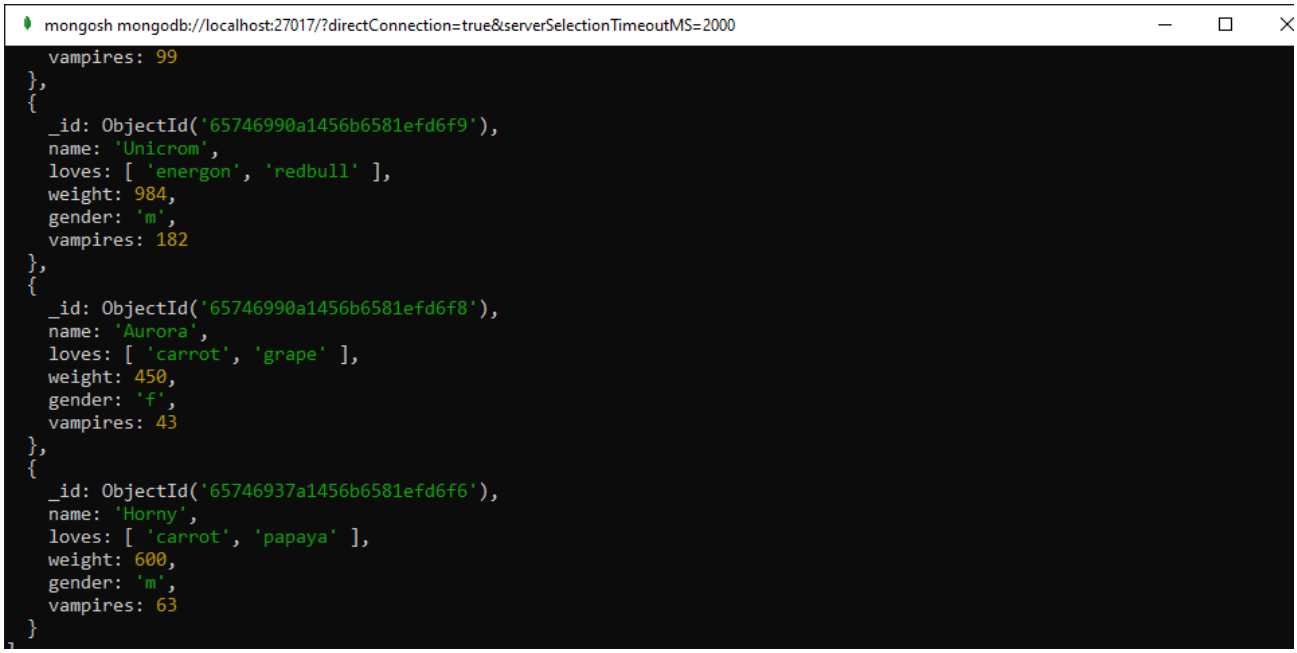
Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления




mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn>
{
_id: ObjectId('65746991a1456b6581efd6ff'),
name: 'Leia',
loves: ['apple', 'watermelon'],
weight: 601,
gender: 'f',
vampires: 33
},
{
_id: ObjectId('65746991a1456b6581efd6fe'),
name: 'Raleigh',
loves: ['apple', 'sugar'],
weight: 421,
gender: 'm',
vampires: 2
},
{
_id: ObjectId('65746990a1456b6581efd6fd'),
name: 'Kenny',
loves: ['grape', 'lemon'],
weight: 690,
gender: 'm',
vampires: 39
},


mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
vampires: 39
},
{
_id: ObjectId('65746990a1456b6581efd6fc'),
name: 'Ayna',
loves: ['strawberry', 'lemon'],
weight: 733,
gender: 'f',
vampires: 40
},
{
_id: ObjectId('65746990a1456b6581efd6fb'),
name: 'Solnara',
loves: ['apple', 'carrot', 'chocolate'],
weight: 550,
gender: 'f',
vampires: 80
},
{
_id: ObjectId('65746990a1456b6581efd6fa'),
name: 'Roooooodles',
loves: ['apple'],
weight: 575,
gender: 'm',
vampires: 99
},



Практическое задание 2.3.1:

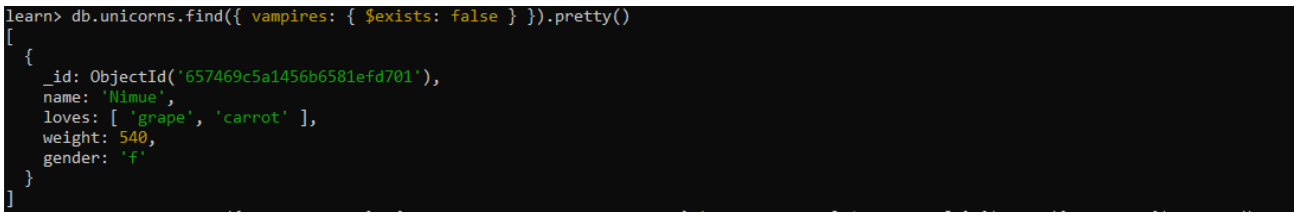
Вывести список самок единорогов весом от полутонны до 700 кг, исключив вывод идентификатора.

Практическое задание 2.3.2:

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

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.



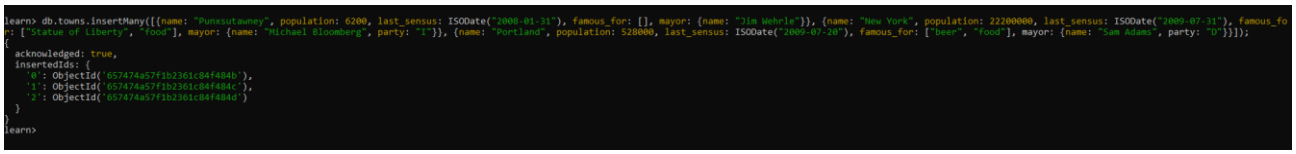
Практическое задание 2.3.4:

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

Практическое задание 3.1.1:

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

```
{name: "Punxsutawney ",
populatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: [""],
mayor: {
   name: "Jim Wehrle"
   }}

{name: "New York",
populatiuon: 22200000,
last_sensus: ISODate("2009-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
   name: "Michael Bloomberg",
   party: "I"}}

{name: "Portland",
populatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
   party: "D"}}
```



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.*

Практическое задание 3.1.2:

1)Сформировать функцию для вывода списка самцов единорогов.

```
mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000

learn>

learn>

learn>

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

learn>

learn> function listMaleUnicorns() {
...   return db.unicorns.find({ gender: 'm' }, { _id: 0, name: 1, weight: 1, loves: 1 }).pretty();
... }
[Function: listMaleUnicorns]
learn> listMaleUnicorns();
[
  { name: 'Horny', loves: [ 'carrot', 'papaya' ], weight: 600 },
  { name: 'Unicrom', loves: [ 'energon', 'redbull' ], weight: 984 },
  { name: 'Roooooodles', loves: [ 'apple' ], weight: 575 },
  { name: 'Kenny', loves: [ 'grape', 'lemon' ], weight: 690 },
  { name: 'Raleigh', loves: [ 'apple', 'sugar' ], weight: 421 },
  { name: 'Pilot', loves: [ 'apple', 'watermelon' ], weight: 650 },
  { name: 'Dunx', loves: [ 'grape', 'watermelon' ], weight: 704 }
]
learn>
```

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

```
learn> var cursor = db.unicorns.find({ gender: 'm' }, { _id: 0, name: 1, weight: 1, loves: 1 }).sort({ name: 1 }).limit(2);

...   printjson(cursor.next());
... }
{ {
  name: 'Dunx',Id('657469c5a1456b6581efd701'),
  loves: ['Nimue',
    'grape', 'grape', 'carrot' ],
    'watermelon'
  ],gender: 'f'
  weight: 704
}
{earn> db.unicorns.find({ gender: 'f', loves: 'carrot' }).limit(1).pretty()
  name: 'Horny',
  loves: [
    'carrot',ctId('65746990a1456b6581efd6f8'),
    'papaya'urora',
  ],loves: [ 'carrot', 'grape' ],
  weight: 60050,
}   gender: 'f',
```

3)Вывести результат, используя forEach.

```
mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000

learn>

learn>

learn>

learn> while (cursor.hasNext()) printjson(cursor.next());

learn>

learn> db.unicorns.find({ gender: 'm' }, { _id: 0, name: 1, weight: 1, loves: 1 }).sort({ name: 1 }).limit(2).forEach(doc => printjson(doc));
{
  name: 'Dunx',
  loves: [
    'grape',
    'watermelon'
  ],
  weight: 704
}
{
  name: 'Horny',
  loves: [
    'carrot',
    'papaya'
  ],
  weight: 600
}
```

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
}

learn> var femaleUnicornCount = db.unicorns.distinct("name", { gender: "f", weight: { $gte: 500, $lte: 600 } }).length;

learn> print("Количество самок: " + femaleUnicornCount);
Количество самок: 2
```

Практическое задание 3.2.2:

Вывести список предпочтений.

```
learn> db.unicorns.aggregate([{ $unwind: "$loves" }, { $group: { _id: "$loves", count: { $sum: 1 } } }, { $project: { _id: 0, preference: "$_id", count: 1 } }]).pretty();
[
  { count: 5, preference: 'apple' },
  { count: 1, preference: 'papaya' },
  { count: 4, preference: 'grape' },
  { count: 1, preference: 'energon' },
  { count: 1, preference: 'sugar' },
  { count: 1, preference: 'redbull' },
  { count: 3, preference: 'watermelon' },
  { count: 2, preference: 'lemon' },
  { count: 1, preference: 'strawberry' },
  { count: 4, preference: 'carrot' },
  { count: 1, preference: 'chocolate' }
```

Практическое задание 3.2.3:

Посчитать количество особей единорогов обоих полов.

```
learn> db.unicorns.aggregate([{ $group: { _id: "$gender", count: { $sum: 1 } } }, { $project: { _id: 0, gender: "$_id", count: 1 } }]).pretty();
[ { count: 7, gender: 'm' }, { count: 5, gender: 'f' } ]
learn>
```

Практическое задание 3.3.2:

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

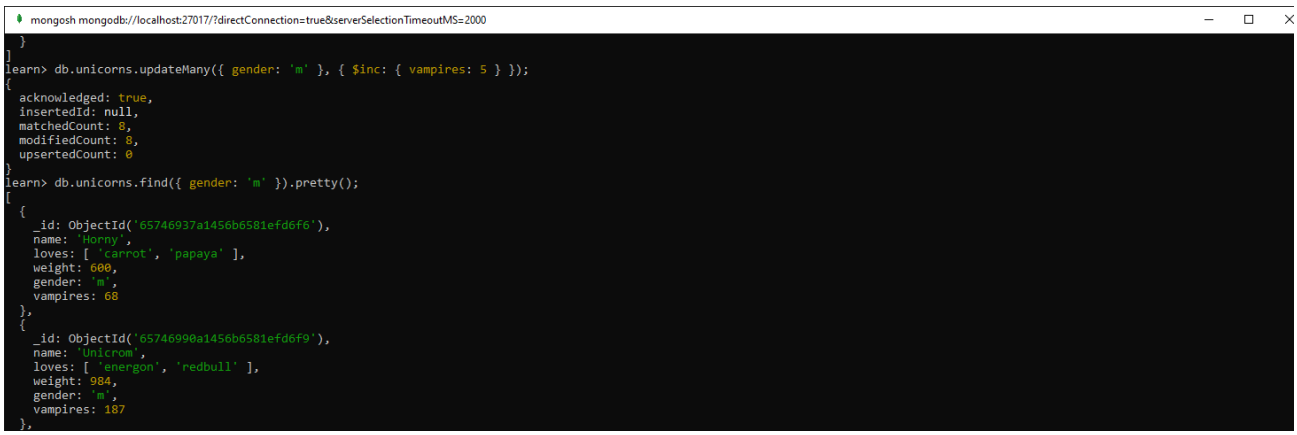
```
mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn>

learn>

learn>

learn>

learn>

learn> db.unicorns.updateOne({ name: 'Ayna', gender: 'f' }, { $set: { weight: 800, vampires: 51 } });
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({ name: 'Ayna', gender: 'f' }).pretty();
[
  {
    _id: ObjectId('65746990a1456b6581efd6fc'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
```

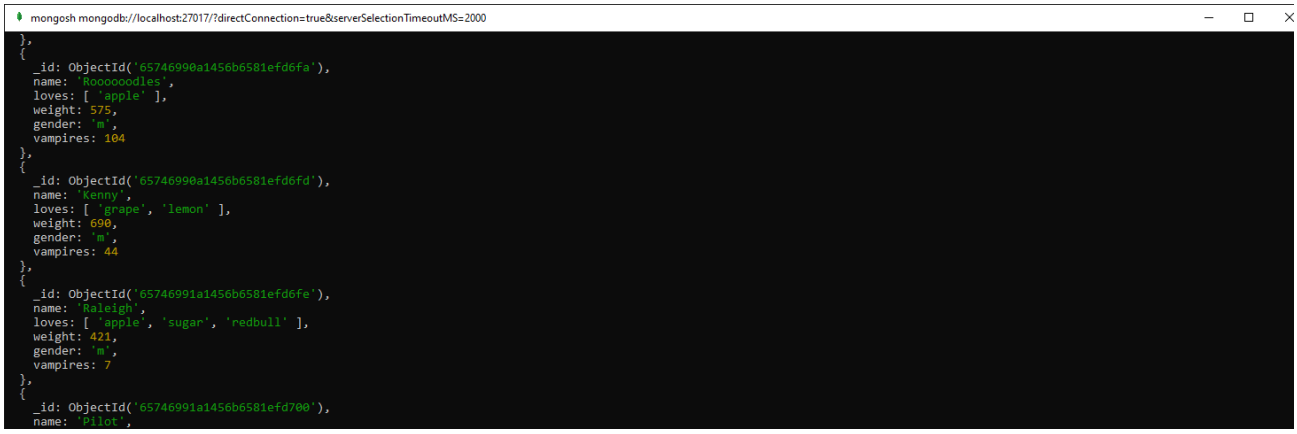
Практическое задание 3.3.3:

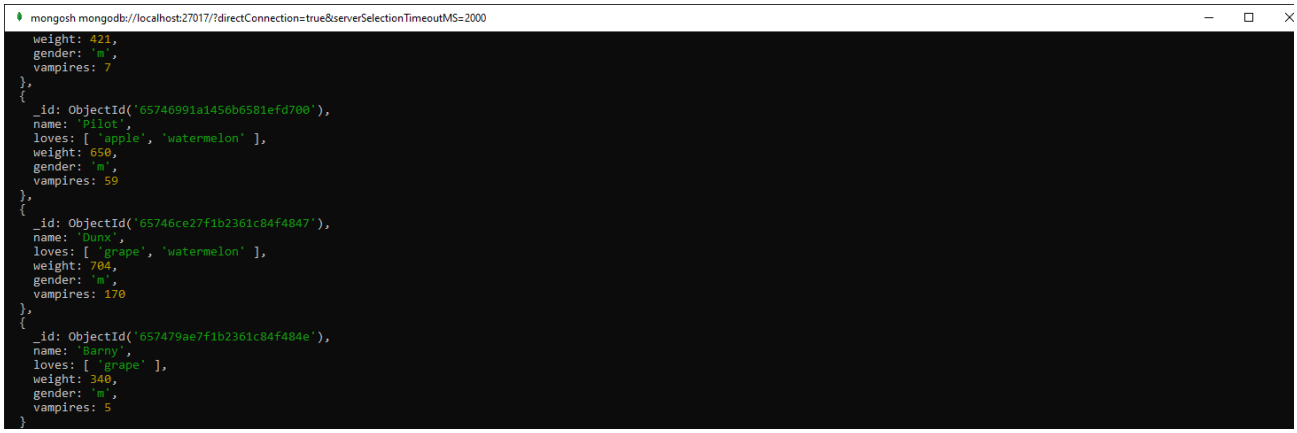
Для самца единорога `Raleigh` *внести изменения в БД: теперь он любит и рэдбул.*

Практическое задание 3.3.4:

Всем самцам единорогов увеличить количество убитых вапмиров на 5.







Практическое задание 3.3.5:

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

Проверить содержимое коллекции `towns`.

Практическое задание 3.3.6:

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

Проверить содержимое коллекции `unicorns`.

Практическое задание 3.3.7:

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

Проверить содержимое коллекции `unicorns`.

```
learn> db.unicorns.updateOne({ name: 'Aurora', gender: 'f' }, { $addToSet: { loves: { $each: ['sugar', 'lemon'] } } });
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({ name: 'Aurora', gender: 'f' }).pretty();
[
  {
    _id: ObjectId('65746990a1456b6581efd6f8'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Практическое задание 3.4.1:

Создайте коллекцию towns, включающую следующие документы:

```
{name: "Punxsutawney ",
popujatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: ["phil the groundhog"],
mayor: {
   name: "Jim Wehrle"
   }}

{name: "New York",
popujatiuon: 22200000,
last_sensus: ISODate("2009-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
   name: "Michael Bloomberg",
   party: "I"}}

{name: "Portland",
popujatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
   party: "D"}}
```

```
learn> db.towns.insertMany([{name: "Punxsutawney", population: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil the groundhog"], mayor: {name: "Jim Wehrle"}},
... {name: "New York", population: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["Statue of Liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}},
... {name: "Portland", population: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: {name: "Sam Adams", party: "D"}}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65747bd17f1b2361c84f484f'),
    '1': ObjectId('65747bd17f1b2361c84f4850'),
    '2': ObjectId('65747bd17f1b2361c84f4851')
  }
```

2. Удалите документы с беспартийными мэрами.

```
learn> db.towns.insertMany([{name: "Punxsutawney", population: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil the groundhog"], mayor: {name: "Jim Wehrle"}},
... {name: "New York", population: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["Statue of Liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}},
... {name: "Portland", population: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: {name: "Sam Adams", party: "D"}}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65747bd17f1b2361c84f484f'),
    '1': ObjectId('65747bd17f1b2361c84f4850'),
    '2': ObjectId('65747bd17f1b2361c84f4851')
  }
}
learn> db.towns.deleteMany({"mayor.party": {$exists: false}});
```

3. Проверьте содержание коллекции.

```
mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65747bd17f1b2361c84f484f'),
    '1': ObjectId('65747bd17f1b2361c84f4850'),
    '2': ObjectId('65747bd17f1b2361c84f4851')
  }
}
learn> db.towns.deleteMany({"mayor.party": {$exists: false}});
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find().pretty();
[
  {
    _id: ObjectId('65747bd17f1b2361c84f4850'),
    name: 'New York',
    population: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'Statue of Liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('65747bd17f1b2361c84f4851'),
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

4. Очистите коллекцию.

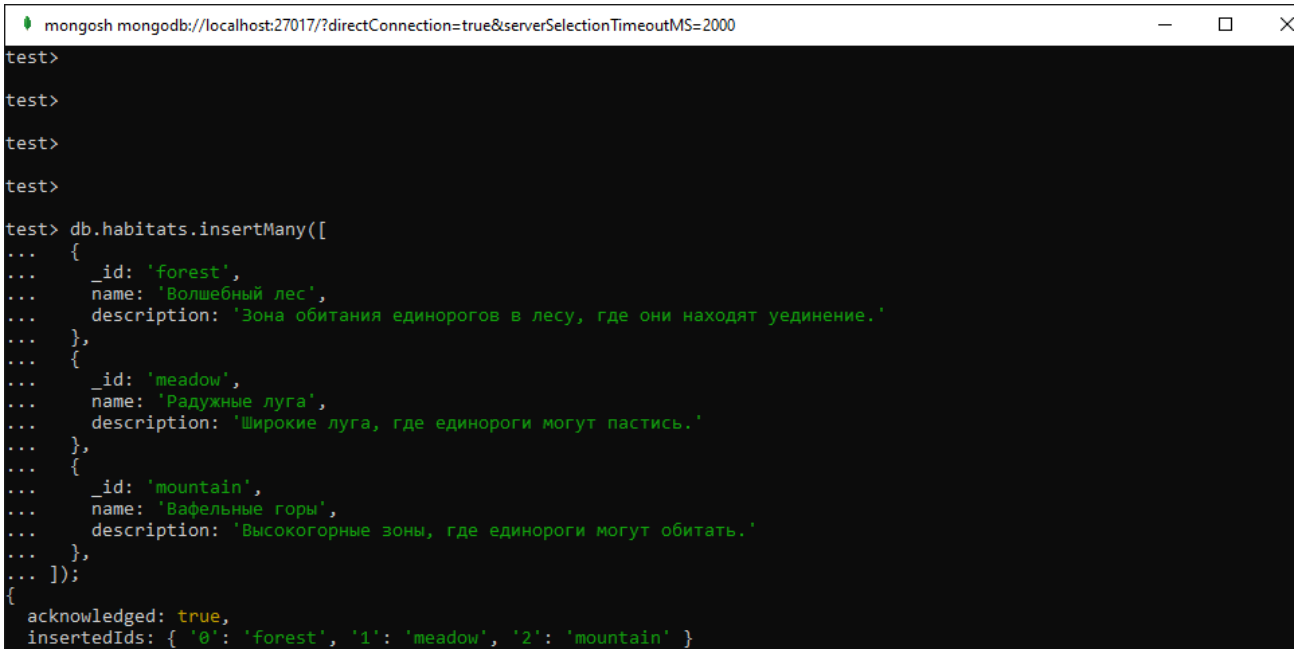
```
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.deleteMany({});
```

5. Просмотрите список доступных коллекций.

```
mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
    '2': ObjectId('65747bd17f1b2361c84f4851')
  }
}
learn> db.towns.deleteMany({"mayor.party": {$exists: false}});
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find().pretty();
[
  {
    _id: ObjectId('65747bd17f1b2361c84f4850'),
    name: 'New York',
    population: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'Statue of Liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('65747bd17f1b2361c84f4851'),
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.deleteMany({});
{ acknowledged: true, deletedCount: 2 }
learn> show collections;
towns
```

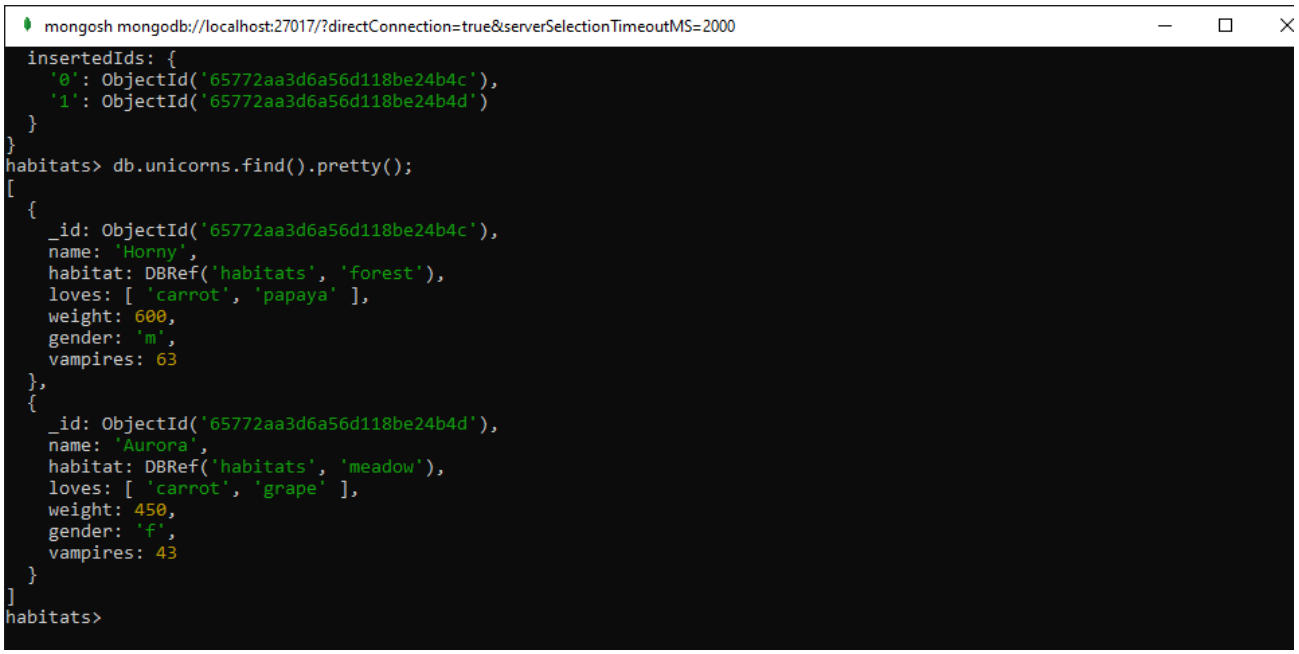
Практическое задание 4.1.1:

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



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

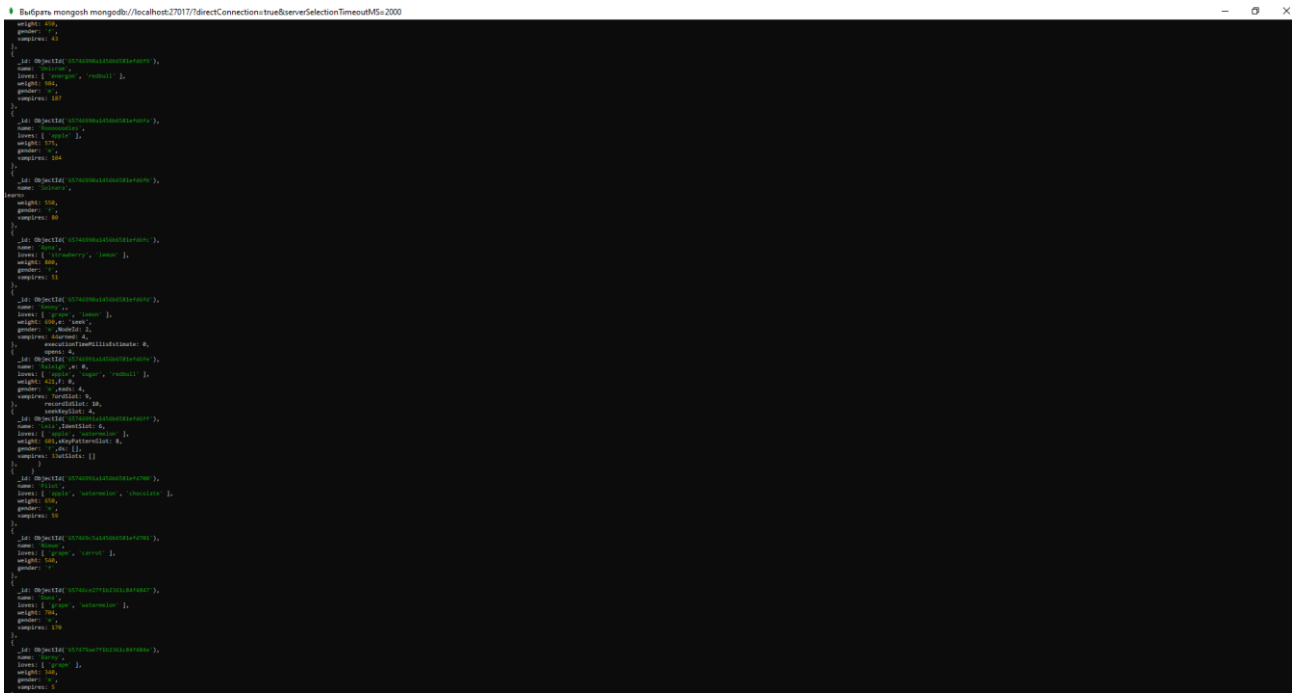
Проверьте содержание коллекции единорогов.

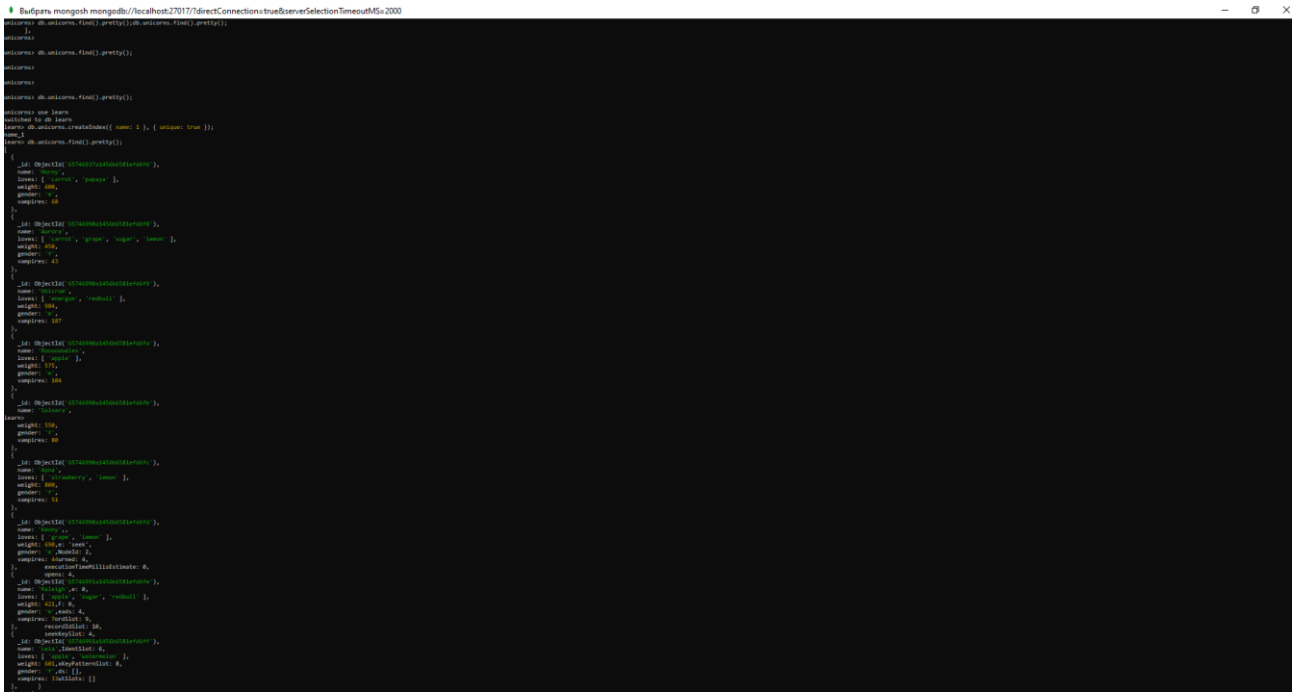


Практическое задание 4.2.1:

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

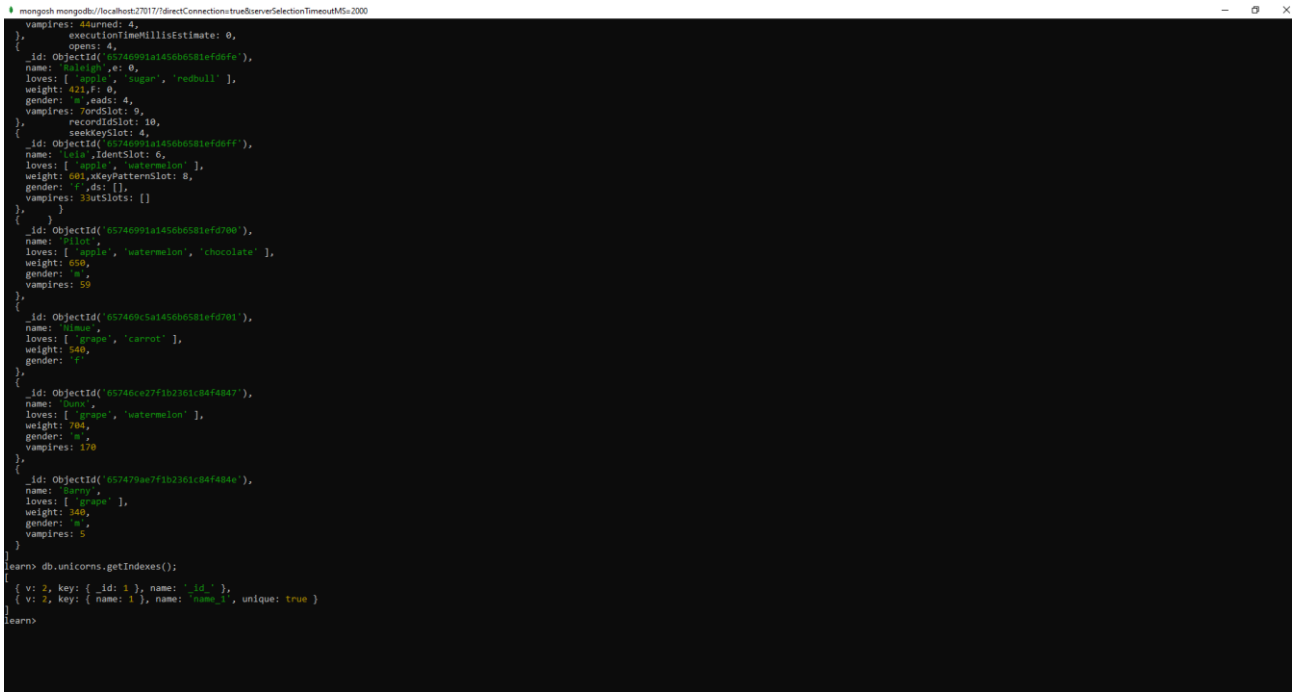
Содержание коллекции единорогов `unicorns`*:*


Выбрать mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000

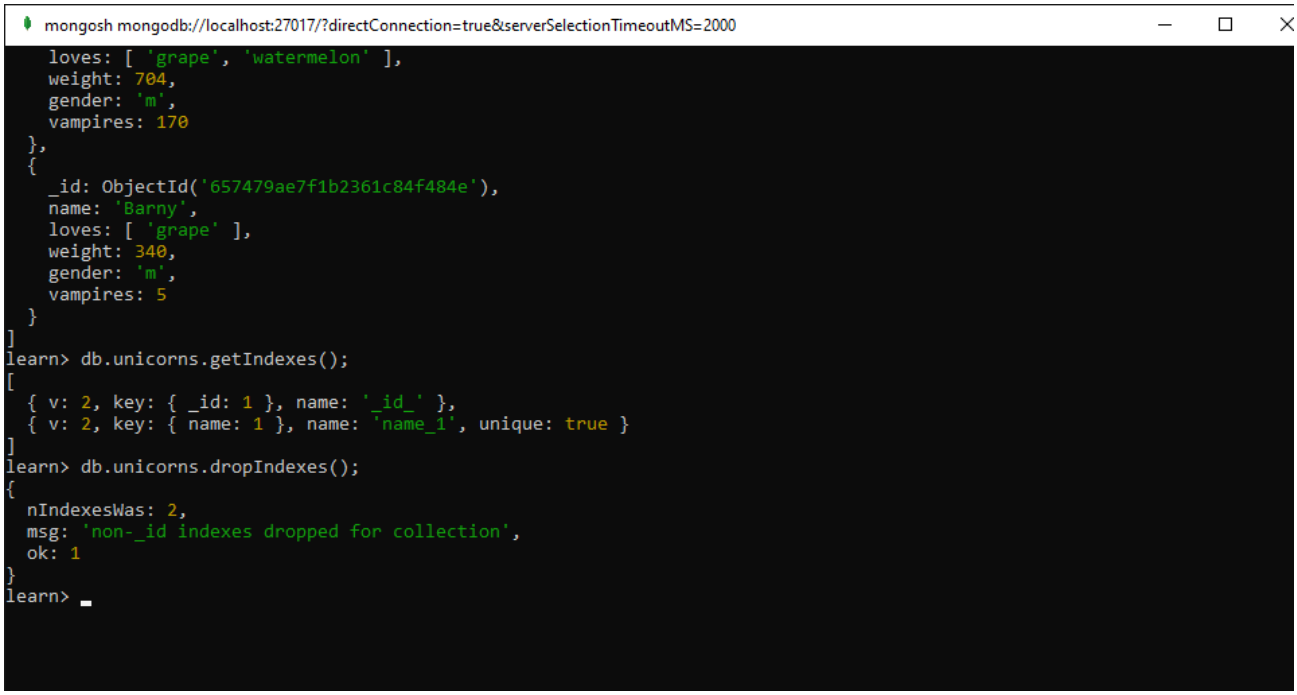

Выбрать mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000

Практическое задание 4.3.1:

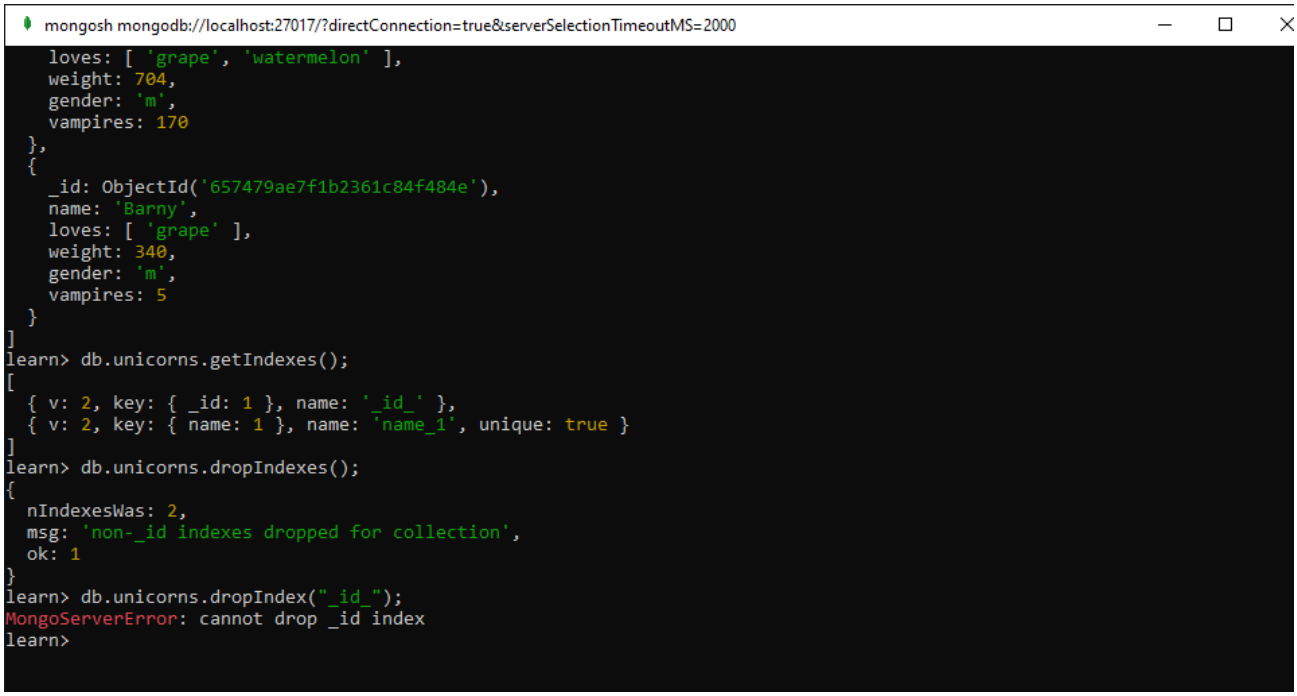
Получите информацию о всех индексах коллекции `unicorns`.



Удалите все индексы, кроме индекса для идентификатора.



Попытайтесь удалить индекс для идентификатора.



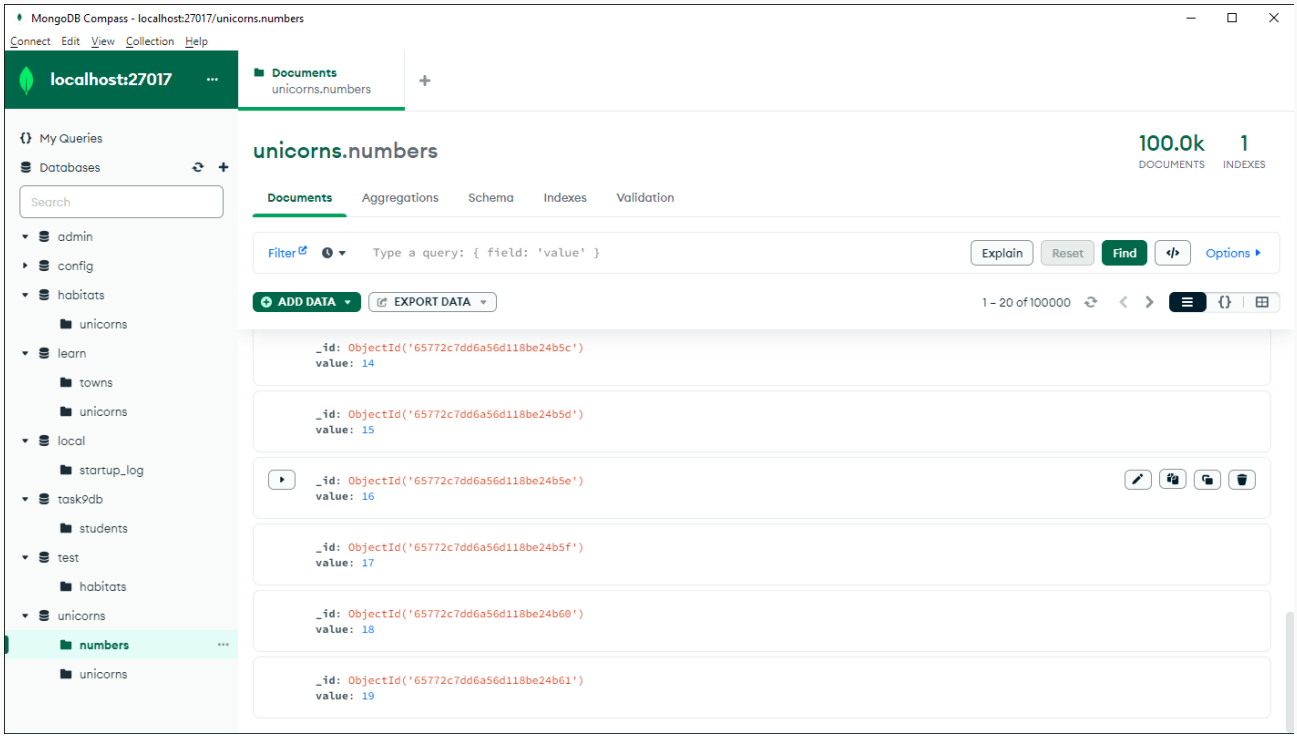
Удалять индекс для идентификатора нельзя! Ибо это может сломать нашу БД, даже система не даёт нам этого сделать

Практическое задание 4.4.1:

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

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

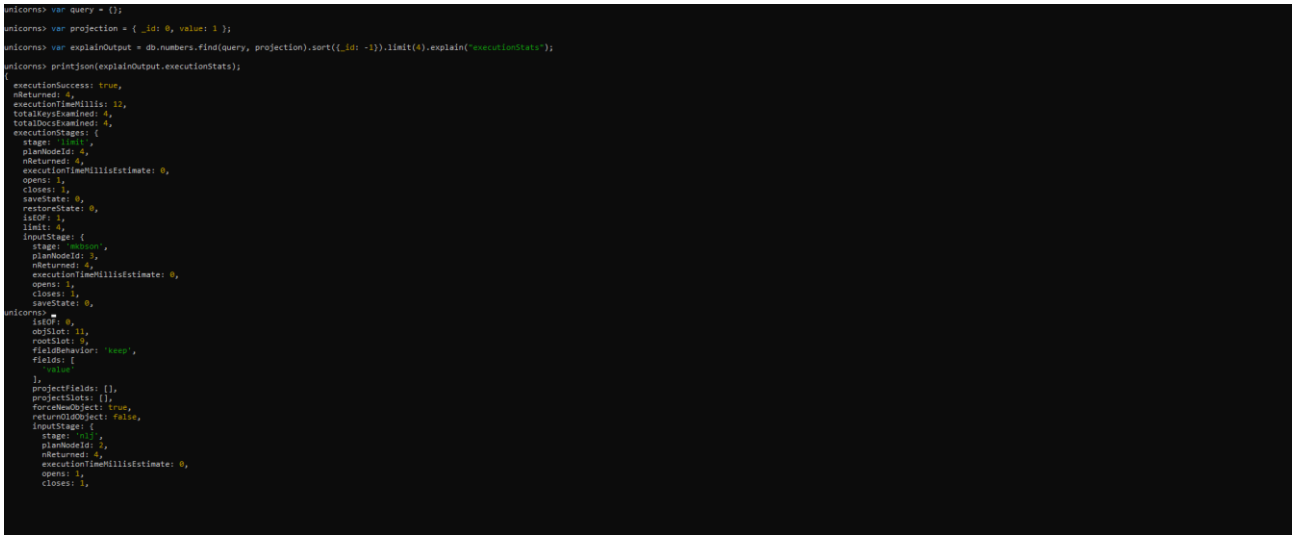
```
unicorns> for (let i = 0; i < 100000; i++) {
...   db.numbers.insert({ value: i });
... };
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.


;


{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65772ce0d6a56d118be3d1ed') }
}
```



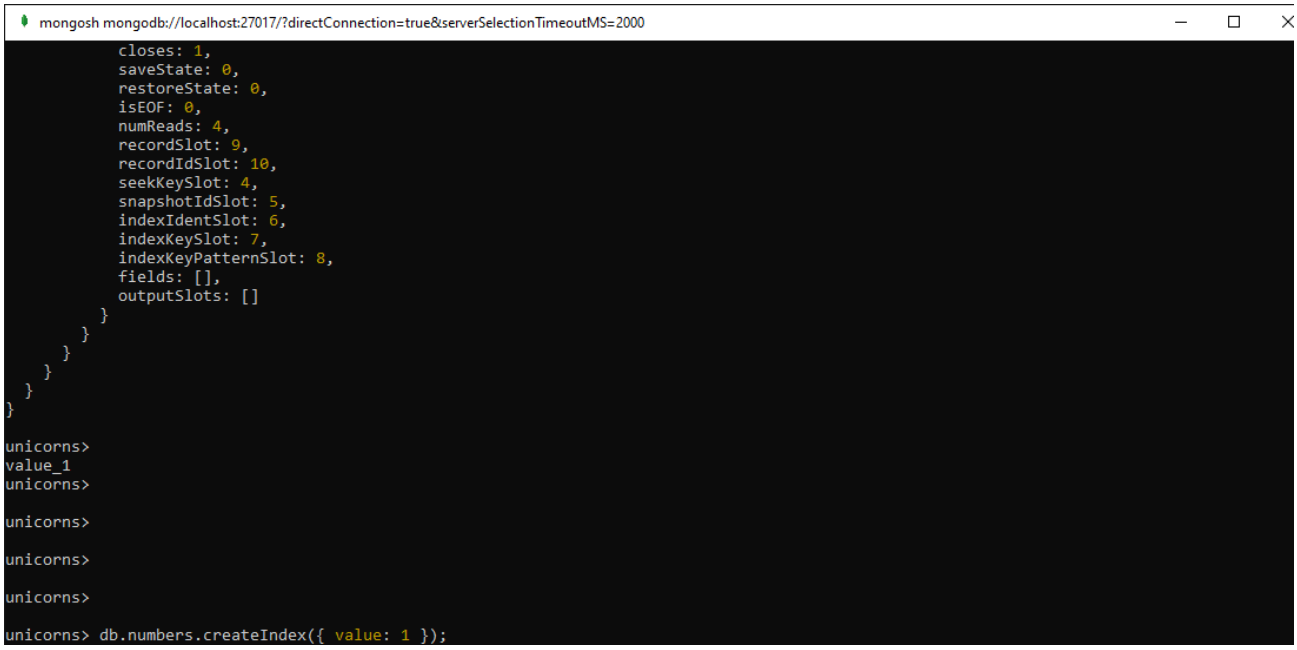
Выберите последних четыре документа.

```
mongosh mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000

unicorns> ;

unicorns>

unicorns>

unicorns> var lastFourDocuments = db.numbers.find().sort({_id: -1}).limit(4).toArray();

unicorns> printjson(lastFourDocuments);
[
  {
    _id: ObjectId('65772ce0d6a56d118be3d1ed'),
    value: 99999
  },
  {
    _id: ObjectId('65772ce0d6a56d118be3d1ec'),
    value: 99998
  },
  {
    _id: ObjectId('65772ce0d6a56d118be3d1eb'),
    value: 99997
  },
  {
    _id: ObjectId('65772ce0d6a56d118be3d1ea'),
    value: 99996
  }
]
```

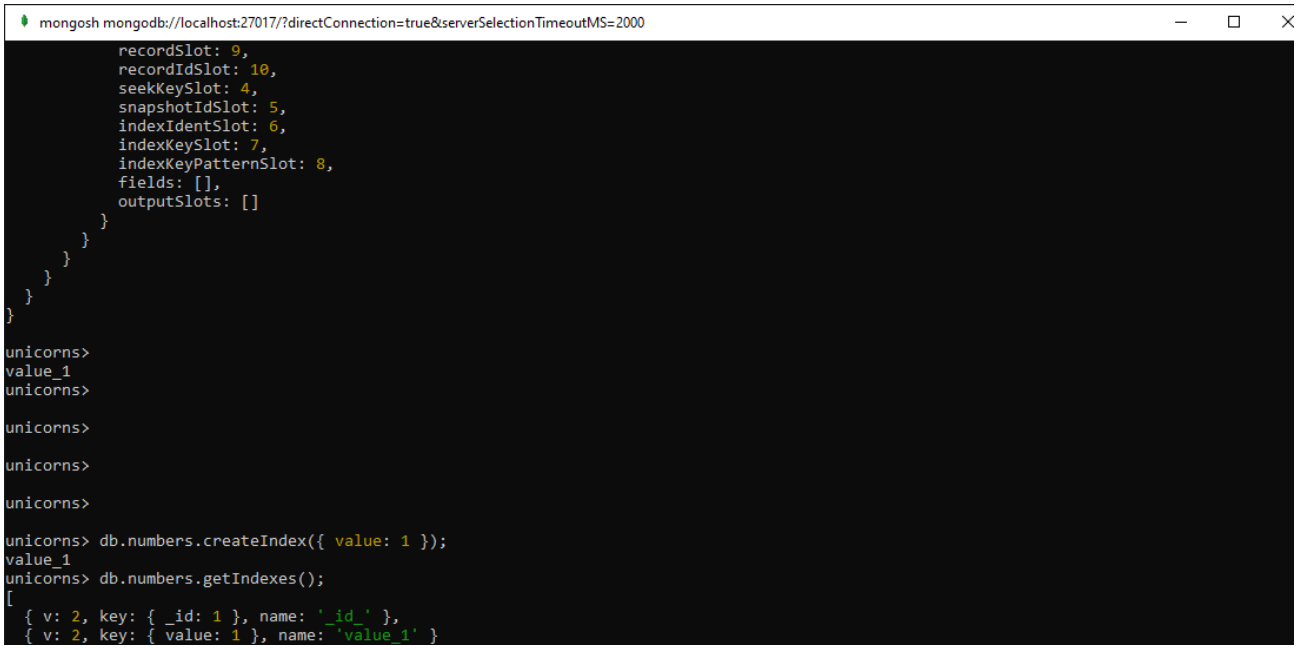
Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра `executionTimeMillis`*)*



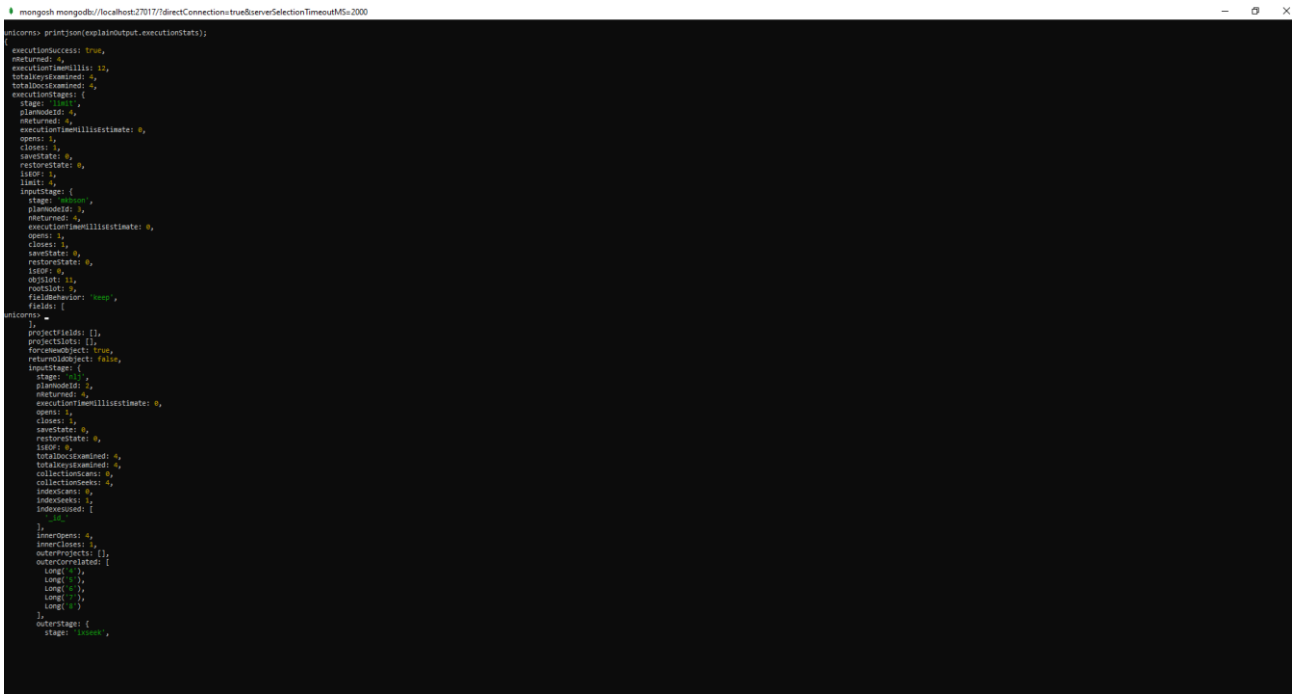
Создайте индекс для ключа `value`.



Получите информацию о всех индексах коллекции `numbres`.



Выполните запрос 2.



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

В обоих случаях потребовалось 12 миллисекунд, скорее всего это произошло из-за того что выборка очень мала.

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

Запрос с индексами в MongoDB должен работать быстрее, чем без них, так как Индекс позволяет MongoDB эффективно находить необходимые документы, сокращая количество документов, которые нужно просмотреть компьютеру.

Вывод

В ходе лабораторной работы были освоены практические навыки по созданию, функций, документов и коллекций в СУБД MongoDB. Были созданы функции на выборку данных, а также были созданы запросы для удаления, замены и вставки данных разными способами. Также был создан курсор и различные запросы на выборку данных.