

**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»

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

Автор: Макунина А.А.

Факультет: ИКТ

Группа: K32421

Преподаватель: Говорова М.М.



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.

**Оборудование:** компьютерный класс.

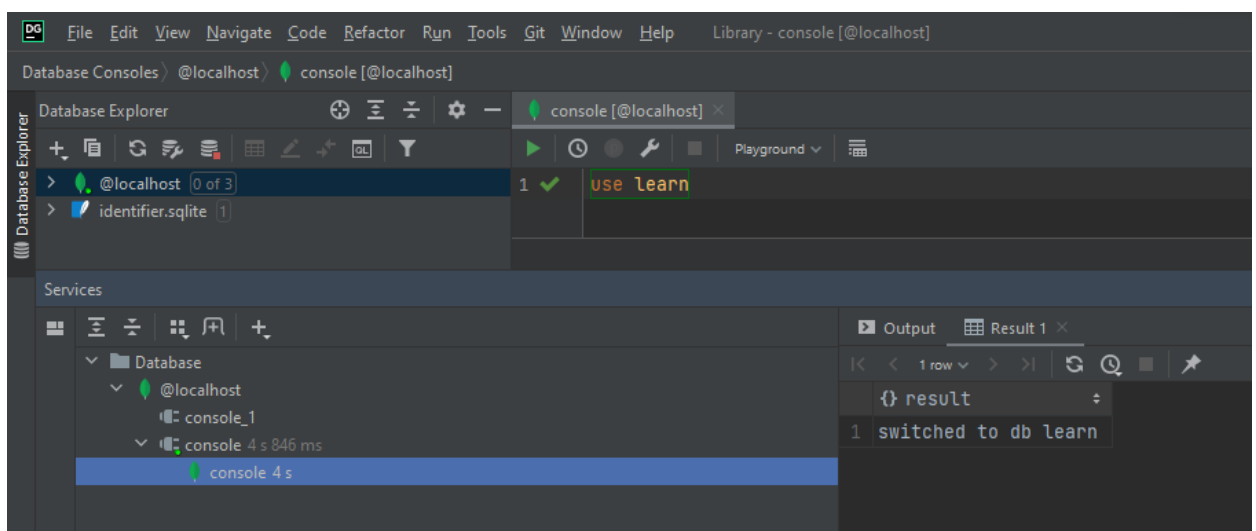
**Программное обеспечение:** СУБД MongoDB 4+, 6.0.6 (текущая).

**Практическое задание:**

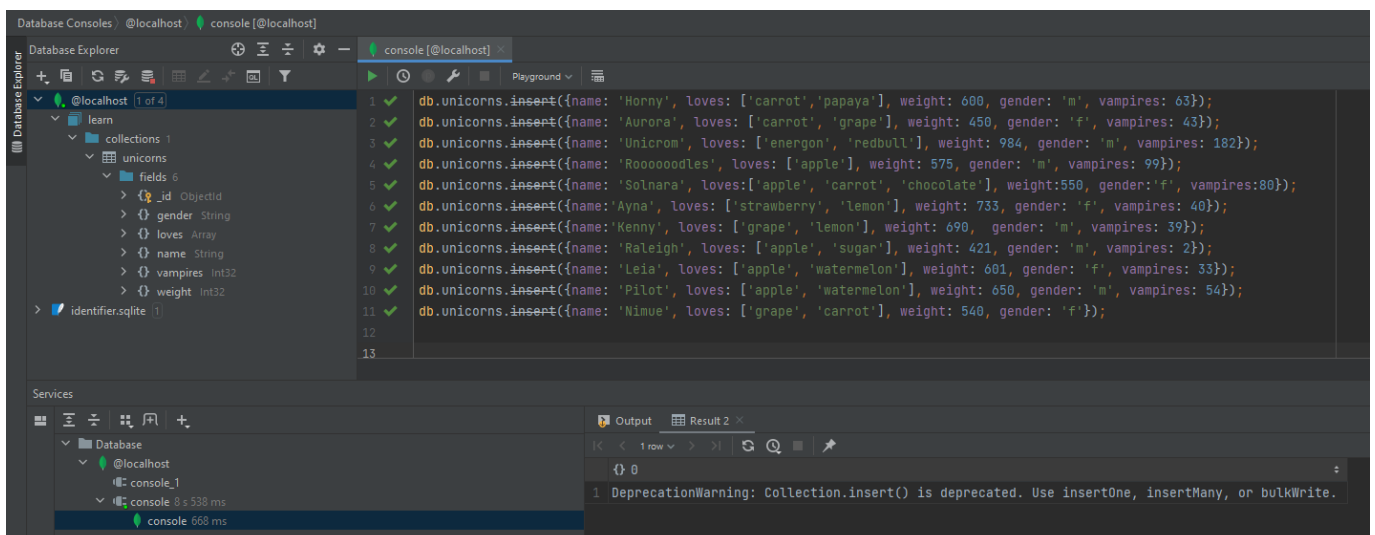
**Вывод:**

### Практическое задание 8.1.1:

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}
```

```
console [localhost] x
1 db.unicorns.insertOne({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165});
2

Output Result 2 x
1 DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
```

4) Проверьте содержимое коллекции с помощью метода *find*.

```
console [localhost] x
1 db.unicorns.find();
2

Output learn.unicorns x
12 rows x
{ "_id" : ..., "gender" : ..., "loves" : ..., "name" : ..., "vampires" : ..., "weight" : ... }
```

	_id	gender	loves	name	vampires	weight
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	63	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	182	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Rooodoodles	99	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	40	733
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	39	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar"]	Raleigh	2	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	54	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca6e5de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	165	704

### 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.

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

Output learn.unicorns x
7 rows x
{ "_id" : ..., "gender" : ..., "loves" : ..., "name" : ..., "vampires" : ..., "weight" : ... }
```

	_id	gender	loves	name	vampires	weight
1	6468ca6e5de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	165	704
2	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	63	600
3	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	39	690
4	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	54	650
5	6468ca035de6ee37cfe6c086	m	["apple", "sugar"]	Raleigh	2	421
6	6468ca005de6ee37cfe6c082	m	["apple"]	Rooodoodles	99	575
7	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	182	984

console [localhost] ×

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

Output learn.unicorns ×

	{_id}	{gender}	{loves}	{name}	{vampires}	{weight}
1	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
2	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	40	733
3	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601

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

console [localhost] ×

```
1 db.unicorns.findOne({loves: 'carrot', gender: 'f'}, {name: 1, loves: 1}).limit(1);
```

Output Result 9 ×

	{_id}	{loves}	{name}
1	6468c9fe5de6ee37cfe6c080	["carrot", "grape"]	Aurora

### Практическое задание 8.1.3:

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

console [localhost] ×

```
1 db.unicorns.find({ gender: 'm' }, { loves: 0, gender: 0 }).sort({ $natural: 1 });
```

Output learn.unicorns ×

	{_id}	{name}	{vampires}	{weight}
1	6468c9fd5de6ee37cfe6c07f	Horny	63	600
2	6468c9ff5de6ee37cfe6c081	Unicrom	182	984
3	6468ca005de6ee37cfe6c082	Roooooodles	99	575
4	6468ca025de6ee37cfe6c085	Kenny	39	690
5	6468ca035de6ee37cfe6c086	Raleigh	2	421
6	6468ca045de6ee37cfe6c088	Pilot	54	650
7	6468ca0e5de6ee37cfe6c08b	Dunx	165	704

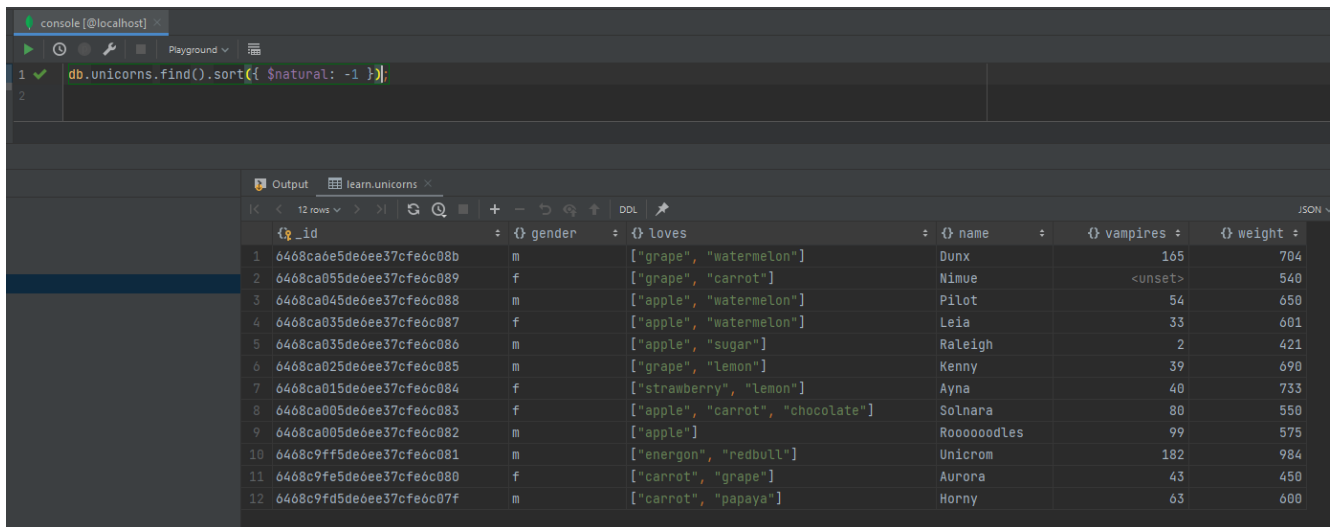
В этом запросе `{ loves: 0, gender: 0 }` является параметром проекции, где `0` указаны поля, которые должны быть исключены из результата.

`sort({ $natural: 1 })` - часть в данном случае необязательна, поскольку естественный порядок документов в коллекции уже будет основан на порядке их вставки. Однако, если

надо явно отсортировать результат в порядке возрастания на основе естественного порядка, можно включить его.

### Практическое задание 8.1.4:

*Вывести список единорогов в обратном порядке добавления.*

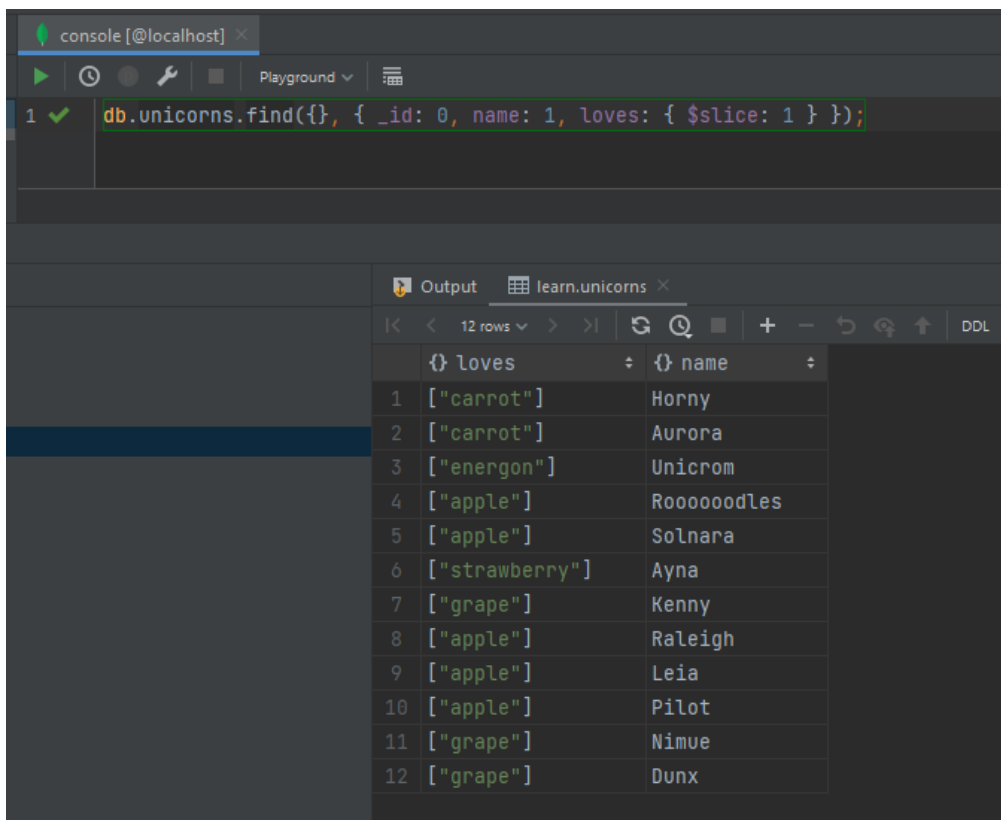


	{_id}	{gender}	{loves}	{name}	{vampires}	{weight}
1	6468ca0e5de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	165	704
2	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
3	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	54	650
4	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
5	6468ca035de6ee37cfe6c086	m	["apple", "sugar"]	Raleigh	2	421
6	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	39	690
7	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	40	733
8	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
9	6468ca005de6ee37cfe6c082	m	["apple"]	Roooooodles	99	575
10	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	182	984
11	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
12	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	63	600

В этом запросе `sort({ $natural: -1 })` сортирует документы в порядке, обратном порядку вставки, используя `$natural` параметр сортировки. `-1` значение указывает на порядок убывания.

### Практическое задание 8.1.5:

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



	{loves}	{name}
1	["carrot"]	Horny
2	["carrot"]	Aurora
3	["energon"]	Unicrom
4	["apple"]	Roooooodles
5	["apple"]	Solnara
6	["strawberry"]	Ayna
7	["grape"]	Kenny
8	["apple"]	Raleigh
9	["apple"]	Leia
10	["apple"]	Pilot
11	["grape"]	Nimue
12	["grape"]	Dunx

В этом запросе:

- `{}` задает пустой фильтр, означающий, что он будет извлекать все документы из коллекции.
- `{_id: 0, name: 1, loves: { $slice: 1 }}` является параметром проекции, который определяет, какие поля включать или исключать в результате. `_id: 0` исключает `_id` поле, `name: 1` включает `name` поле и `loves: { $slice: 1 }` включает только первый элемент `loves` массива.

Этот запрос вернет список единорогов с их именами и первым предпочтением. `_id` поле будет исключено из результата.

### **Практическое задание 8.1.6:**

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

The screenshot shows a MongoDB Playground interface. The console at the top contains the following query:

```
db.unicorns.find({ $and: [ { gender: 'f' }, { weight: { $gte: 500, $lte: 700 } } ], { _id: 0 } });
```

Below the console, the 'Output' tab displays the results in a table format:

	gender	loves	name	vampires	weight
1	f	["apple", "watermelon"]	Leia	33	601
2	f	["apple", "carrot", "chocolate"]	Solnara	80	550
3	f	["grape", "carrot"]	Nimue	<unset>	540

В этом запросе:

- `{ $and: [...] }` используется для указания нескольких условий, которые все должны быть истинными. В этом случае у нас есть два условия внутри массива.
- `{ gender: 'f' }` проверяет наличие единорогов женского пола.
- `{ weight: { $gte: 500, $lte: 700 } }` проверяет наличие единорогов с весом от 500 кг до 700 кг с помощью операторов `$gte` (больше или равно) и `$lte` (меньше или равно).
- `{ _id: 0 }` исключает `_id` поле из результата.

Этот запрос вернет список единорогов женского пола в пределах указанного диапазона веса, без включения поля идентификатора (`_id`).

**\$ne** Оператор используется для извлечения документов, в которых определенное поле не равно определенному значению. **\$in** оператор, с другой стороны, используется для поиска документов, в которых значение поля соответствует любому из значений, указанных в массиве.

### **Практическое задание 8.1.7:**

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

	gender	loves	name	vampires	weight
1	m	["grape", "lemon"]	Kenny	39	690
2	m	["grape", "watermelon"]	Dunx	165	704

**\$exists** оператор позволяет извлекать документы, в которых присутствует или отсутствует определенный ключ. Установив для параметра значение **true**, запрос вернет только те документы, в которых присутствует указанный ключ.

### Практическое задание 8.1.8:

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

	_id	gender	loves	name	weight
1	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	540

### Практическое задание 8.1.9:

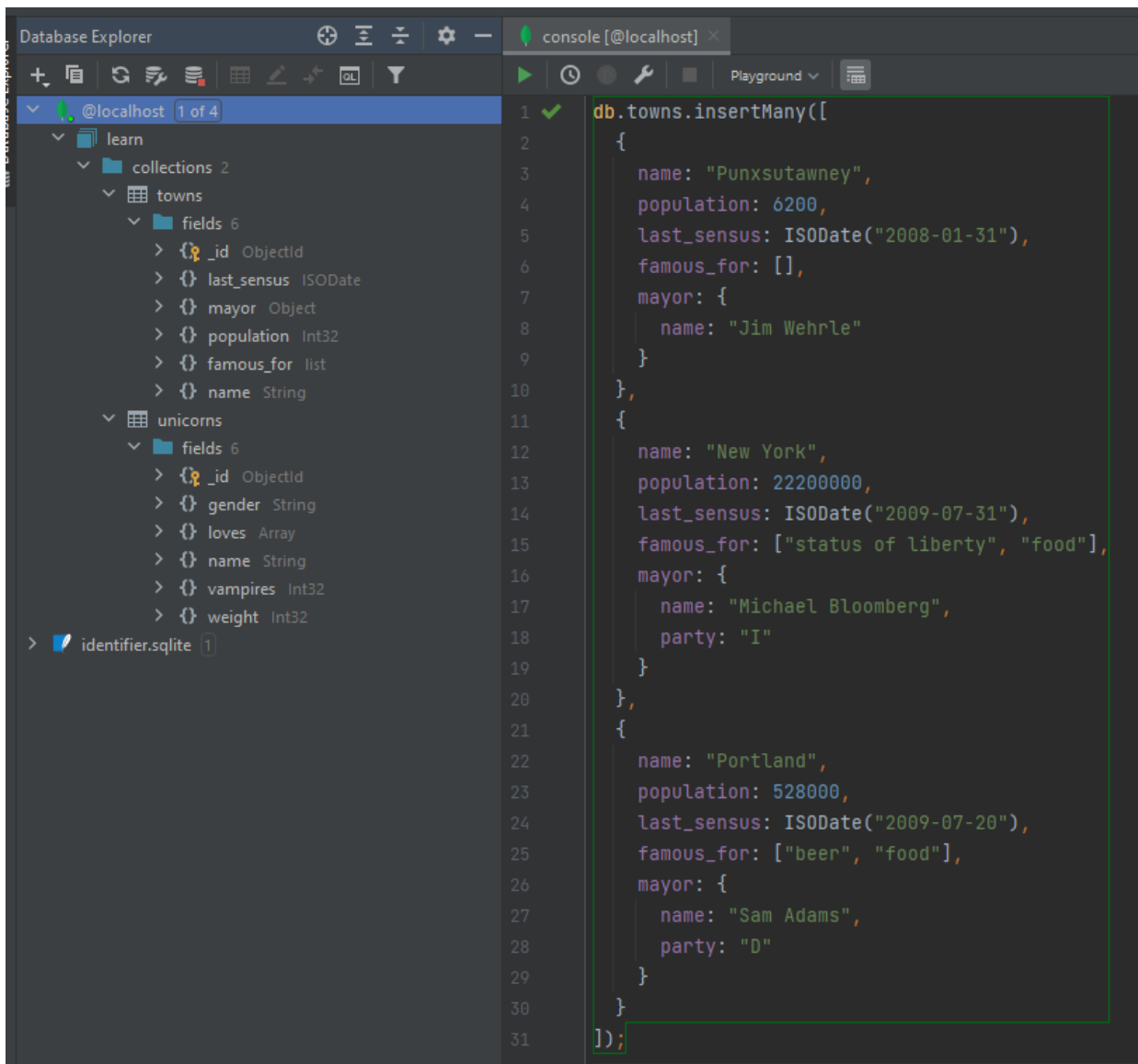
*Вывести список упорядоченный список имен самцов единорогов с информацией об их первом предпочтении.*

	firstPreference	name
1	grape	Dunx
2	carrot	Horny
3	grape	Kenny
4	apple	Pilot
5	apple	Raleigh
6	apple	Rooooooodles
7	energon	Unicrom

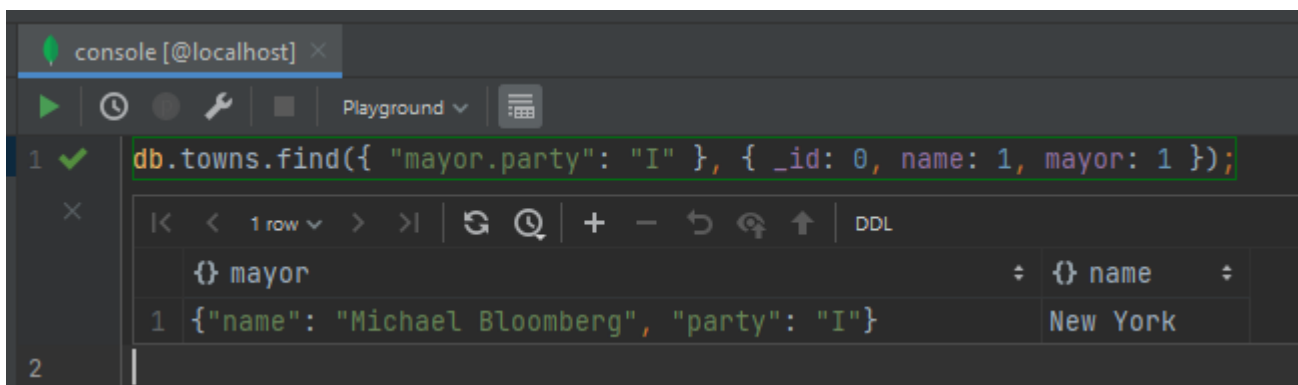
В этом запросе **\$match** этап фильтрует самцов единорогов. **\$project** Этап включает только **name** поле и извлекает первое предпочтение с помощью **\$arrayElemAt** оператора. **\$sort** Этап упорядочивает результаты по имени.

### Практическое задание 8.2.1:

*1) Создайте коллекцию `towns`, включающую следующие документы:*



- 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.



```
console [localhost] x
Playground
1 db.towns.find({ "mayor.party": { $exists: false } }, { _id: 0, name: 1, mayor: 1 });
```

{ } mayor	{ } name
{ "name": "Jim Wehrle" }	Punxsutawney

### 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`.

```
console [localhost] x
Playground
1 function printMaleUnicorns() {
2   db.unicorns.find({ gender: 'm' }).forEach(function(unicorn) {
3     print(unicorn.name);
4   });
5 }
6 var cursor = db.unicorns.find({ gender: 'm' }).sort({ name: 1 }).limit(2);
7
8 cursor.forEach(function(unicorn) {
9   print(unicorn.name);
10 });
```

{ } result
1 js function

{ } 0
1 Dunx
2 Horny

### Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
1 var count = db.unicorns.count({ gender: 'f', weight: { $gte: 500, $lte: 600 } });
2 print("Количество самок единорогов: " + count);
```

The console output shows a table with one row: "Количество самок единорогов: 2".

Этот запрос использует оператор **\$gte** (больше или равно) и **\$lte** (меньше или равно) для определения диапазона веса от полутонны (500 кг) до 600 кг и фильтрует только самок (**gender: 'f'**). Затем функция **count()** используется для подсчета количества соответствующих документов. Результат выводится с помощью функции **print()**.

#### Практическое задание 8.2.4:

*Вывести список предпочтений.*

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

The console output shows a table with 11 rows of preferences: apple, carrot, chocolate, energon, grape, lemon, papaya, redbull, strawberry, sugar, and watermelon.

Если требуется выполнить более сложные операции над данными, то рекомендуется использовать метод **aggregate**.

#### Практическое задание 8.2.5:

*Посчитать количество особей единорогов обоих полов.*

```
console [:@localhost] x
Playground
1 ✓ db.unicorns.aggregate([ { $group: { _id: "$gender", count: { $sum: 1 } } } ]);
```

	_id	count
1	m	7
2	f	5

### Практическое задание 8.2.6:

1. Выполнить команду:

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

2. Проверить содержимое коллекции unicorns.

```
console [:@localhost] x
Playground
1 db.unicorns.insertOne({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'});
2
3 ✓ db.unicorns.find();
```

	_id	gender	loves	name	vampires	weight
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	63	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	182	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Roooooodles	99	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	40	733
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	39	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar"]	Raleigh	2	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	54	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca0e5de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	165	704
13	6469e8ad9a165e1c306c78e2	m	["grape"]	Barney	<unset>	340

В MongoDB команда `save()` устарела и рекомендуется использовать команду `insertOne()` для добавления нового документа в коллекцию.

### Практическое задание 8.2.7:

1. Для самки единорога Ауна внести изменения в БД: теперь ее вес 800, она убила 51 вампира.
2. Проверить содержимое коллекции unicorns.

console [localhost] x

Playground

```

1 db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight: 800, vampires: 51}});
2
3 db.unicorns.find();

```

	{_id}	{gender}	{loves}	{name}	{vampires}	{weight}
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	63	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	182	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Rooodooles	99	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	51	800
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	39	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar"]	Raleigh	2	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	54	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca06e5de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	165	704
13	6469e8ad9a165e1c306c78e2	m	["grape"]	Barney	<unset>	340

В MongoDB команда update() устарела и рекомендуется использовать команду updateOne() для добавления нового документа в коллекцию.

### Практическое задание 8.2.8:

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

console [localhost] x

Playground

```

1 db.unicorns.updateOne({name: 'Raleigh'}, {$addToSet: {loves: 'redbull'}});
2
3 db.unicorns.find();
4
5

```

	{_id}	{gender}	{loves}	{name}	{vampires}	{weight}
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	63	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	182	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Rooodooles	99	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	51	800
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	39	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar", "redbull"]	Raleigh	2	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	54	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca06e5de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	165	704
13	6469e8ad9a165e1c306c78e2	m	["grape"]	Barney	<unset>	340

### Практическое задание 8.2.9:

1. Всем самцам единорогов увеличить количество убитых вампиров на 5.
2. Проверить содержимое коллекции unicorns.

console [localhost] x

```

1 db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}});
2
3 db.unicorns.find();
4

```

	{_id}	{gender}	{loves}	{name}	{vampires}	{weight}
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	68	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	187	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Roooooodles	104	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	51	800
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	44	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar", "redbull"]	Raleigh	7	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon"]	Pilot	59	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca065de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	170	704
13	6469e8ad9a165e1c306c78e2	m	["grape"]	Barney	5	340

Команда '\$inc' увеличит значение поля "vampires" на 5 для всех документов, у которых пол "gender" равен "m" (самцы единорогов).

### Практическое задание 8.2.10:

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

console [localhost] x

```

1 db.towns.updateOne({name: "Portland"}, {$unset: {mayor.party: 1}});
2
3 db.towns.find();
4

```

	{_id}	{famous_for}	{last_sensus}	{mayor}	{name}	{population}
1	6469e5949a165e1c306c78de	["status of liberty", "food"]	2008-01-31T00:00:00.000Z	{name: "Jim Wehrle"}	Punxsutawney	6200
2	6469e5949a165e1c306c78df	["beer", "food"]	2009-07-31T00:00:00.000Z	{name: "Michael Bloomberg", "party": "I"}	New York	22200000
3	6469e5949a165e1c306c78e0	["beer", "food"]	2009-07-20T00:00:00.000Z	{name: "Sam Adams"}	Portland	528000

### Практическое задание 8.2.11:

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

console [localhost] x

```

1 db.unicorns.updateOne({name: "Pilot"}, {$addToSet: {loves: "chocolate"}});
2
3 db.unicorns.find();
4

```

	{_id}	{gender}	{loves}	{name}	{vampires}	{weight}
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	68	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	187	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Roooooodles	104	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	51	800
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	44	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar", "redbull"]	Raleigh	7	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon", "chocolate"]	Pilot	59	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca065de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	170	704
13	6469e8ad9a165e1c306c78e2	m	["grape"]	Barney	5	340

### 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.

	_id	gender	loves	name	vampires	weight
1	6468c9fd5de6ee37cfe6c07f	m	["carrot", "papaya"]	Horny	68	600
2	6468c9fe5de6ee37cfe6c080	f	["carrot", "grape", "sugar", "lemon"]	Aurora	43	450
3	6468c9ff5de6ee37cfe6c081	m	["energon", "redbull"]	Unicrom	187	984
4	6468ca005de6ee37cfe6c082	m	["apple"]	Rooodles	104	575
5	6468ca005de6ee37cfe6c083	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6468ca015de6ee37cfe6c084	f	["strawberry", "lemon"]	Ayna	51	800
7	6468ca025de6ee37cfe6c085	m	["grape", "lemon"]	Kenny	44	690
8	6468ca035de6ee37cfe6c086	m	["apple", "sugar", "redbull"]	Raleigh	7	421
9	6468ca035de6ee37cfe6c087	f	["apple", "watermelon"]	Leia	33	601
10	6468ca045de6ee37cfe6c088	m	["apple", "watermelon", "chocolate"]	Pilot	59	650
11	6468ca055de6ee37cfe6c089	f	["grape", "carrot"]	Nimue	<unset>	540
12	6468ca065de6ee37cfe6c08b	m	["grape", "watermelon"]	Dunx	170	704
13	6469e8ad9a165e1c306c78e2	m	["grape"]	Barney	5	340

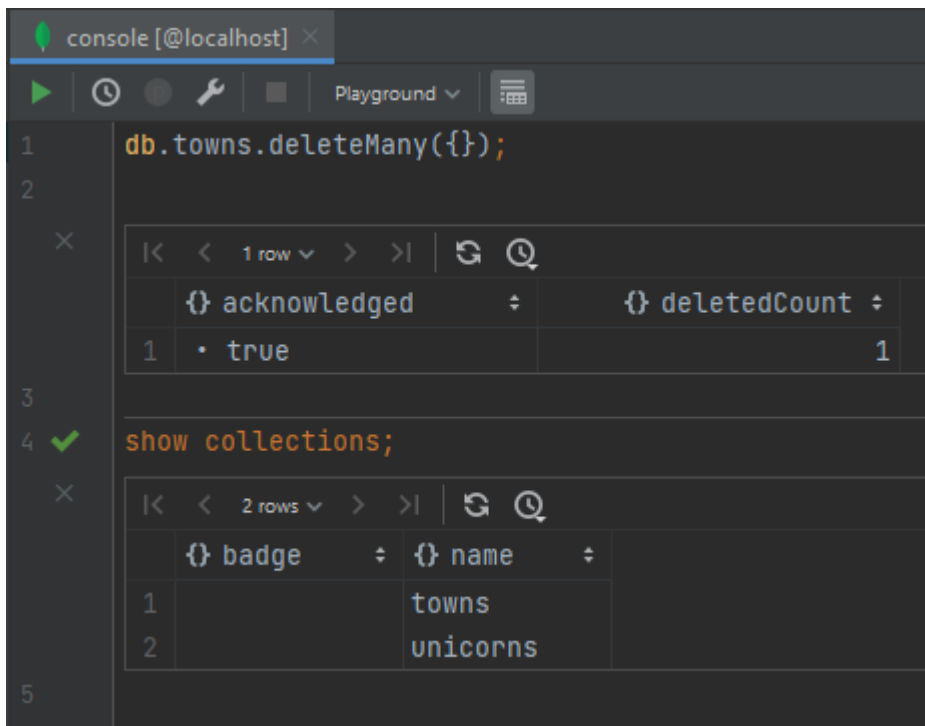
Чтобы изменить информацию о самке единорога Aurora и добавить ей предпочтения к сахару и лимонам, можно использовать оператор \$addToSet с оператором \$each. Эта команда добавит значения "sugar" и "lemon" в массив "loves" для единорога с именем "Aurora" в коллекции "unicorns". Если значения уже существуют в массиве, они не будут добавлены повторно.

### Практическое задание 8.2.13:

- 1) Удалите документы с беспартийными мэрами.
- 2) Проверьте содержание коллекции.

	_id	famous_for	last_sensus	mayor	name	population
1	6469e5949a165e1c306c78df	["status of liberty", "food"]	2009-07-31T00:00:00.000Z	{"name": "Michael Bloomberg", "party": "I"}	New York	22200000

- 3) Очистите коллекцию.
- 4) Просмотрите список доступных коллекций.



### Контрольные вопросы:

1. Как используется оператор точка?

Оператор точка (.) используется для доступа к полям внутри вложенных объектов. Например, `document.field.subfield` обращается к подполю `subfield` внутри поля `field`.

2. Как можно использовать курсор?

Курсоры в MongoDB представляют результат запроса, который можно итерировать для получения последовательности документов. Курсор позволяет эффективно работать с большими наборами данных, возвращая результаты по мере необходимости.

3. Какие возможности агрегирования данных существуют в MongoDB?

В MongoDB существуют различные возможности агрегирования данных, такие как операторы `$group`, `$match`, `$sort`, `$project` и другие. Они позволяют выполнять сложные запросы и аналитику на данных внутри коллекций.

4. Какая из функций `save` или `update` более детально позволит настроить редактирование документов коллекции?

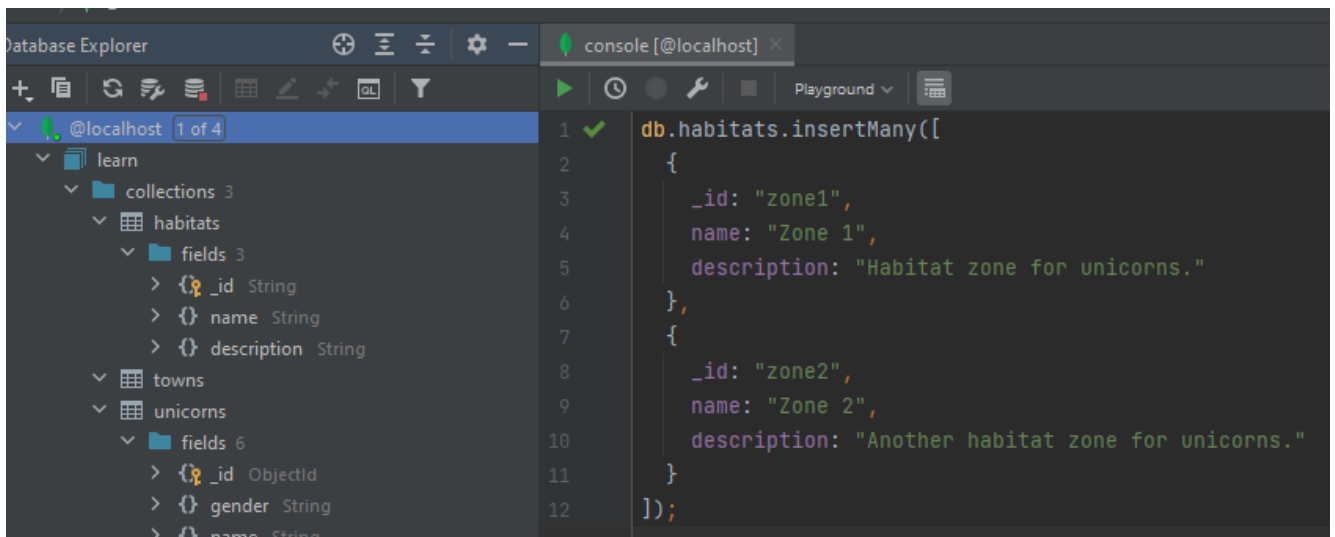
Функция `update(One)` позволяет более детально настроить редактирование документов, так как она предоставляет больше операторов и возможностей, включая `$set`, `$unset`, `$inc`, `$push`, `$addToSet` и другие.

5. Как происходит удаление документов из коллекции по умолчанию?

При удалении документов из коллекции по умолчанию используется функция `deleteMany({})`, которая удаляет все документы в коллекции.

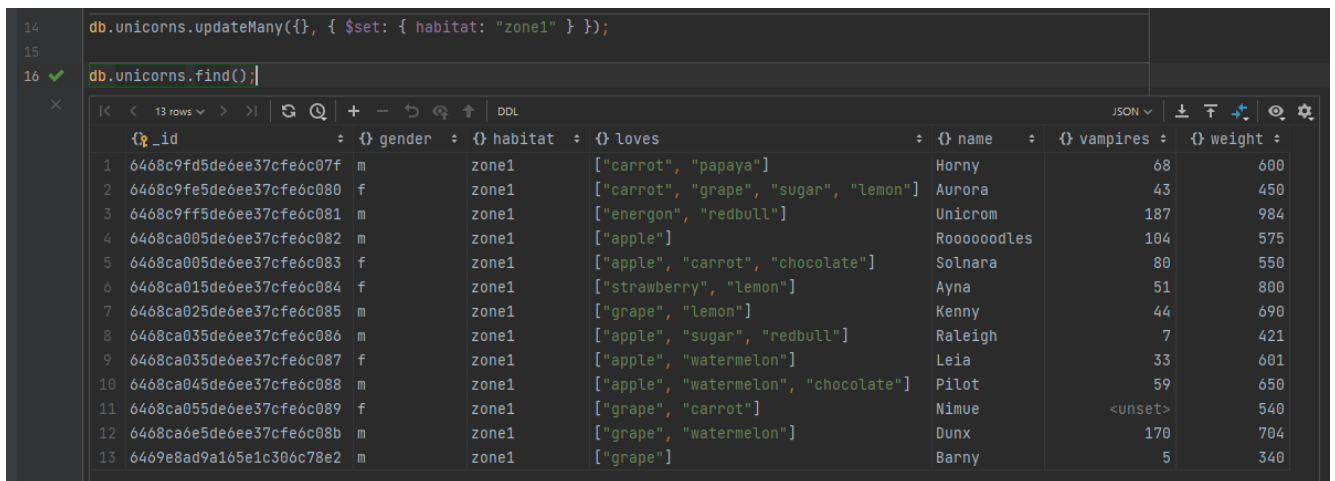
### Практическое задание 8.3.1:

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



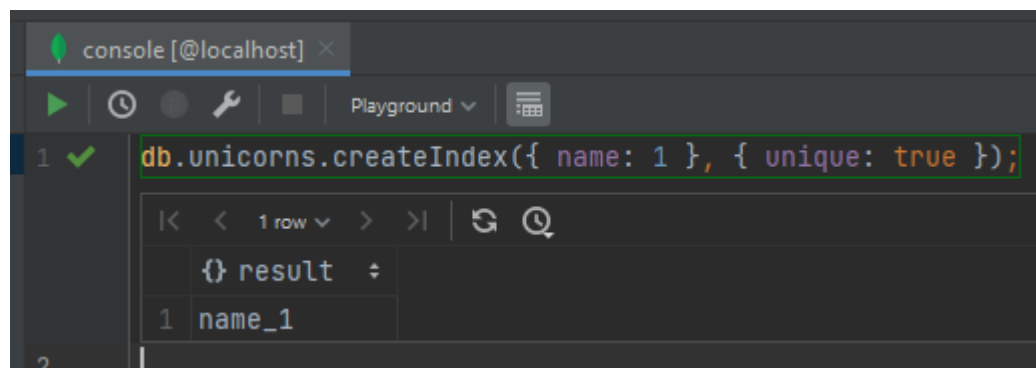
2) Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания.

3) Проверьте содержание коллекции единорогов.



### Практическое задание 8.3.2:

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



Ограничения: значение поля, по которому идет индексация, не должно быть больше 1024 байт.

### Практическое задание 8.3.3:



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

```

1 db.unicorns.getIndexes();
2
3 db.unicorns.dropIndexes();
4
5 db.unicorns.dropIndex('_id_');
6

```

key	name	v	unique
{ "_id": new NumberInt("1") }	_id_	2	<unset>
{ "name": new NumberInt("1") }	name_1	2	true
{ "dob": new NumberInt("1") }	dob_1	2	<unset>
{ "weight": new NumberInt("1") }	weight_1	2	<unset>
{ "gender": new NumberInt("1") }	gender_1	2	<unset>
{ "vampires": new NumberInt("1") }	vampires_1	2	<unset>

msg	nIndexesWas	ok
non-_id indexes dropped for collection	6	1

Command failed with error 72 (InvalidOptions): 'cannot drop \_id index' on server localhost:27017. The full response is {"ok":0.0,"errmsg":"'cannot drop \_id index','code':72,'codeName':'InvalidOptions'}

### 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.

```

1 for (i = 0; i < 100000; i++) {
2   db.numbers.insertOne({value: i});
3 }
4
5 db.numbers.find().sort({'_id': -1}).limit(4);
6

```

_id	value
6469f2b69a165e1c306dff84	99999
6469f2b69a165e1c306dff83	99998
6469f2b69a165e1c306dff82	99997
6469f2b69a165e1c306dff81	99996

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

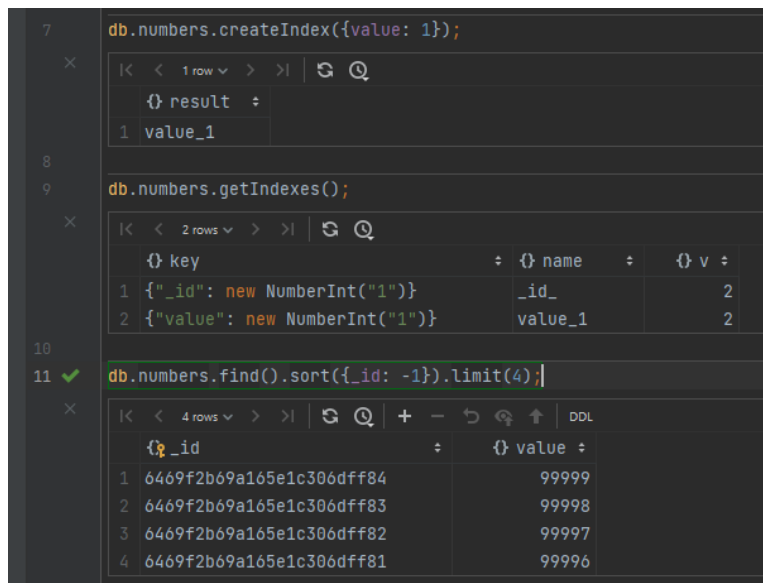
300 ms

- 4) Создайте индекс для ключа *value*.

5) Получите информацию о всех индексах коллекции *numbers*.

6) Выполните запрос 2.

```
7 db.numbers.createIndex({value: 1});
8
9 db.numbers.getIndexes();
10
11 db.numbers.find().sort({_id: -1}).limit(4);
```



key	name	v
{ "_id": new NumberInt("1") }	_id_	2
{ "value": new NumberInt("1") }	value_1	2

_id	value
6469f2b69a165e1c306dff84	99999
6469f2b69a165e1c306dff83	99998
6469f2b69a165e1c306dff82	99997
6469f2b69a165e1c306dff81	99996

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

225 ms

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

Запрос с индексацией обычно выполняется быстрее, потому что индекс позволяет базе данных эффективно найти и выбрать нужные документы, минимизируя количество проверок и сравнений.

Когда индекс не используется, СУБД должна просмотреть все документы в коллекции и выполнить полное сканирование (*full scan*), чтобы найти и выбрать нужные документы. Это требует больше времени и ресурсов, особенно при большом объеме данных.

### Контрольные вопросы:

1. Назовите способы связывания коллекций в MongoDB.

Вложенные документы: В этом случае одна коллекция может содержать документы, которые включают вложенные документы другой коллекции. Связь между документами осуществляется по значению полей.

Ссылки на документы: В этом случае одна коллекция содержит ссылки на документы из другой коллекции. Связь между документами осуществляется по идентификатору документа.

2. Сколько индексов можно установить на одну коллекцию в БД MongoDB.

В MongoDB можно установить до 64 индексов на одну коллекцию. Ограничение в 64 индекса связано с ограничениями хранилища индексов в MongoDB.

3. Как получить информацию обо всех индексах базы данных MongoDB?

Для получения информации о всех индексах базы данных MongoDB можно использовать метод `getIndexes()` в рамках операций на коллекции или команду `db.collection.getIndexes()` в оболочке командной строки.

**Вывод:** в ходе данной лабораторной работе были приобретены практические навыки работы с базой данных MongoDB, используя интерфейс DataGrip. Основные задачи, с которыми я столкнулась, включали создание, чтение, обновление и удаление данных (CRUD операции), работу с вложенными объектами в коллекциях, выполнение агрегации и изменение данных.

Особое внимание было уделено работе со ссылками и индексами в MongoDB: были изучены различные способы связывания коллекций, включая вложенные объекты и ссылки на документы из других коллекций. Это позволило эффективно организовать структуру данных и упростить выполнение запросов. Также были изучены принципы работы с индексами в MongoDB. Создание индексов на ключевые поля позволяет значительно ускорить поиск и сортировку данных.

В процессе выполнения работы я столкнулась с устаревшими методами, указанными в методичке, и пришлось использовать альтернативные методы, чтобы достичь требуемого результата. Об этом мне сообщил современный интерфейс DataGrip.

В целом, данная лабораторная работа дала мне ценный опыт работы с MongoDB. Эти навыки будут полезны при разработке баз данных и оптимизации производительности систем, работающих с СУБД NoSQL.