

Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО  
ОБРАЗОВАНИЯ  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет: **Инфокоммуникационных технологий**  
Образовательная программа: **Интеллектуальные системы в гуманитарной сфере**  
Направление подготовки: **45.03.04 Интеллектуальные системы в гуманитарной сфере**

Лабораторная работа №5  
«Работа с БД в СУБД MongoDB»

по дисциплине:  
«ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ БАЗ ДАННЫХ»

Выполнила:  
Чагина Вероника Александровна,  
группа К32422  
Преподаватель:  
Говорова Марина Михайловна

**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.

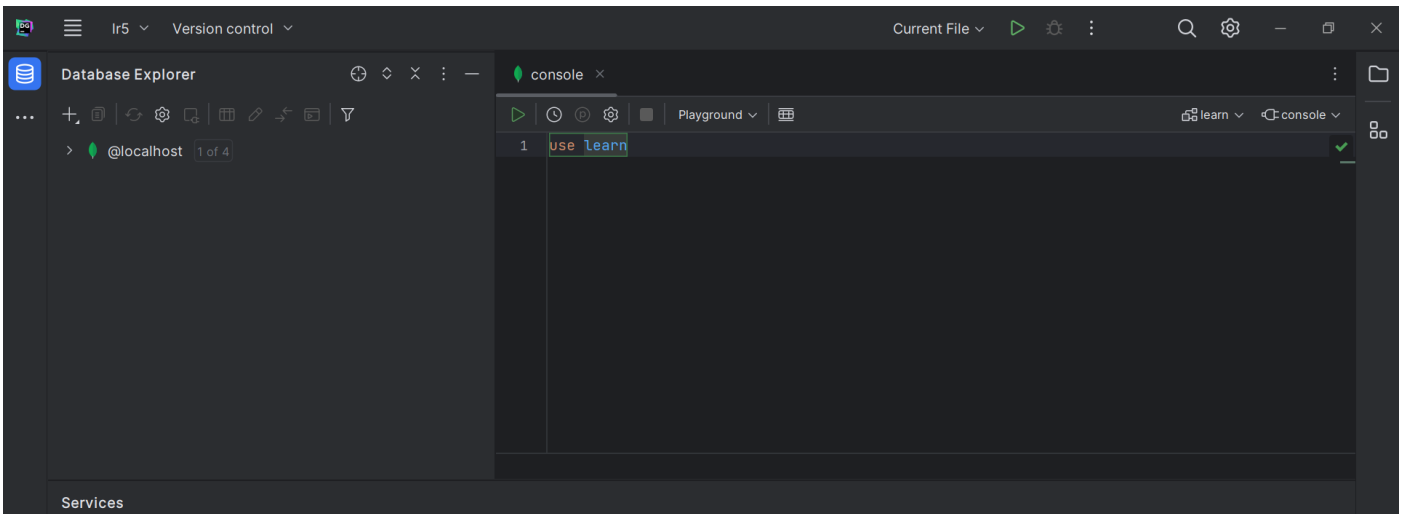
**Оборудование:** компьютерный класс.

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

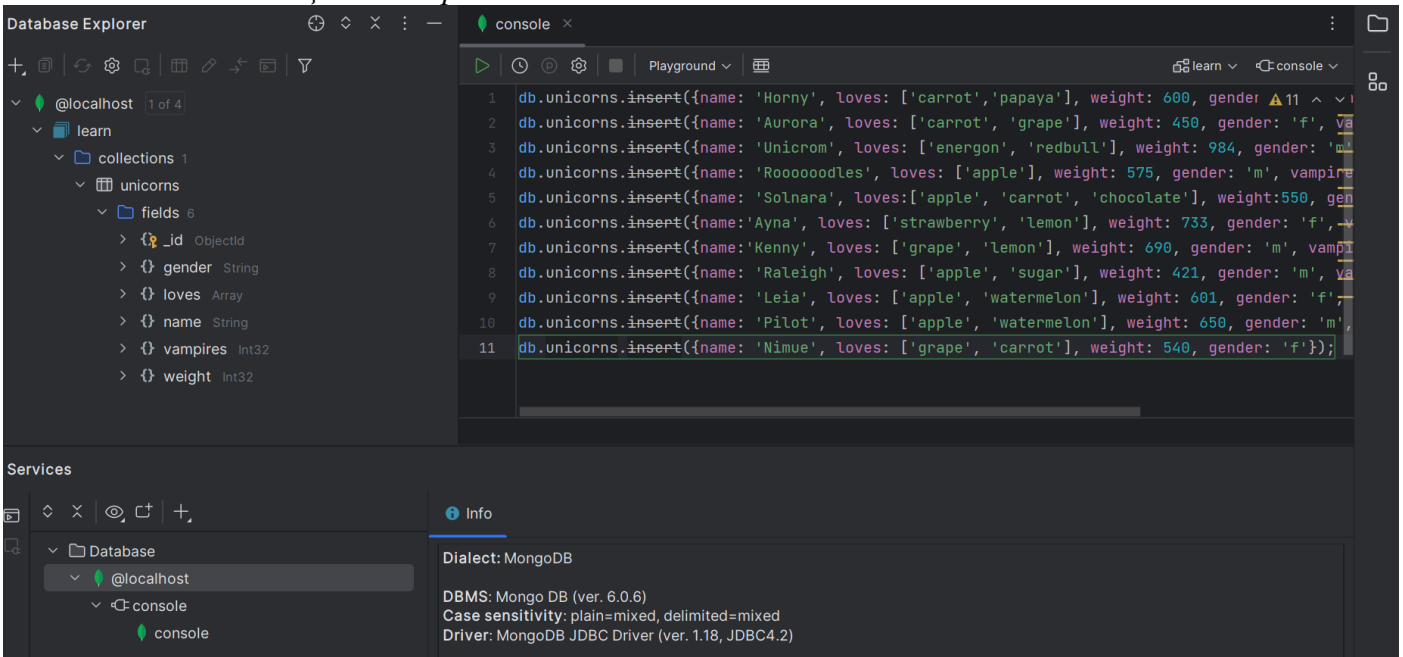
## Выполнение:

### Практическое задание 8.1.1:

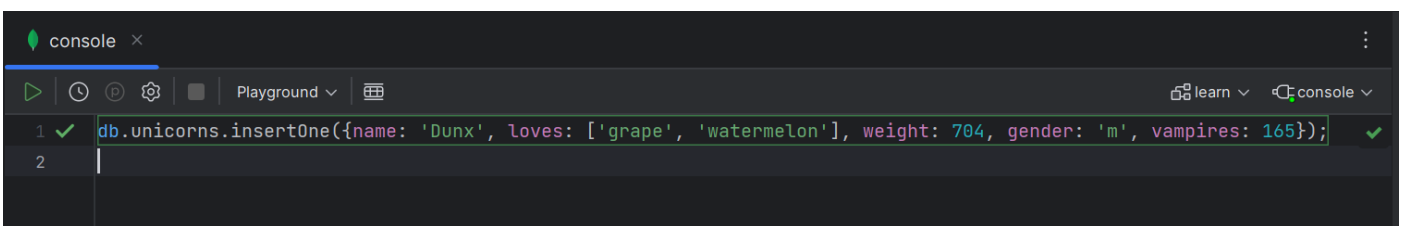
#### 1. Создайте базу данных *learn*.



#### 2. Заполните коллекцию единорогов *unicorns*:



#### 3. Используя второй способ, вставьте в коллекцию единорогов документ:



4. Проверьте содержимое коллекции с помощью метода `find`.

The screenshot shows a MongoDB Playground interface. The console at the top displays the command `db.unicorns.find()` which has been executed successfully. Below the console, the output is displayed as a table with 12 rows. The table has columns for `_id`, `gender`, `loves`, `name`, and `vampires`. The data is as follows:

	<code>_id</code>	<code>gender</code>	<code>loves</code>	<code>name</code>	<code>vampires</code>
1	64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	
2	64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	
3	64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	
4	64735098d08ff908a77a8d25	m	["apple"]	Rooodooles	
5	64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	SoInara	
6	64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	
7	64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	
8	64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	
9	6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	
10	6473509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	
11	6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<uns
12	6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	

### 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.

The screenshot shows a MongoDB Playground interface. The console at the top displays the command `db.unicorns.find({gender: 'm'}).sort({name: 1})` which has been executed successfully. Below the console, the output is displayed as a table with 7 rows. The table has columns for `_id`, `gender`, `loves`, `name`, `vampires`, and `weight`. The data is as follows:

	<code>_id</code>	<code>gender</code>	<code>loves</code>	<code>name</code>	<code>vampires</code>	<code>weight</code>
1	6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	165	704
2	64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	63	600
3	64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	39	690
4	6473509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	54	650
5	64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	2	421
6	64735098d08ff908a77a8d25	m	["apple"]	Rooodooles	99	575
7	64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	182	984

console x

Playground

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

Output learn.unicorns

	_id	gender	loves	name	vampires	weight
1	64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	450
2	64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	40	733
3	6473509ad08ff908a77a8d2a	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 x

Playground

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

Output Result 8

	_id	loves	name
1	64735097d08ff908a77a8d23	["carrot", "grape"]	Aurora

console x

Playground

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

Output learn.unicorns

	_id	gender	loves	name	vampires	weight
1	64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	450

### Практическое задание 8.1.3:

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

The screenshot shows a MongoDB console with the following query in the input field:

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

The output is a table with 7 rows, showing the results of the query. The columns are: `_id`, `name`, `vampires`, and `weight`.

	<code>_id</code>	<code>name</code>	<code>vampires</code>	<code>weight</code>
1	64735096d08ff908a77a8d22	Horny	63	600
2	64735097d08ff908a77a8d24	Unicrom	182	984
3	64735098d08ff908a77a8d25	Roooooodles	99	575
4	64735099d08ff908a77a8d28	Kenny	39	690
5	64735099d08ff908a77a8d29	Raleigh	2	421
6	6473509ad08ff908a77a8d2b	Pilot	54	650
7	6473886b78222618b5756dc0	Dunx	165	704

### Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

The screenshot shows a MongoDB console with the following query in the input field:

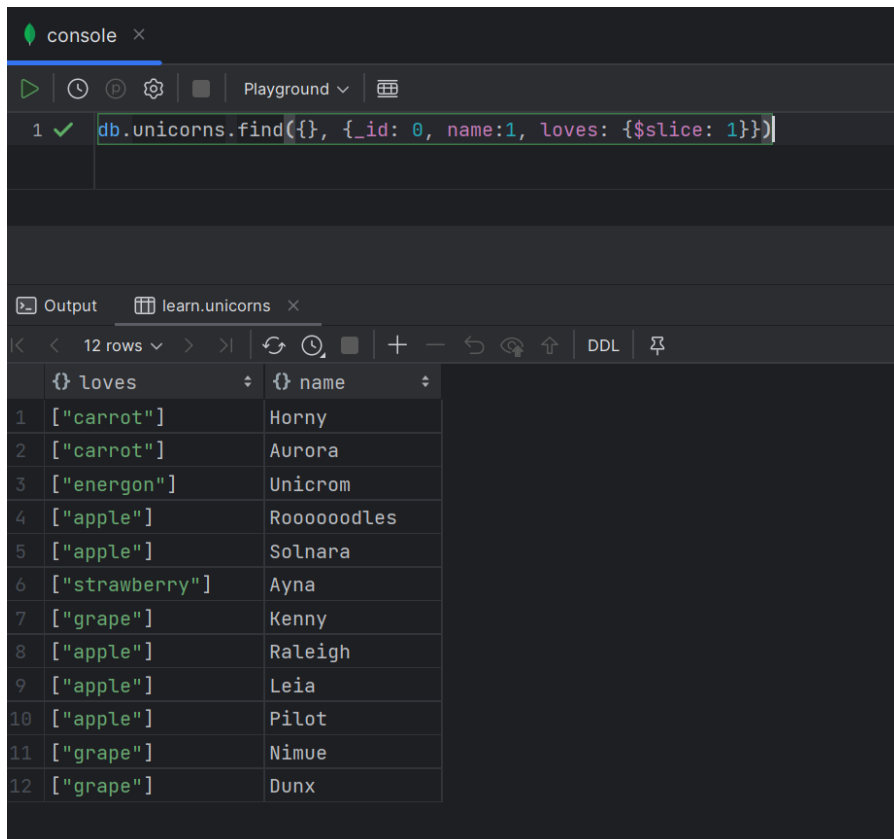
```
db.unicorns.find().sort({$natural: -1})
```

The output is a table with 12 rows, showing the results of the query. The columns are: `_id`, `gender`, `loves`, `name`, `vampires`, and `weight`.

	<code>_id</code>	<code>gender</code>	<code>loves</code>	<code>name</code>	<code>vampires</code>	<code>weight</code>
1	6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	165	704
2	6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	540
3	6473509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	54	650
4	6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	601
5	64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	2	421
6	64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	39	690
7	64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	40	733
8	64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	550
9	64735098d08ff908a77a8d25	m	["apple"]	Roooooodles	99	575
10	64735097d08ff908a77a8d24	m	["energion", "redbull"]	Unicrom	182	984
11	64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	450
12	64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	63	600

### Практическое задание 8.1.5:

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

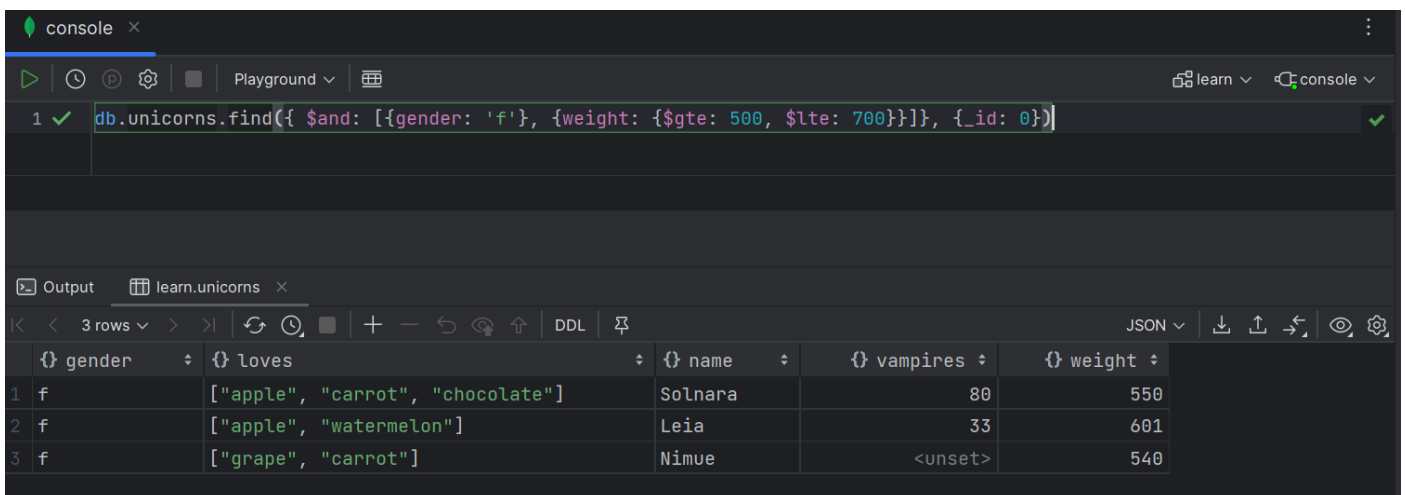


The screenshot shows a MongoDB Playground interface. The console at the top contains the query: `db.unicorns.find({}, {_id: 0, name: 1, loves: {$slice: 1}})`. Below the console, the 'Output' tab displays 12 rows of results. Each row contains a 'loves' field (an array of one string) and a 'name' field (a string).

	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

### 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 query: `db.unicorns.find({ $and: [{gender: 'f'}, {weight: {$gte: 500, $lte: 700}}]}, {_id: 0})`. Below the console, the 'Output' tab displays 3 rows of results. Each row contains a 'gender' field (a string 'f'), a 'loves' field (an array of strings), a 'name' field (a string), a 'vampires' field (a number), and a 'weight' field (a number).

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

### Практическое задание 8.1.7:

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

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

Output:

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

### Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

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

Output:

_id	gender	loves	name	weight
6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	540

### Практическое задание 8.1.9:

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

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

Output:

	loves	name
1	["grape"]	Dunx
2	["carrot"]	Horny
3	["grape"]	Kenny
4	["apple"]	Pilot
5	["apple"]	Raleigh
6	["apple"]	Roooooodles
7	["energon"]	Unicrom



## Практическое задание 8.2.1:

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

The screenshot shows the MongoDB Database Explorer on the left, displaying the 'town' collection with its fields: `_id` (ObjectId), `famous_for` (Array), `last_sensus` (ISODate), `mayor` (Object), `name` (String), and `populatiuon` (Int32). The console on the right shows the execution of the following JavaScript code:

```
db.town.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"}}])
```

The Services panel at the bottom shows the database structure and the execution of the command, indicating it completed in 602 ms.

2. Сформировать запрос, который возвращает список городов с независимыми мэрами

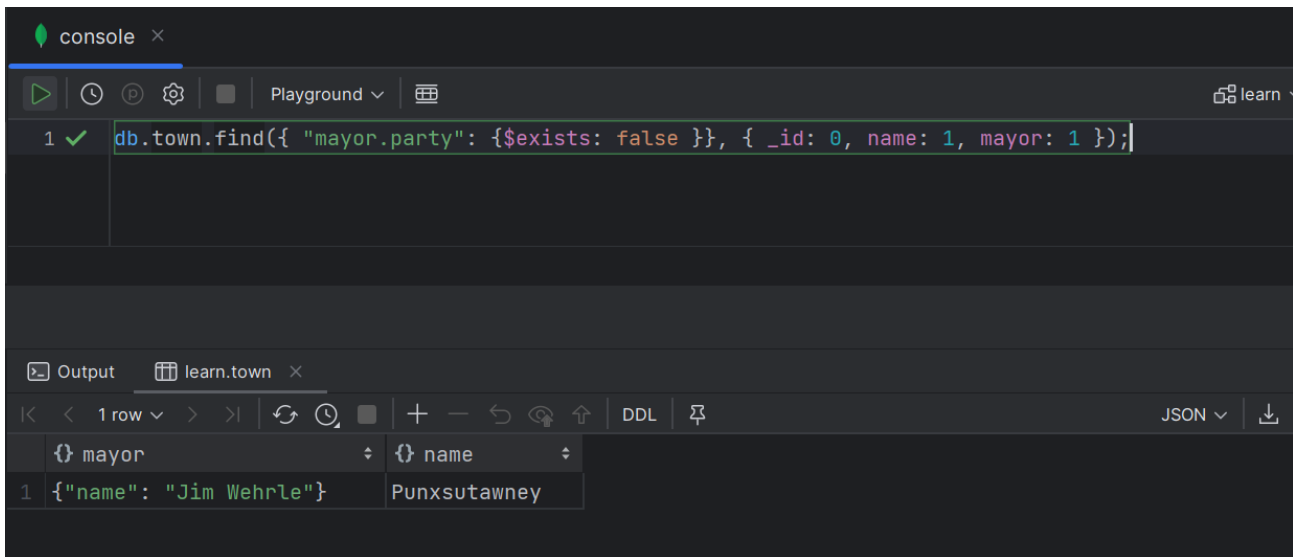
The screenshot shows the MongoDB console with the following query:

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

The output is displayed in a table below the console:

	{ mayor	{ name
1	{"name": "Michael Bloomberg", "party": "I"}	New York

3. Вывести только название города и информацию о мэре. Сформировать запрос, который возвращает список беспартийных мэров. Вывести только название города и информацию о мэре.



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

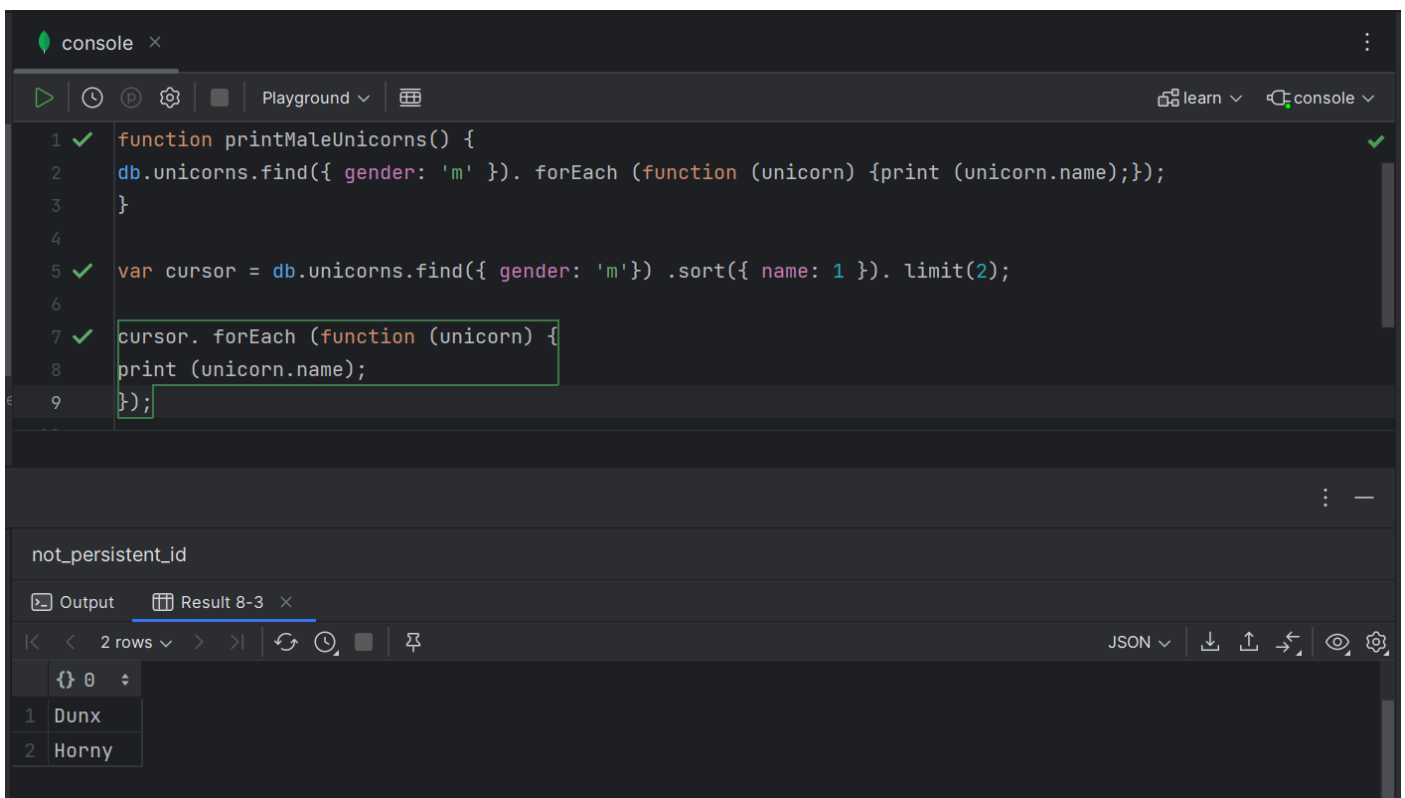
```
1 db.town.find({ "mayor.party": { $exists: false } }, { _id: 0, name: 1, mayor: 1 });
```

Below the console, the 'Output' tab is selected, showing a single row of results in a table format:

	mayor	name
1	{ "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`.



The screenshot shows a MongoDB Playground interface. The console contains the following code:

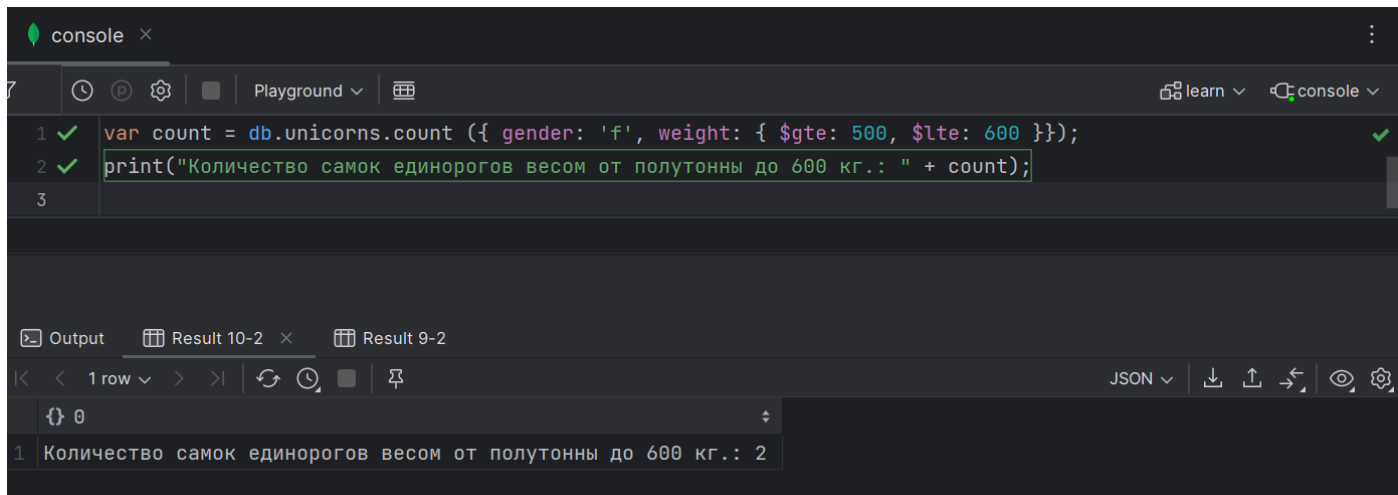
```
1 function printMaleUnicorns() {  
2   db.unicorns.find({ gender: 'm' }). forEach (function (unicorn) {print (unicorn.name)});  
3 }  
4  
5 var cursor = db.unicorns.find({ gender: 'm'}) .sort({ name: 1 }). limit(2);  
6  
7 cursor. forEach (function (unicorn) {  
8   print (unicorn.name);  
9 });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

Below the console, the 'Output' tab is selected, showing two rows of results in a table format:

	0
1	Dunx
2	Horny

### Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.



The screenshot shows a MongoDB Playground interface. The console has two lines of code: `var count = db.unicorns.count ({ gender: 'f', weight: { $gte: 500, $lte: 600 }});` and `print("Количество самок единорогов весом от полутонны до 600 кг.: " + count);`. Below the console, the 'Output' tab is active, displaying a single row with the result: 'Количество самок единорогов весом от полутонны до 600 кг.: 2'.

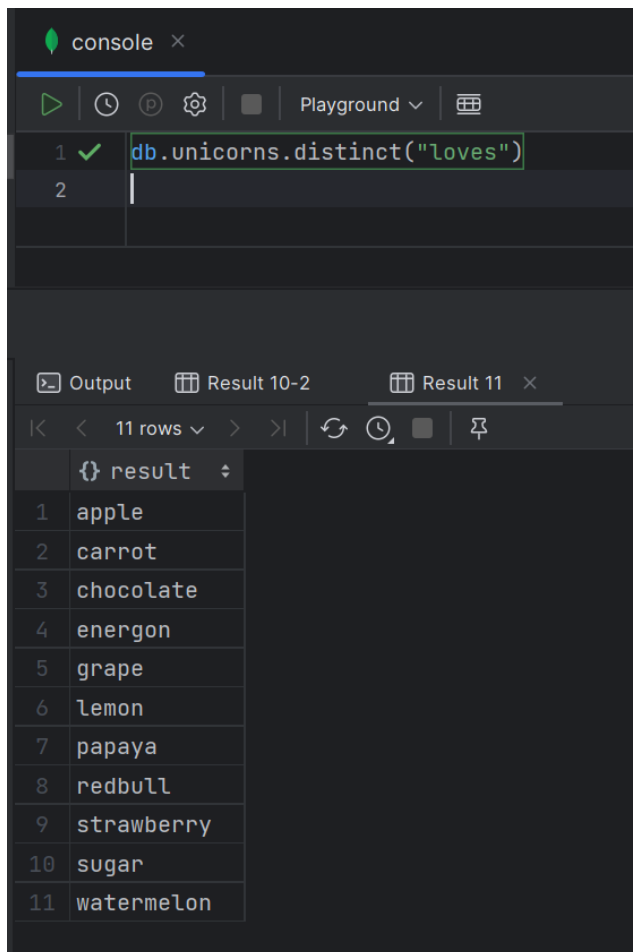
```
1 ✓ var count = db.unicorns.count ({ gender: 'f', weight: { $gte: 500, $lte: 600 }});
2 ✓ print("Количество самок единорогов весом от полутонны до 600 кг.: " + count);
3
```

Output

1	Количество самок единорогов весом от полутонны до 600 кг.: 2

### Практическое задание 8.2.4:

Вывести список предпочтений.



The screenshot shows a MongoDB Playground interface. The console has one line of code: `db.unicorns.distinct("loves")`. Below the console, the 'Output' tab is active, displaying 11 rows of results under the column 'result'.

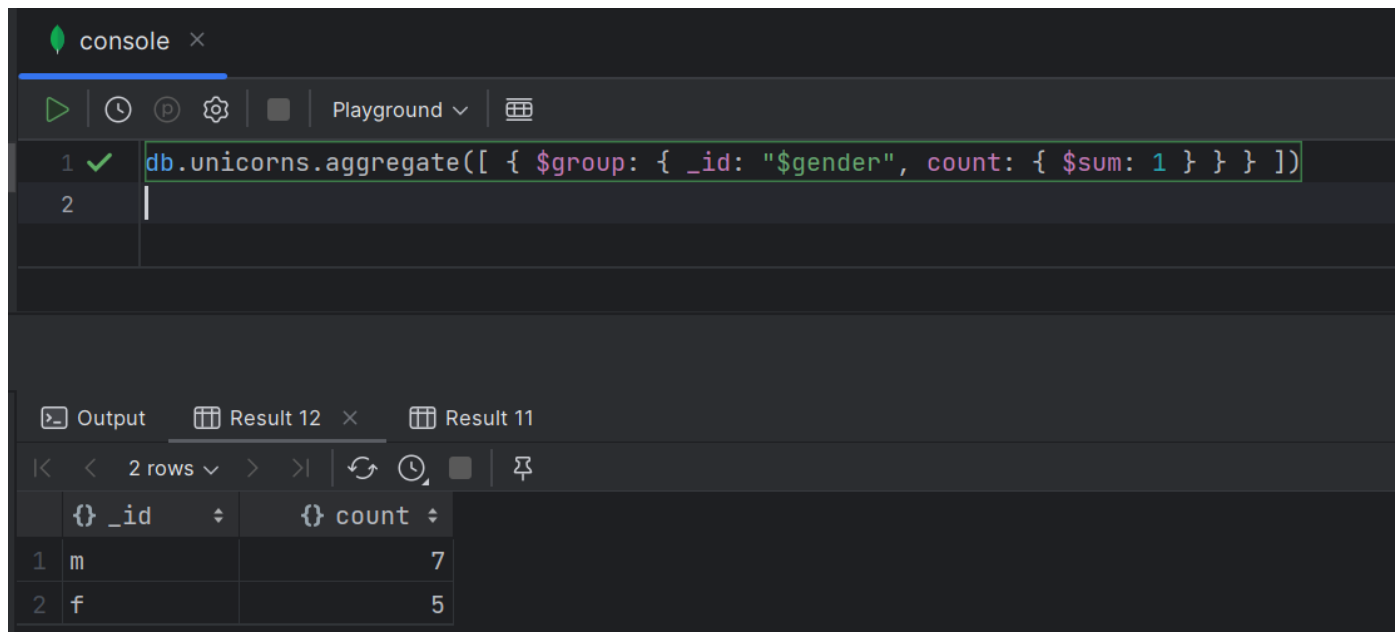
```
1 ✓ db.unicorns.distinct("loves")
2
```

Output

	result
1	apple
2	carrot
3	chocolate
4	energon
5	grape
6	lemon
7	papaya
8	redbull
9	strawberry
10	sugar
11	watermelon

### Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.



The screenshot shows a MongoDB Playground interface. In the console, the following aggregate query is entered and executed:

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

The result is displayed in a table with 2 rows:

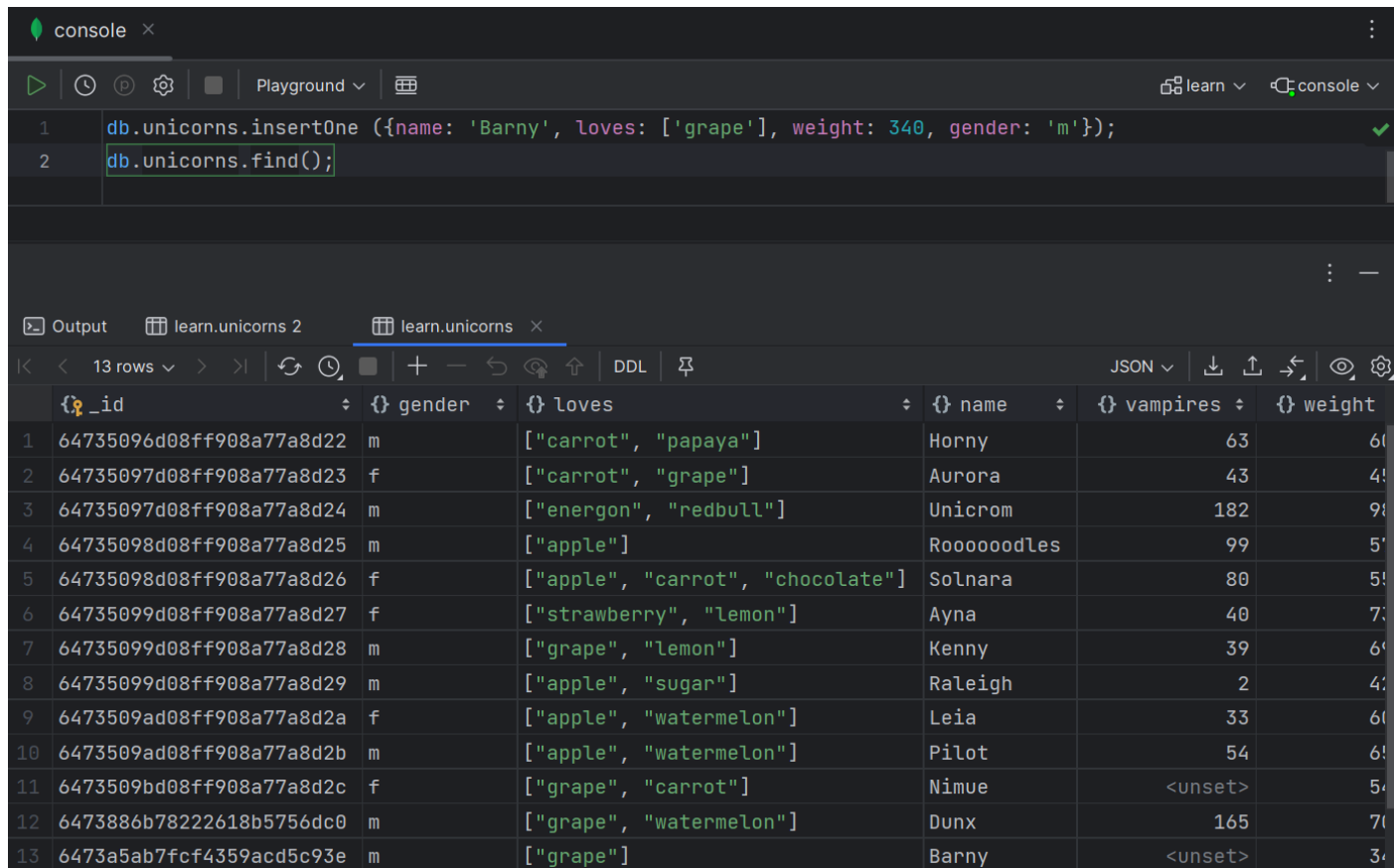
	_id	count
1	m	7
2	f	5

### Практическое задание 8.2.6:

1. Выполнить команду:

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

2. Проверить содержимое коллекции `unicorns`.



The screenshot shows a MongoDB Playground interface. In the console, the following insertOne command is entered and executed:

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

The result is displayed in a table with 13 rows, showing the contents of the `unicorns` collection:

	_id	gender	loves	name	vampires	weight
1	64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	63	60
2	64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	40
3	64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	182	90
4	64735098d08ff908a77a8d25	m	["apple"]	Rooodles	99	50
5	64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	50
6	64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	40	70
7	64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	39	60
8	64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	2	40
9	6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	60
10	6473509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	54	60
11	6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	50
12	6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	165	70
13	6473a5ab7fcf4359acd5c93e	m	["grape"]	Barney	<unset>	340

### Практическое задание 8.2.7:

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

The screenshot shows a MongoDB Playground interface. The console has two lines of code: `db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight: 800, vampires: 51}});` and `db.unicorns.find();`. The output displays a table of 13 unicorns. The unicorn named 'Ayna' (id: 735099d08ff908a77a8d27) now has a weight of 800 and 51 vampires. The table includes columns for \_id, gender, loves, name, vampires, and weight.

	_id	gender	loves	name	vampires	weight
1	735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	63	600
2	735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	450
3	735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	182	984
4	735098d08ff908a77a8d25	m	["apple"]	Rooodoodles	99	575
5	735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	51	800
7	735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	39	690
8	735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	2	421
9	73509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	601
10	73509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	54	650
11	73509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	540
12	73886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	165	704
13	73a5ab7fc4359acd5c93e	m	["grape"]	Barney	<unset>	310

### Практическое задание 8.2.8:

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

The screenshot shows a MongoDB Playground interface. The console has two lines of code: `db.unicorns.updateOne({name: 'Raleigh'}, {$addToSet: {loves: 'redbull'}});` and `db.unicorns.find();`. The output displays a table of 13 unicorns. The unicorn named 'Raleigh' (id: 64735099d08ff908a77a8d29) now has 'redbull' added to his loves array. The table includes columns for \_id, gender, loves, name, vampires, and weight.

	_id	gender	loves	name	vampires	weight
3	64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	182	984
4	64735098d08ff908a77a8d25	m	["apple"]	Rooodoodles	99	575
5	64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	51	800
7	64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	39	690
8	64735099d08ff908a77a8d29	m	["apple", "sugar", "redbull"]	Raleigh	2	421
9	6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	601
10	6473509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	54	650
11	6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	540
12	6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	165	704
13	6473a5ab7fc4359acd5c93e	m	["grape"]	Barney	<unset>	310

### Практическое задание 8.2.9:

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

The screenshot shows a MongoDB Playground interface. The console at the top contains two commands:

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

Below the console, the document viewer displays 13 rows of data from the `unicorns` collection. Each row represents a unicorn document with fields: `_id`, `gender`, `loves`, `name`, `vampires`, and `weight`.

	<code>_id</code>	<code>gender</code>	<code>loves</code>	<code>name</code>	<code>vampires</code>	<code>weight</code>
1	64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	78	60
2	64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	40
3	64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	197	90
4	64735098d08ff908a77a8d25	m	["apple"]	Roooooodles	114	50
5	64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	50
6	64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	51	80
7	64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	54	60
8	64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	17	40
9	6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	60
10	6473509ad08ff908a77a8d2b	m	["apple", "watermelon"]	Pilot	69	60
11	6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	50
12	6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	180	70
13	6473a5ab7fcf4359acd5c93e	m	["grape"]	Barney	15	30

### Практическое задание 8.2.10:

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

The screenshot shows a MongoDB Playground interface. The console at the top contains two commands:

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

Below the console, the document viewer displays 3 rows of data from the `towns` collection. Each row represents a town document with fields: `_id`, `famous_for`, `last_sensus`, and `mayor`.

	<code>_id</code>	<code>famous_for</code>	<code>last_sensus</code>	<code>mayor</code>
1	64739b7559be0f16f42d0892	[""]	2008-01-31T00:00:00.000Z	{"name": "Jim Wehrle"}
2	64739b7559be0f16f42d0893	["status of liberty", "food"]	2009-07-31T00:00:00.000Z	{"name": "Michael Bloomberg",
3	64739b7559be0f16f42d0894	["beer", "food"]	2009-07-20T00:00:00.000Z	{"name": "Sam Adams", "party"

### Практическое задание 8.2.11:

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

The screenshot shows the MongoDB Playground interface. The console has two commands executed successfully:

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

The output displays 13 rows of the `unicorns` collection. The `Pilot` entry now includes "chocolate" in its `loves` array.

<code>_id</code>	<code>gender</code>	<code>loves</code>	<code>name</code>	<code>vampires</code>	<code>weight</code>
64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	78	
64735097d08ff908a77a8d23	f	["carrot", "grape"]	Aurora	43	
64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	197	
64735098d08ff908a77a8d25	m	["apple"]	Rooooooodles	114	
64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	
64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	51	
64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	54	
64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	17	
6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	
6473509ad08ff908a77a8d2b	m	["apple", "watermelon", "chocolate"]	Pilot	69	
6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	
6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	180	
6473a5ab7fcf4359acd5c93e	m	["grape"]	Barney	15	

### 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*.

The screenshot shows the MongoDB Playground interface. The console has two commands executed successfully:

```
1 db.unicorns.updateOne ({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}});
2 db.unicorns.find();
```

The output displays 13 rows of the `unicorns` collection. The `Aurora` entry now includes "sugar" and "lemon" in its `loves` array.

<code>_id</code>	<code>gender</code>	<code>loves</code>	<code>name</code>	<code>vampires</code>	<code>weight</code>
64735096d08ff908a77a8d22	m	["carrot", "papaya"]	Horny	78	
64735097d08ff908a77a8d23	f	["carrot", "grape", "sugar", "lemon"]	Aurora	43	
64735097d08ff908a77a8d24	m	["energon", "redbull"]	Unicrom	197	
64735098d08ff908a77a8d25	m	["apple"]	Rooooooodles	114	
64735098d08ff908a77a8d26	f	["apple", "carrot", "chocolate"]	Solnara	80	
64735099d08ff908a77a8d27	f	["strawberry", "lemon"]	Ayna	51	
64735099d08ff908a77a8d28	m	["grape", "lemon"]	Kenny	54	
64735099d08ff908a77a8d29	m	["apple", "sugar"]	Raleigh	17	
6473509ad08ff908a77a8d2a	f	["apple", "watermelon"]	Leia	33	
6473509ad08ff908a77a8d2b	m	["apple", "watermelon", "chocolate"]	Pilot	69	
6473509bd08ff908a77a8d2c	f	["grape", "carrot"]	Nimue	<unset>	
6473886b78222618b5756dc0	m	["grape", "watermelon"]	Dunx	180	
6473a5ab7fcf4359acd5c93e	m	["grape"]	Barney	15	

## Практическое задание 8.2.13:

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"}}
```

2. Удалите документы с беспартийными мэрами.

3. Проверьте содержание коллекции.

console ×

1 ✓ `db.towns.deleteMany({'mayor.party': {'$exists': false}});`

2

3 ✓ `db.towns.find()`

Output Result 33 learn.towns ×

	{_id}	{famous_for}	{last_sensus}	{mayor}
1	6473aa537fcf4359acd5c947	["status of liberty", "food"]	2009-07-31T00:00:00.000Z	{"name": "Michael Bloomberg",
2	6473aa537fcf4359acd5c948	["beer", "food"]	2009-07-20T00:00:00.000Z	{"name": "Sam Adams", "party":

4. Очистите коллекцию.

5. Просмотрите список доступных коллекций.

console ×

1 ✓ `db.towns.deleteMany({});`

2

3 ✓ `show collections;`

Output Result 35-2 × Result 35

	{badge}	{name}
1		towns
2		unicorns



## 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?

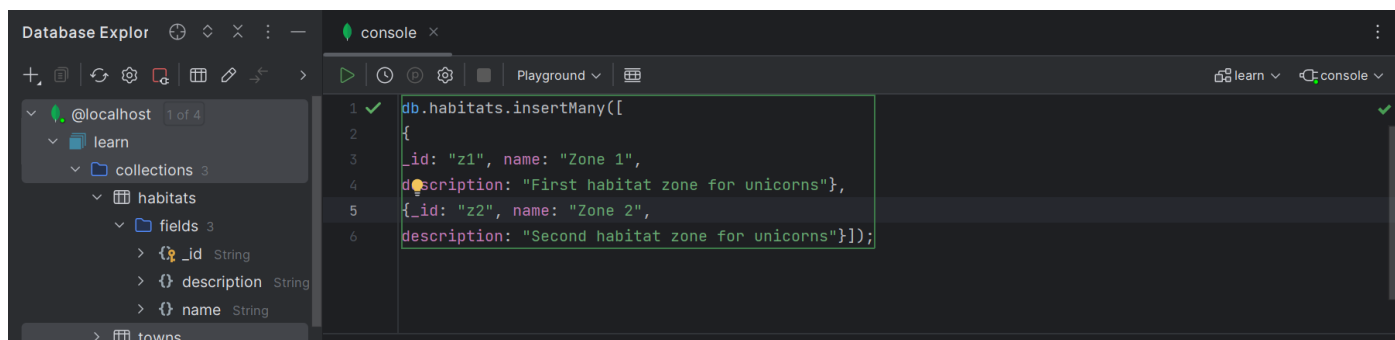
Функция `update(One)` позволяет более детально настроить редактирование документов, так как она предоставляет больше операторов и возможностей, включая `$set`, `$unset`, `$inc`, `$push`, `$addToSet` и другие.

### 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.

console x

Playground

```

1 db.unicorns.updateMany({}, { $set: { habitat: "z1" }});
2 db.unicorns.find()

```

Output Result 35-2 learn.unicorns x

13 rows

	_id	gender	habitat	loves	name	var
1	64735096d08ff908a77a8d22	m	z1	["carrot", "papaya"]	Horny	
2	64735097d08ff908a77a8d23	f	z1	["carrot", "grape", "sugar", "lemon"]	Aurora	
3	64735097d08ff908a77a8d24	m	z1	["energon", "redbull"]	Unicrom	
4	64735098d08ff908a77a8d25	m	z1	["apple"]	Roooooodles	
5	64735098d08ff908a77a8d26	f	z1	["apple", "carrot", "chocolate"]	Solnara	
6	64735099d08ff908a77a8d27	f	z1	["strawberry", "lemon"]	Ayna	
7	64735099d08ff908a77a8d28	m	z1	["grape", "lemon"]	Kenny	
8	64735099d08ff908a77a8d29	m	z1	["apple", "sugar"]	Raleigh	
9	6473509ad08ff908a77a8d2a	f	z1	["apple", "watermelon"]	Leia	
10	6473509ad08ff908a77a8d2b	m	z1	["apple", "watermelon", "chocolate"]	Pilot	
11	6473509bd08ff908a77a8d2c	f	z1	["grape", "carrot"]	Nimue	
12	6473886b78222618b5756dc0	m	z1	["grape", "watermelon"]	Dunx	
13	6473a5ab7fcf4359acd5c93e	m	z1	["grape"]	Barney	

#### 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}
```

### Практическое задание 8.3.2:

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

The screenshot shows the MongoDB Playground interface. In the console, the command `db.unicorns.createIndex ({ name: 1 }, { unique: true });` has been executed successfully. Below the console, the 'Output' tab shows the result of the command, which is a single row with the column `name_1`.

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.

The screenshot shows the MongoDB Playground interface. In the console, the command `db.unicorns.getIndexes();` has been executed successfully. Below the console, the 'Output' tab shows the result of the command, which is a single row with the columns `key`, `name`, and `v`. The `key` column contains the object `{ "_id": new NumberInt("1") }`, the `name` column contains `_id_`, and the `v` column contains `2`.

3. Попробуйте удалить индекс для идентификатора.

The screenshot shows the MongoDB Playground interface. In the console, the command `db.unicorns.dropIndex("_id_");` has been executed, but it failed. Below the console, the 'Output' tab shows the error message: "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.

The screenshot shows the MongoDB Playground interface. The console contains the following code:

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

The output tab shows the result of the query, displaying 4 rows of data:

	_id	value
1	6473af6b7fcf4359acd75030	99999
2	6473af6b7fcf4359acd7502f	99998
3	6473af6b7fcf4359acd7502e	99997
4	6473af6b7fcf4359acd7502d	99996

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

202 ms

5. Создайте индекс для ключа `value`.

The screenshot shows the MongoDB Playground interface. The console contains the following code:

```
1 db.numbers.createIndex ({value: 1});  
2
```

The output tab shows the result of the command, displaying 1 row of data:

	result
1	value_1

6. Получите информацию обо всех индексах коллекции `numbers`.

console x

Playground

```
1 db.numbers.getIndexes();
```

2

Output Result 53

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

7. Выполните запрос 2.

console x

Playground

```
1 db.numbers.find().sort({_id: -1}).limit(4);
```

2

Output learn.numbers

	_id	value
1	6473af6b7fcf4359acd75030	99999
2	6473af6b7fcf4359acd7502f	99998
3	6473af6b7fcf4359acd7502e	99997
4	6473af6b7fcf4359acd7502d	99996

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

169 ms

9. 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, чтобы найти и выбрать нужные документы. Это требует больше времени и ресурсов, особенно при большом объеме данных.

**Время** выполнения запроса с индексом оказалось меньше, чем время запроса без индекса, с учетом того, что запрос был один и тот же и выполнялся на одной и той же коллекции. Таким образом, можно утверждать, что **запрос с индексом более эффективен** по времени. Тем не менее, необходимо помнить об ограничениях использования индексов: их можно установить ограниченное количество, каждый из них использует оперативную память, и т.д.

### **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:**

При выполнении данной лабораторной работы были приобретены практические навыки по работе с MongoDB, а именно овладение 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. Таким образом, был получен опыт по работе и использованию MongoDB.