

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

**ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6**

по теме:

«Работа с БД в СУБД *MongoDB*»

по дисциплине: Проектирование и реализация баз данных

Специальность:

09.03.03 Мобильные и сетевые технологии

Проверила:

Говорова М.М.

Дата: «..» ... 2023 г.

Оценка _____

Выполнил:

студент группы К3239

Прокопец С. Р.

Санкт-Петербург 2023

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

Выполнение:

Практическая работа 6.2

Практическое задание 2.1.1:

1. Создайте базу данных learn.
2. Заполните коллекцию единорогов unicorns:

```
db.unicorns.insertOne({name: 'Horny', loves: ['carrot','papaya'],  
weight: 600, gender: 'm', vampires: 63});  
db.unicorns.insertOne({name: 'Aurora', loves: ['carrot', 'grape'],  
weight: 450, gender: 'f', vampires: 43});  
db.unicorns.insertOne({name: 'Unicrom', loves: ['energon',  
'redbull'], weight: 984, gender: 'm', vampires: 182});  
db.unicorns.insertOne({name: 'Roooooodles', loves: ['apple'],  
weight: 575, gender: 'm', vampires: 99});  
db.unicorns.insertOne({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:'Kennny', 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'});
```

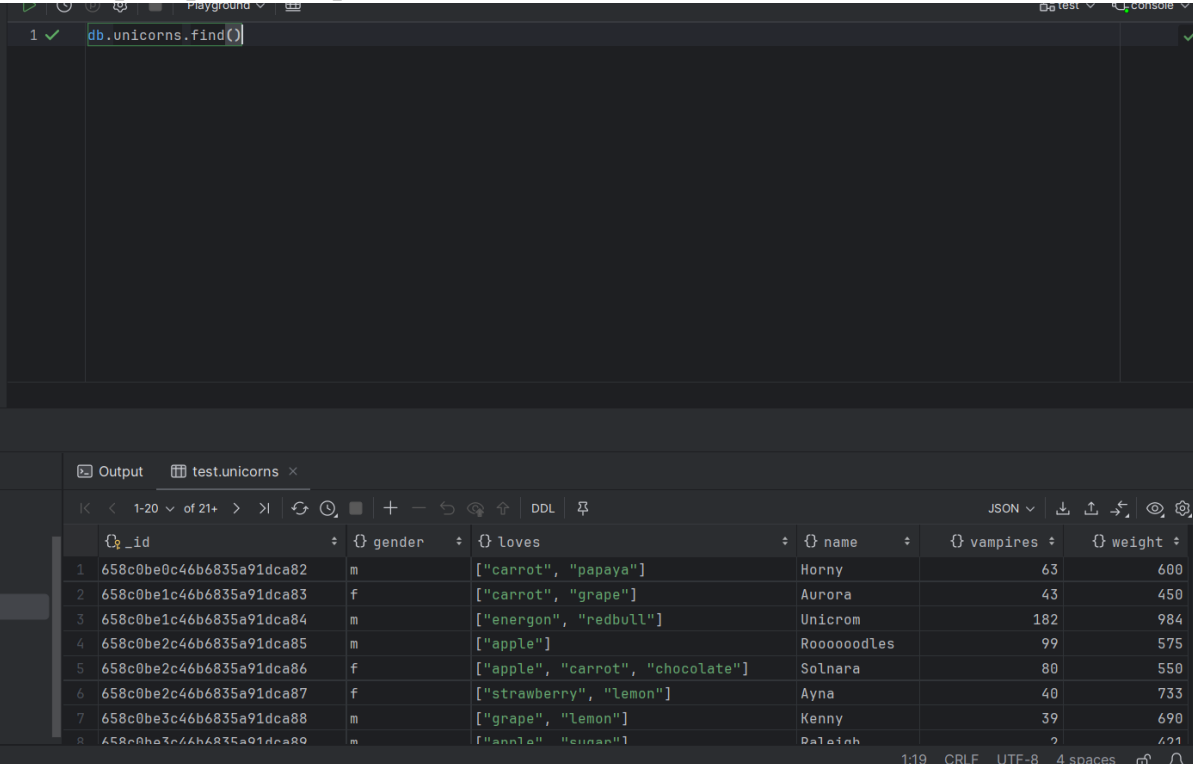
```
Playground test console
1 ✓ db.unicorns.insertOne({name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63});
2 ✓ db.unicorns.insertOne({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
3 ✓ db.unicorns.insertOne({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
4 ✓ db.unicorns.insertOne({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
5 ✓ db.unicorns.insertOne({name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f', vampires: 80});
6 ✓ db.unicorns.insertOne({name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
7 ✓ db.unicorns.insertOne({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
8 ✓ db.unicorns.insertOne({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
9 ✓ db.unicorns.insertOne({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
10 ✓ db.unicorns.insertOne({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
11 ✓ db.unicorns.insertOne({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});

test> db.unicorns.insertOne({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
[2023-12-27 14:44:26] completed in 420 ms
test> db.unicorns.insertOne({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
[2023-12-27 14:44:26] completed in 424 ms
test> db.unicorns.insertOne({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
[2023-12-27 14:44:27] completed in 485 ms
test> db.unicorns.insertOne({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
[2023-12-27 14:44:27] completed in 432 ms
test> db.unicorns.insertOne({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
[2023-12-27 14:44:28] completed in 436 ms
```

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.



The screenshot shows a MongoDB Playground interface. In the top console, the command `db.unicorns.find()` has been executed successfully, indicated by a green checkmark. Below the console, the output is displayed in a table format. The table has 8 columns: `_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	658c0be0c46b6835a91dca82	m	["carrot", "papaya"]	Horny	63	600
2	658c0be1c46b6835a91dca83	f	["carrot", "grape"]	Aurora	43	450
3	658c0be1c46b6835a91dca84	m	["energon", "redbull"]	Unicrom	182	984
4	658c0be2c46b6835a91dca85	m	["apple"]	Rooooooodles	99	575
5	658c0be2c46b6835a91dca86	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	658c0be2c46b6835a91dca87	f	["strawberry", "lemon"]	Ayna	40	733
7	658c0be3c46b6835a91dca88	m	["grape", "lemon"]	Kenny	39	690
8	658c0be3c46b6835a91dca89	m	["apple", "euphor"]	Daleinh	2	421

At the bottom right of the output panel, the following settings are visible: 1:19, CRLF, UTF-8, 4 spaces, and a bell icon.

2.2.1:

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

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

The screenshot shows a database playground interface. At the top, there's a console window with a query: `db.unicorns.find({gender: 'm'}, {_id: 0, name: 1}).sort({name: 1})`. Below the console, there's a 'Database Sessions' panel on the left and an 'Output' panel on the right. The 'Database Sessions' panel shows a tree view of database sessions, including 'learn', 'unicorns', 'console', and 'postgres@localhost'. The 'Output' panel shows the results of the query, which are 8 rows of unicorn names sorted alphabetically.

name
1 Dunx
2 Horny
3 Kenny
4 Pilot
5 Pilot
6 Raleigh
7 Roopoooodles
8 Unicrom

1 ✓

db.unicorns.findOne({gender: 'f'}, {_id: 0, name: 1, loves: 'corrot'})

vices

⏏ × 👁 📄 +

Database Sessions

Database

learn

unicorns 317 ms

Output Result 18 ×

1 row

🔄 ⌚ ⏏

loves name

1 corrot Aurora

1 ✓

db.unicorns.find({gender: 'f'}, {_id: 0, name: 1, loves: 'corrot'}).limit(1)

Output test.unicorns ×

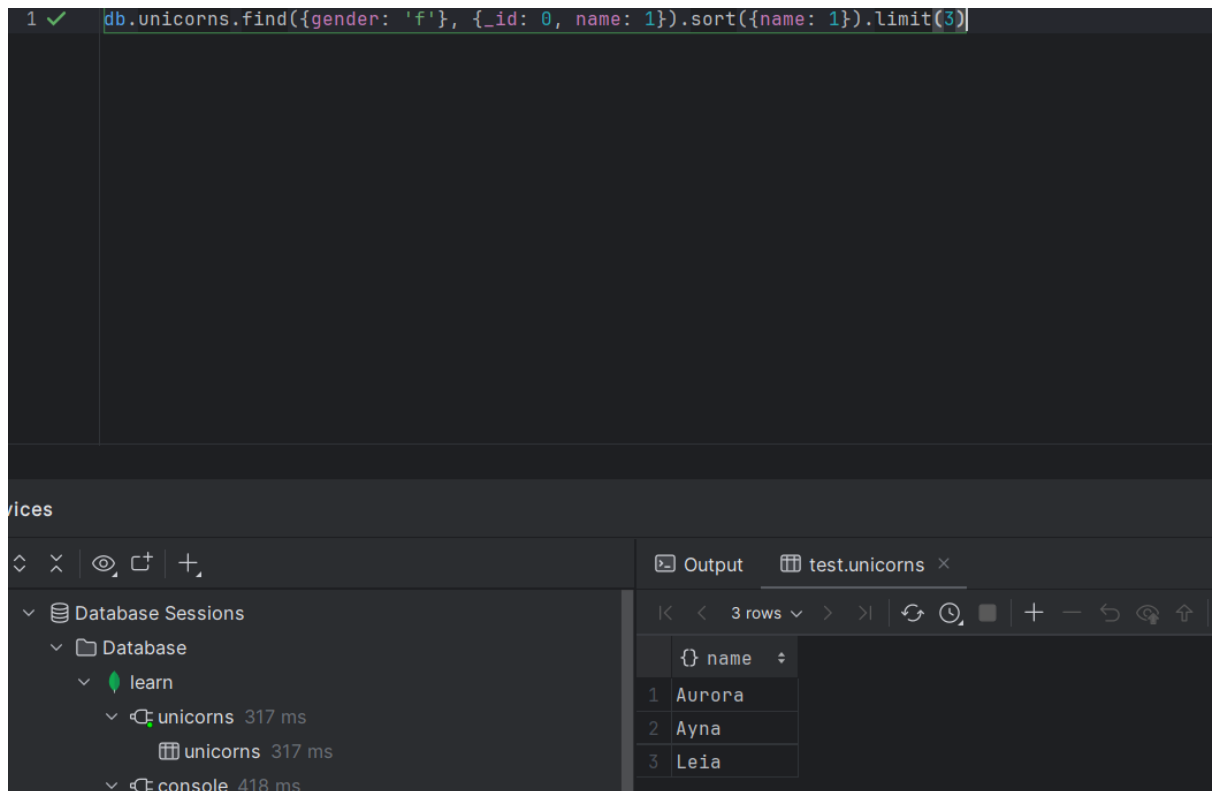
1 row

🔄 ⌚ ⏏ + - ↶ 🔍 ⬆

DDL ⏏

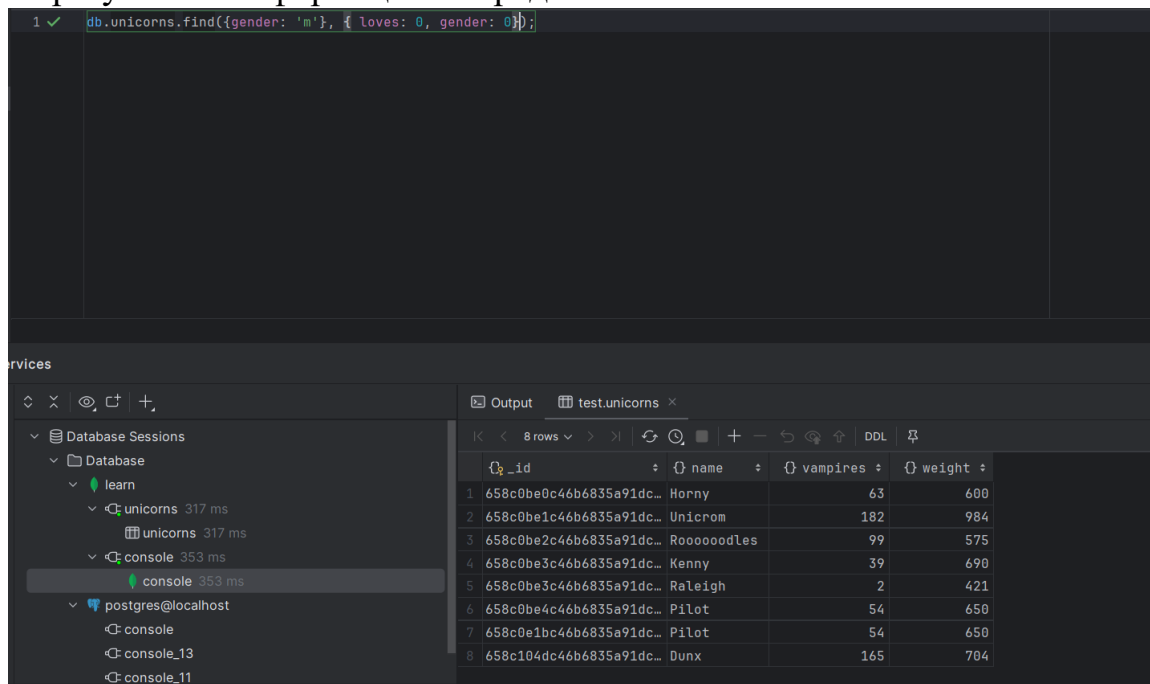
loves name

1 corrot Aurora



Практическое задание 2.2.2:

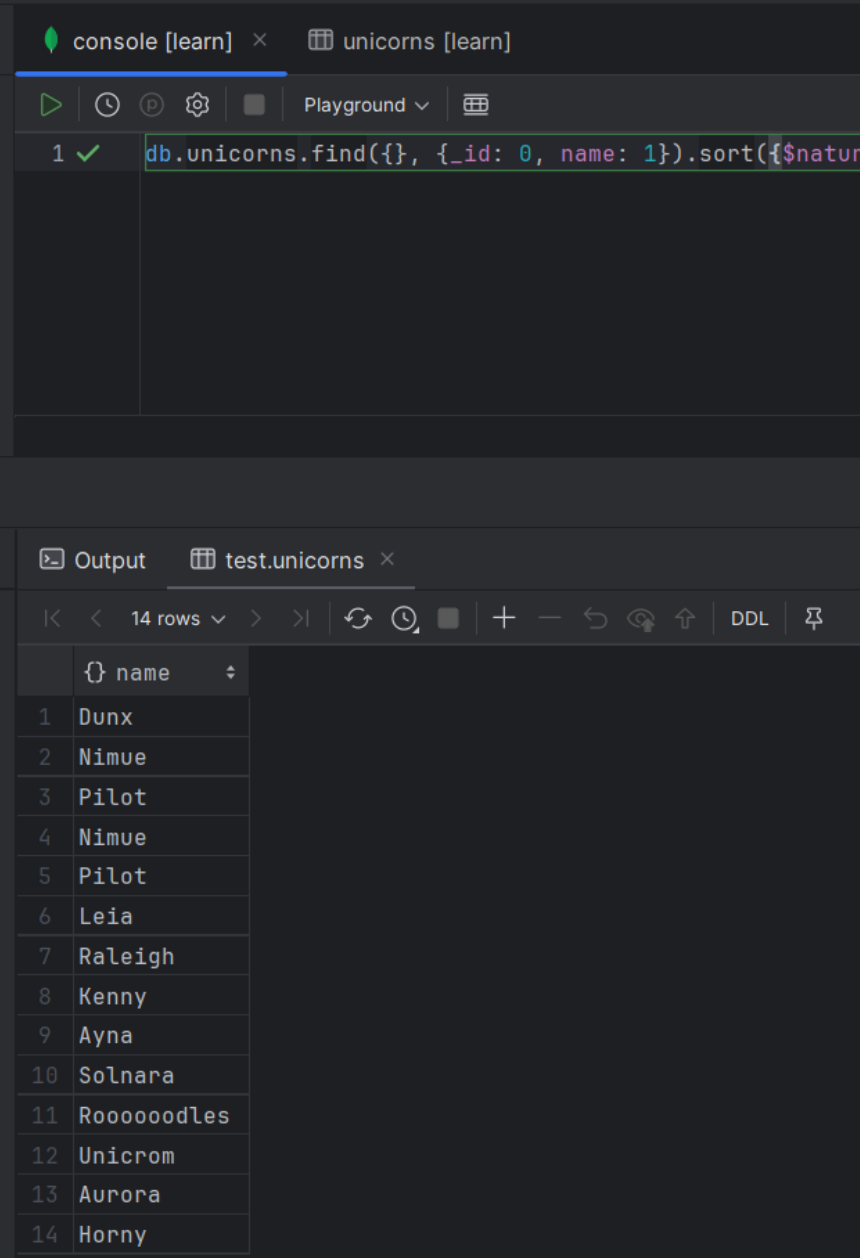
Модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпочтениях и поле.



Практическое задание 2.2.3:

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

Вывести список единорогов в обратном порядке добавления.



The screenshot shows a MongoDB Playground interface. At the top, there are tabs for 'console [learn]' and 'unicorns [learn]'. Below the tabs is a toolbar with icons for running, saving, and other actions. The main area contains a code editor with the following query:

```
1 db.unicorns.find({}, {_id: 0, name: 1}).sort({$natural: -1})
```

Below the code editor is an 'Output' tab, which is currently selected. It shows a table with 14 rows of data. The table has a column 'name' and a column for row numbers. The data is as follows:

	name
1	Dunx
2	Nimue
3	Pilot
4	Nimue
5	Pilot
6	Leia
7	Raleigh
8	Kenny
9	Ayna
10	Solnara
11	Rooodoodles
12	Unicrom
13	Aurora
14	Horny

Практическое задание 2.2.4:
Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор.

Playground

test

1 ✓

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

vices

>

Output

test.unicorns

14 rows

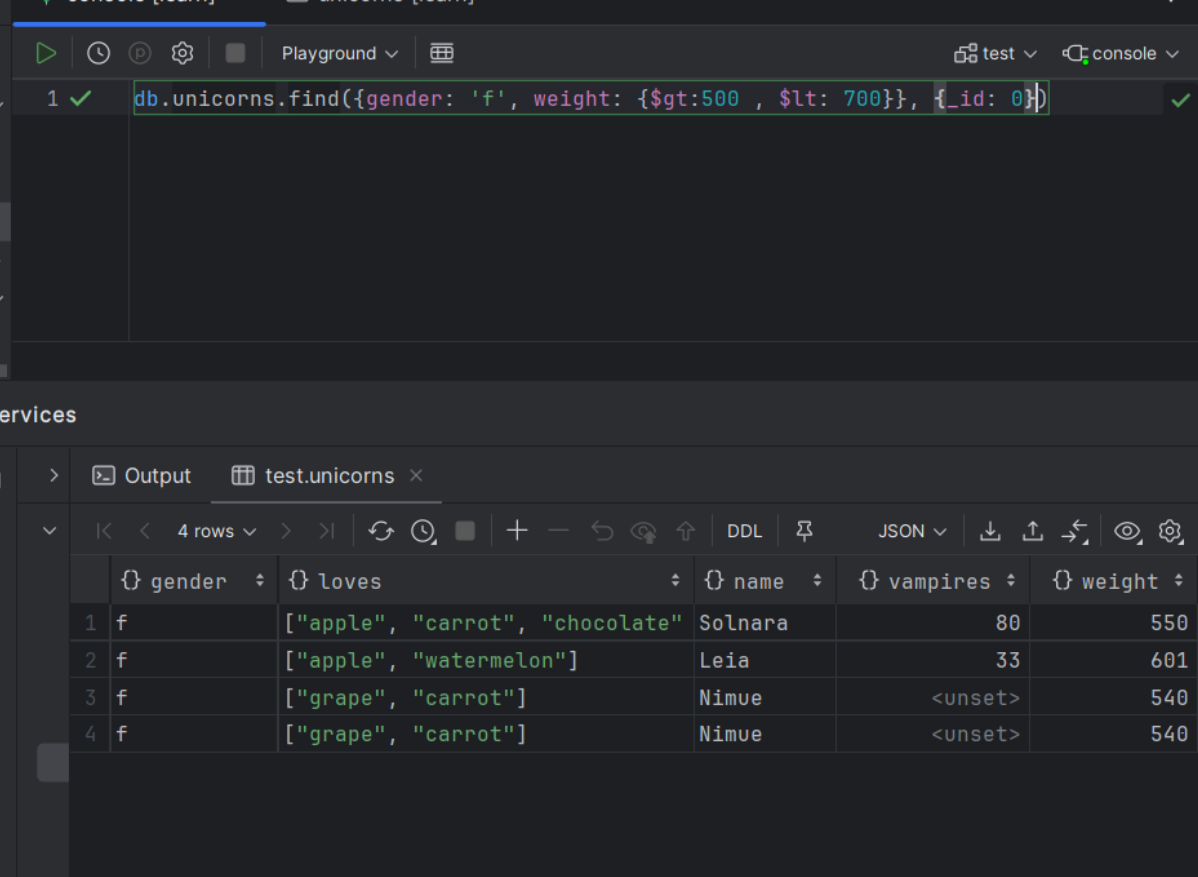
DDL

JSON

	gender	loves	name	vampires	weight
1	m	["carrot"]	Horny	63	600
2	f	["carrot"]	Aurora	43	450
3	m	["energon"]	Unicrom	182	984
4	m	["apple"]	Roooooodles	99	575
5	f	["apple"]	Solnara	80	550
6	f	["strawberry"]	Ayna	40	733
7	m	["grape"]	Kenny	39	690
8	m	["apple"]	Raleigh	2	421
9	f	["apple"]	Leia	33	601
10	m	["apple"]	Pilot	54	650
11	f	["grape"]	Nimue	<unset>	540
12	m	["apple"]	Pilot	54	650
13	f	["grape"]	Nimue	<unset>	540
14	m	["grape"]	Dunx	165	704

Практическое задание 2.3.1:

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



The screenshot shows a MongoDB Playground interface. At the top, a query is entered in the editor: `db.unicorns.find({gender: 'f', weight: {$gt: 500, $lt: 700}}, {_id: 0})`. Below the editor, the results are displayed in a table format. The table has five columns: `gender`, `loves`, `name`, `vampires`, and `weight`. There are four rows of data, all with `gender` 'f'. The `weight` values are 550, 601, 540, and 540. The `loves` column contains arrays of strings. The `vampires` column contains the value 80 for the first row, 33 for the second, and <unset> for the last two. The `name` column contains 'Solnara', 'Leia', and 'Nimue' for the first three rows, and 'Nimue' for the last.

	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
4	f	["grape", "carrot"]	Nimue	<unset>	540

Практическое задание 2.3.2:

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

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

vices

Output test.unicorns x

1 row v

	gender	loves	name	vampires	weight
1	m	["grape", "lemon"]	Kenny	39	690

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ vampires

```
1 ✓ db.unicorns.find({vampires: {$exists: 0}}) ✓
```

vices

Output test.unicorns x

1 row v

	_id	gender	loves	name	weight
1	658c0be5c46b6835a91dca8c	f	["grape", "carrot"]	Nimue	540

Практическое задание 2.3.4:

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

The screenshot shows a MongoDB Playground interface. At the top, a query is entered in the editor: `unicorns.find({gender: 'm'}, {_id: 0, name: 1, loves: {$slice: 1}}).sort({name: 1})`. Below the editor, the results are displayed in a table format. The table has two columns: 'loves' and 'name'. The results are sorted by name in ascending order.

	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 3.1.1:

1. Создайте коллекцию towns, включающую следующие документы:

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

```

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

```

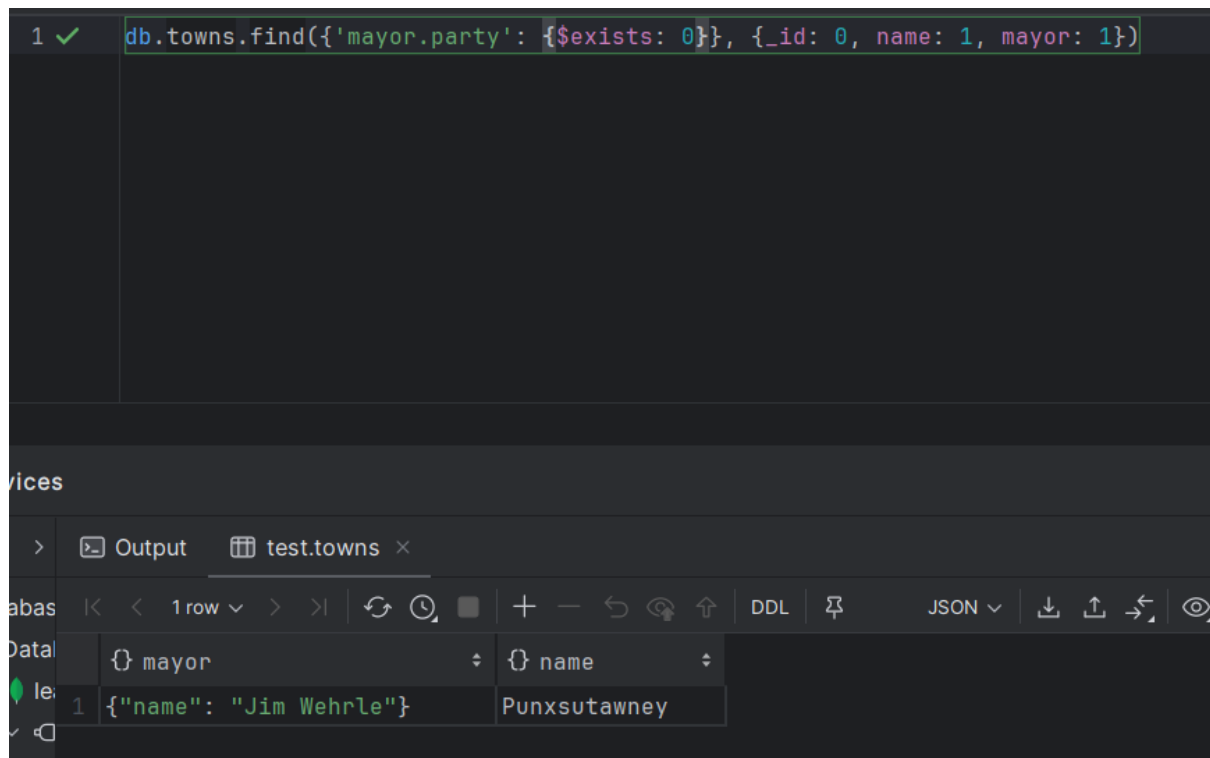
1. Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре.
2. Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре

The screenshot shows the MongoDB Playground interface. At the top, a query is entered in the editor:

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

Below the editor, the results are displayed in a table. The table has two columns: 'mayor' and 'name'. The first row shows the result for Michael Bloomberg in New York.

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



Практическое задание 3.1.2:

1. Сформировать функцию для вывода списка самцов единорогов.

The screenshot shows a code editor with the following code:

```
1 ✓ function males(){
2   db.unicorns.find({gender: 'm'}).forEach(function(x){ print(x.name)})
3   };
4
5 ✓ males();
```

Below the code editor, there is a console window titled "Output" showing the results of the function call. The console displays a list of 7 rows, which are the names of the unicorns found:

1	Horny
2	Unicrom
3	Rooooooodles
4	Kenny
5	Raleigh
6	Pilot
7	Dunx

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

3. Вывести результат, используя forEach.

```
var curs = db.unicorns.find({gender: 'm'}).limit(2).sort({name: 1});
curs.forEach(print)
```

```
1 ✓ var curs = db.unicorns.find({gender: 'm'}).limit(2).sort({name: 1});
2
3 ✓ curs.forEach(print)
```

Services

Database Sessions

Database

learn

unicorns

Result 42-2

1	{ "_id": new ObjectId("658c104dc46b6835a91dca9c"), "name": "Dunx", "loves": ["grape"] }
2	{ "_id": new ObjectId("658c0be0c46b6835a91dca82"), "name": "Horny", "loves": ["carro"] }

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Services

Database Sessions

Database

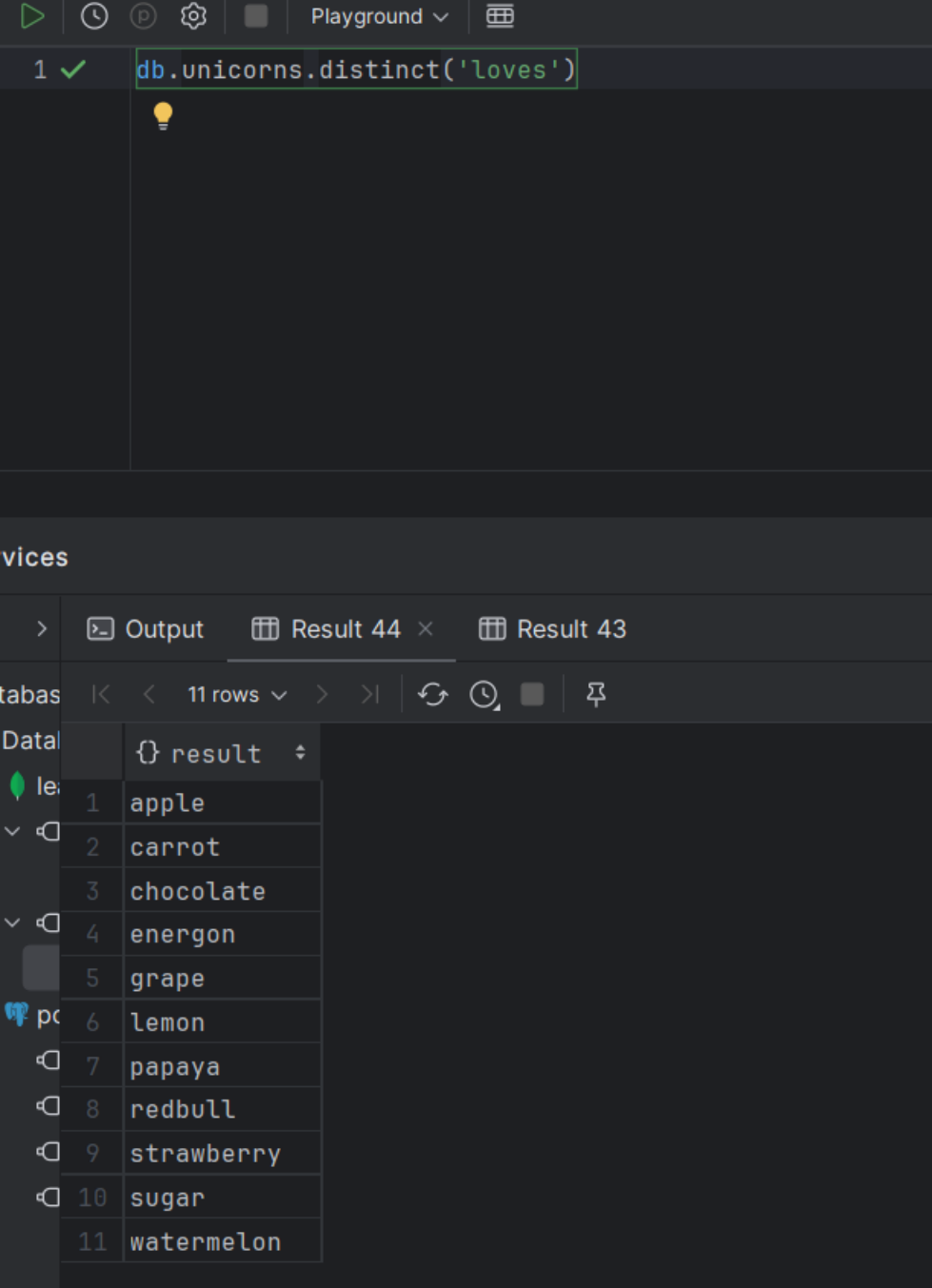
learn

unicorns

Result 43

1	2

Практическое задание 3.2.2:
Вывести список предпочтений.



The screenshot shows a Jupyter Notebook interface with a 'Playground' tab. A SQL query is entered in the code cell:

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

Below the code cell, the results are displayed in a table format. The table has 11 rows and 1 column, showing the distinct values of the 'loves' field for unicorns. The results are:

	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 3.2.3:
Подсчитать количество особей единорогов обоих полов.

The screenshot shows a MongoDB Playground interface. At the top, a query is entered in the editor: `db.unicorns.aggregate({$group: {_id: '$gender', count: {$sum: 1}}})`. Below the editor, the results are displayed in a table format. The table has two columns: `_id` and `count`. There are two rows of data: one for 'm' with a count of 7, and one for 'f' with a count of 5. The interface also shows tabs for 'Output', 'Result 44', and 'Result 46'. The 'Output' tab is selected, showing the table of results.

	_id	count
1	m	7
2	f	5

Практическое задание 3.3.1:

1. Выполнить команду:

```
1 db.unicorns.save({name: 'Barney', loves: ['grape'], weight: 340,  
2 gender: 'm'})  
3
```

Метод устарел((((((((((((((((((((

2. Проверить содержимое коллекции unicorns.

1 ✓ `db.unicorns.find()` ✓

2

services

> Output test.unicorns x Result 46

tabas 12 rows

	_id	gender	loves	name
1	658c0be0c46b6835a91dca82	m	["carrot", "papaya"]	Horny
2	658c0be1c46b6835a91dca83	f	["carrot", "grape"]	Aurora
3	658c0be1c46b6835a91dca84	m	["energion", "redbull"]	Unicrom
4	658c0be2c46b6835a91dca85	m	["apple"]	Roooooodles
5	658c0be2c46b6835a91dca86	f	["apple", "carrot", "chocolate"]	Solnara
6	658c0be2c46b6835a91dca87	f	["strawberry", "lemon"]	Ayna
7	658c0be3c46b6835a91dca88	m	["grape", "lemon"]	Kenny
8	658c0be3c46b6835a91dca89	m	["apple", "sugar"]	Raleigh
9	658c0be4c46b6835a91dca8a	f	["apple", "watermelon"]	Leia
10	658c0be4c46b6835a91dca8b	m	["apple", "watermelon"]	Pilot
11	658c0be5c46b6835a91dca8c	f	["grape", "carrot"]	Nimue
12	658c104dc46b6835a91dca9c	m	["grape", "watermelon"]	Dunx

Практическое задание 3.3.2:

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

The screenshot shows a MongoDB Playground interface. The top section contains two lines of JavaScript code executed in the console:

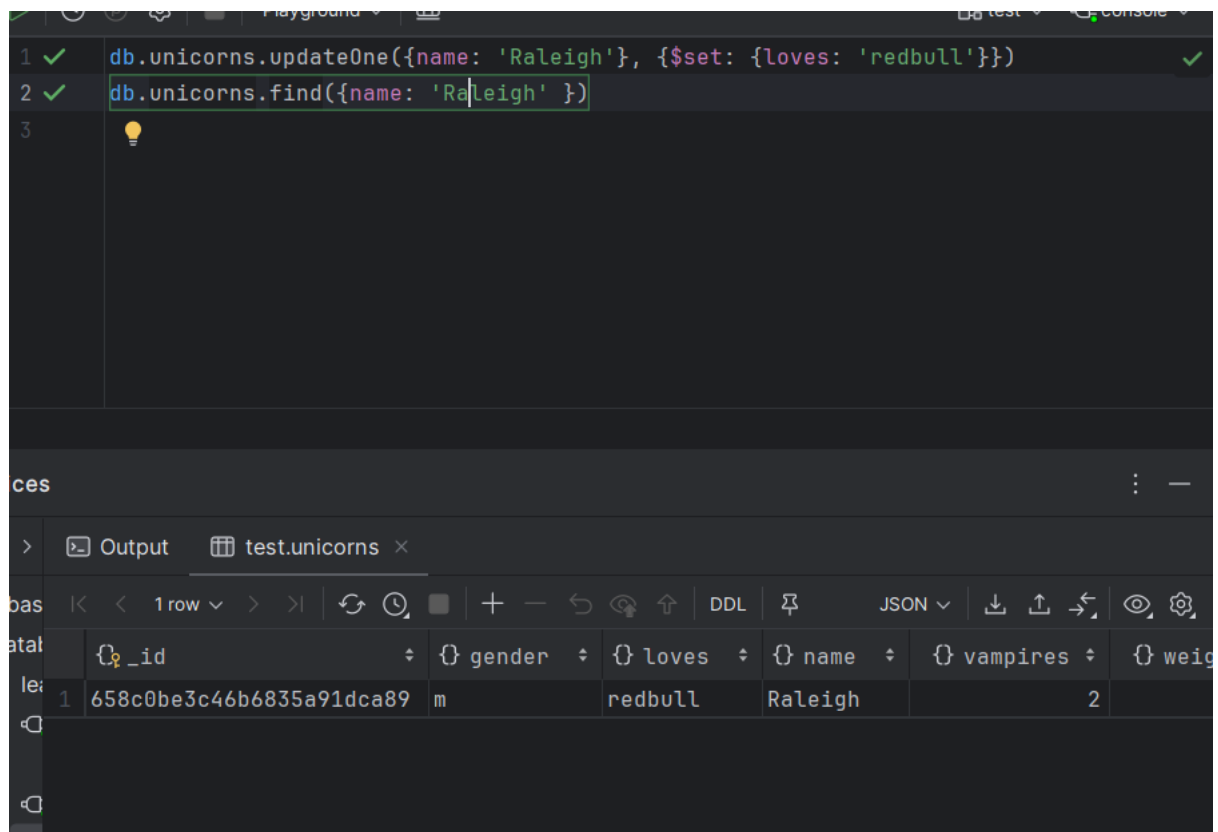
```
1 ✓ db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight: 800, vampires: 51}}) ✓  
2 ✓ db.unicorns.find({name: 'Ayna' })
```

Below the console, the 'test.unicorns 2' tab is active, displaying a table of data. The table has columns for 'gender', 'loves', 'name', 'vampires', and 'weight'. A single row is visible, representing the unicorn 'Ayna'.

	gender	loves	name	vampires	weight
1	f	["strawberry", "lemon"]	Ayna	51	800

Практическое задание 3.3.3:

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



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

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

```
1 ✓ db.unicorns.updateOne({gender: 'm'}, {$inc: {vampires: 5}})
2 ✓ db.unicorns.find({gender: 'm'}, {_id: 0, name: 1, vampires: 1})
3
```

Below the console, the 'Output' tab is selected, showing a table of data for the 'test.unicorns' collection. The table has two columns: 'name' and 'vampires'.

	name	vampires
1	Horny	68
2	Unicrom	182
3	Rooooooodles	99
4	Kenny	39
5	Raleigh	2
6	Pilot	54
7	Dunx	165

Практическое задание 3.3.5:

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 has two lines of code, both marked with green checkmarks:

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

Below the console, the 'test.towns' tab is active, displaying a table with one row of data:

	_id	famous_for	last_sensus	mayor	name
1	658c2085c46b6835a91dcaa4	["beer", "food"]	2009-07-20T00:00:00.000Z	{"name": "Sam Adams", "party": "Portland"}	Portland

Практическое задание 3.3.6:

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

The screenshot shows a MongoDB Playground interface. The console has two lines of code, both marked with green checkmarks:

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

Below the console, the 'test.unicorns' tab is active, displaying a table with one row of data:

	_id	gender	loves	name
1	658c0be4c46b6835a91dca8b	m	["apple", "watermelon", "chocola"]	Pilot

Практическое задание 3.3.7:

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

The screenshot shows a 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({name: 'Aurora' })
```

Below the console, the 'Services' section shows a table view of the 'unicorns' collection. The table has columns: _id, gender, loves, name, vampires, and weight. One document is displayed for 'Aurora'.

	_id	gender	loves	name	vampires	weight
1	658c0be1c46b6835a91dca83	f	["carrot", "grape", "sugar", "lemon"]	Aurora	43	450

Практическое задание 3.4.1:

1. Создайте коллекцию towns, включающую следующие документы:

```
db.towns.insertMany([
  {
    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.
4. Очистите коллекцию.
5. Просмотрите список доступных коллекций.

1 ✓

db.towns.deleteMany({'mayor.party':{'\$exists: 0}})

Services

>

Output

Result 63

×

tabas

<

<

1 row

>

>

↺

⌚

■

📌

JSON

Data

{

acknowledged

}

{

deletedCount

}

le

1

• true

2

✓

📄

Database

console [learn] × towns [learn] unicorns [learn]

+

🔍

↺

🔄

📄

🔧

>

Playground

test test console

learn 1 of 4

test

collections 2

towns

unicorns

postgres@localhost 1

postgres 2 of 4

Server Objects

postgres@localhost (DDL)

1 ✓

db.towns.find({})

✓

Services

test.towns

base Set

<

<

2 rows

>

>

↺

⌚

■

+

-

↶

🔍

DDL

📌

JSON

📄

📌

🔍

🔧

database

learn

famous_for

last_sensus

mayor

name

popujatiuon

1

644a71e1411

["status of liberty", "food"]

2009-07-31T00:00:00.000Z

{"name": "Michael Bloomberg", "party": "New York"

22200000

2

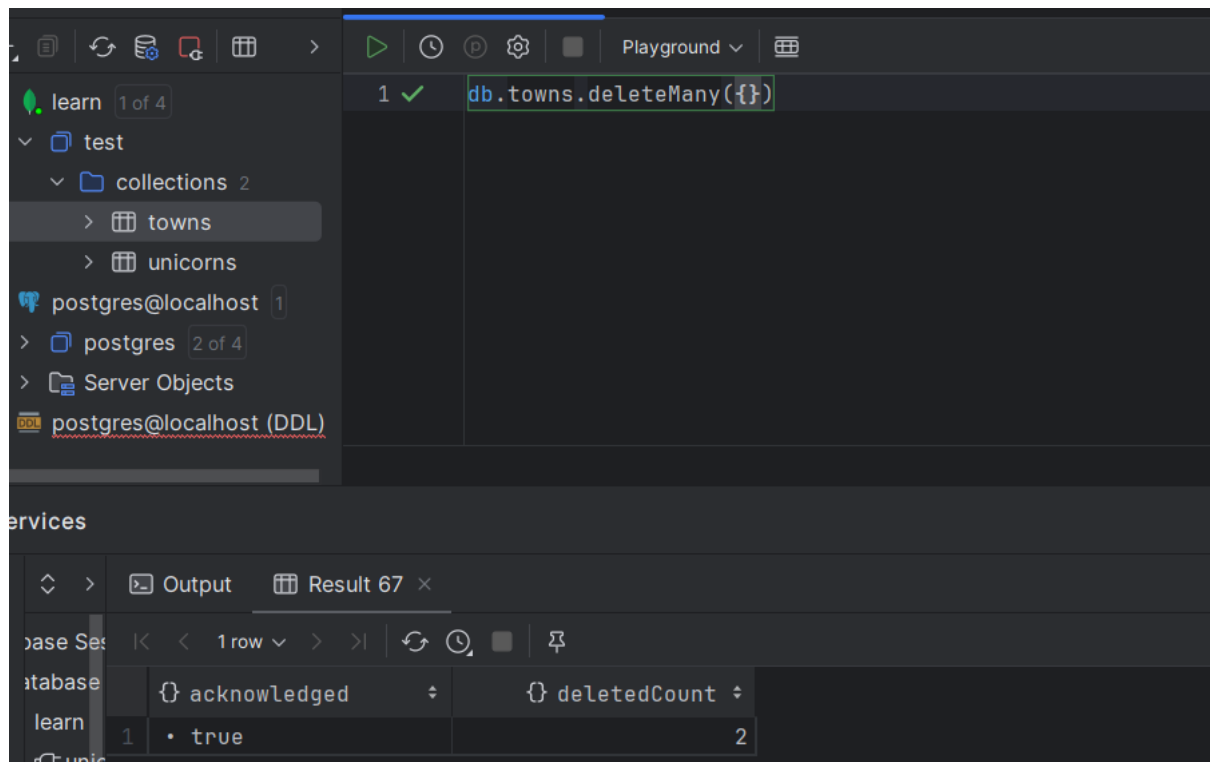
644a71e1412

["beer", "food"]

2009-07-20T00:00:00.000Z

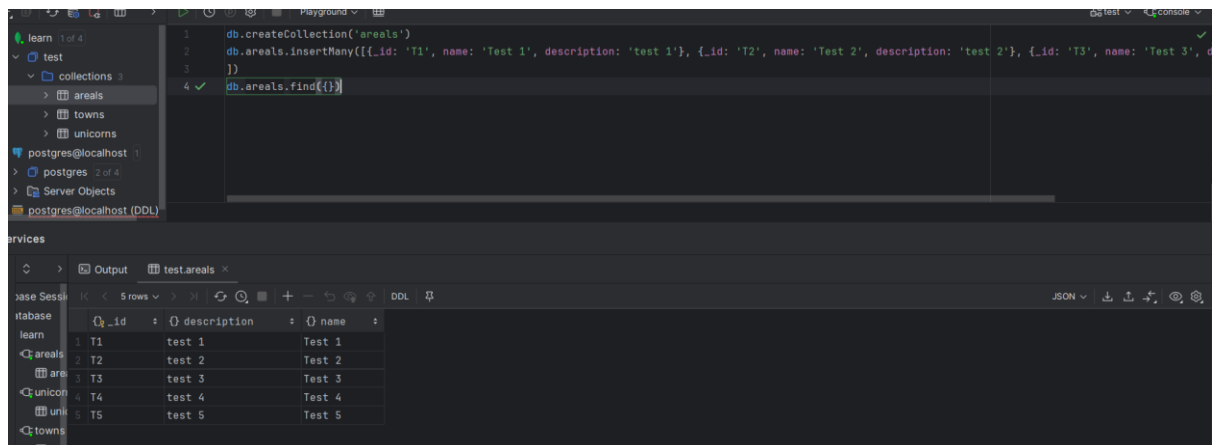
{"name": "Sam Adams", "party": "Portland"

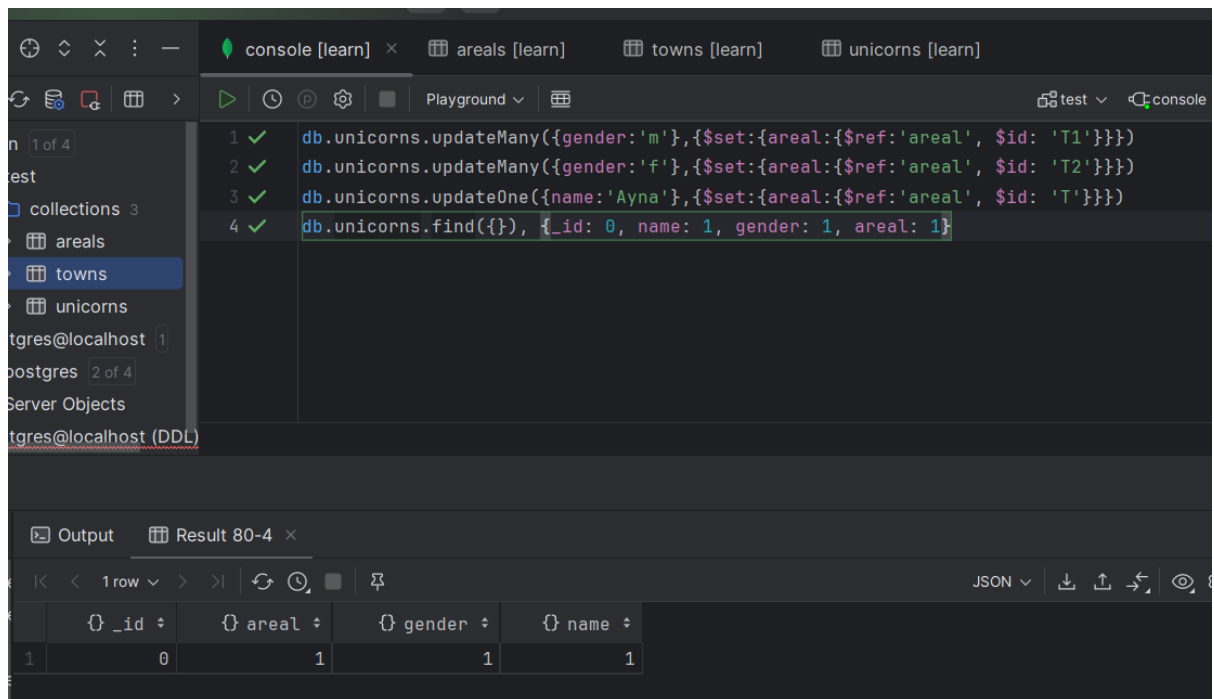
528000



Практическое задание 4.4.1:

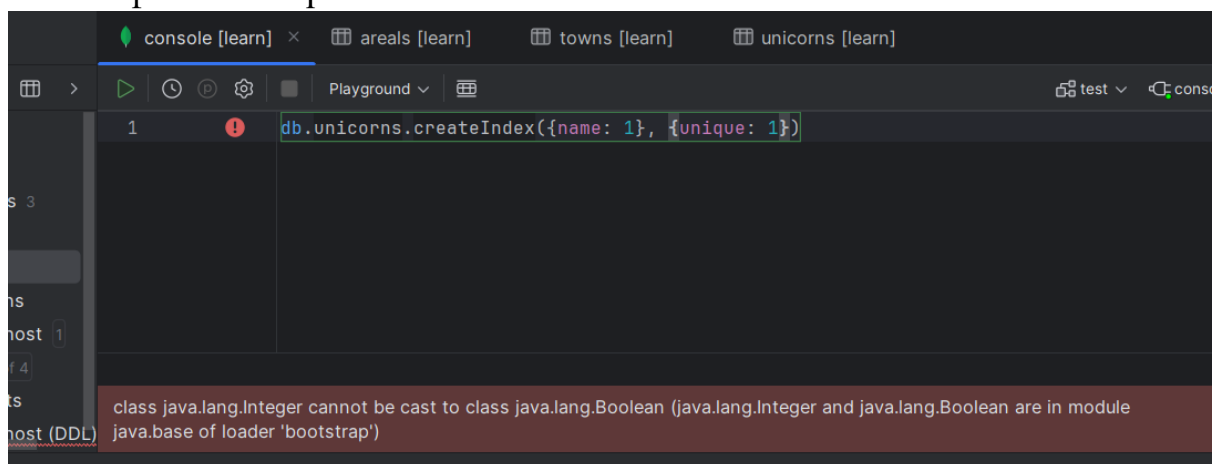
1. Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.





Практическое задание 4.4.2:

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



Закключение:

В процессе выполнения лабораторной задачи был произведен анализ и практическое применение ключевых команд MongoDB. Полученный опыт дает необходимые предпосылки для успешного взаимодействия с указанной системой управления базами данных.

Были изучены основы структурирования данных в MongoDB и ее подходы к управлению информацией. В отличие от классических реляционных баз данных, MongoDB сохраняет данные в гибкой структуре, напоминающей формат JSON.

Проанализированы команды вставки MongoDB для добавления новых записей в коллекцию, включая функции `insertOne`, `insertMany` и `save`. Изучены методы извлечения данных из MongoDB, что включает в себя использование функции `find`, применение различных фильтров, а также методы сортировки и ограничения.

В итоге выполнения этой работы получены значимые инструменты и знания для эффективного управления информацией и создания масштабируемых приложений с помощью MongoDB.