

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

**Отчет**

по лабораторной работе «Работа с БД в СУБД MongoDB»  
по дисциплине «Проектирование и реализация баз данных»

Автор: Булыга Е.А.  
Факультет: ИКТ  
Группа: К32421  
Преподаватель: Говорова М.М.

**itmo**

Санкт-Петербург 2023

# Содержание

<b>1</b>	<b>Введение</b>	<b>3</b>
<b>2</b>	<b>Выполнение</b>	<b>4</b>
2.1	CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ . . . . .	4
2.1.1	ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ . . . . .	4
2.1.2	ВЫБОРКА ДАННЫХ ИЗ БД . . . . .	4
2.1.3	ЛОГИЧЕСКИЕ ОПЕРАТОРЫ . . . . .	6
2.2	ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB. ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ . . . . .	9
2.2.1	ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ . . . . .	9
2.2.2	ИСПОЛЬЗОВАНИЕ JAVASCRIPT. КУРСОРЫ . . . . .	9
2.2.3	АГРЕГИРОВАННЫЕ ЗАПРОСЫ . . . . .	10
2.2.4	РЕДАКТИРОВАНИЕ ДАННЫХ . . . . .	11
2.2.5	УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ . . . . .	15
2.3	ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB . . . . .	16
2.3.1	ССЫЛКИ В БД . . . . .	16
2.3.2	НАСТРОЙКА ИНДЕКСОВ . . . . .	16
2.3.3	УПРАВЛЕНИЕ ИНДЕКСАМИ . . . . .	18
2.3.4	ПЛАН ЗАПРОСА . . . . .	18
<b>3</b>	<b>Вывод</b>	<b>20</b>

# 1 Введение

**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.

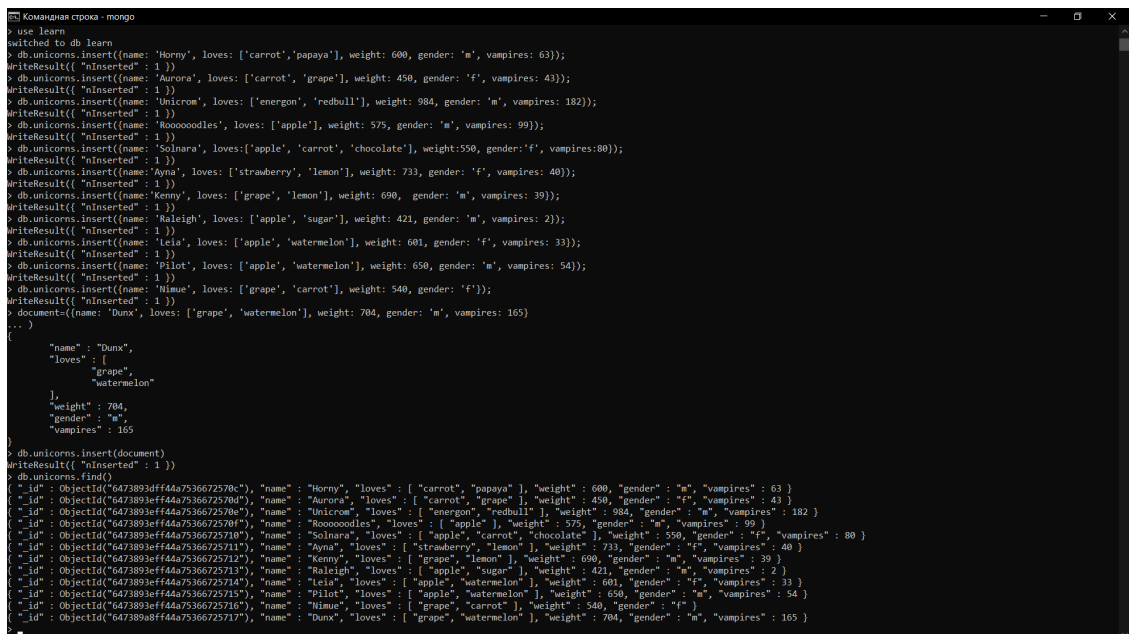
## 2 Выполнение

### 2.1 CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ

#### 2.1.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

##### Практическое задание 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.



```
Командная строка - mongo
> use learn
switched to db learn
> db.unicorns.insert((name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Unicorn', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Roosoodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f', vampires: 80));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Kenni', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54));
WriteResult({"inserted" : 1 })
> db.unicorns.insert((name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'));
WriteResult({"inserted" : 1 })
> document((name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165)
...
{
  "name": "Dunx",
  "loves": {
    "grape",
    "watermelon"
  },
  "weight": 704,
  "gender": "m",
  "vampires": 165
})
> db.unicorns.insert(document)
WriteResult({"inserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("64738934ff44a7336672570e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("64738934ff44a7336672570d"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("64738934ff44a7336672570c"), "name" : "Unicorn", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("64738934ff44a7336672570b"), "name" : "Roosoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("64738934ff44a7336672570a"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("64738934ff44a73366725709"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("64738934ff44a73366725708"), "name" : "Kenni", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("64738934ff44a73366725707"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("64738934ff44a73366725706"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("64738934ff44a73366725705"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("64738934ff44a73366725704"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("64738934ff44a73366725703"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Рис. 1: Практическое задание 8.1.1

#### 2.1.2 ВЫБОРКА ДАННЫХ ИЗ БД

##### Практическое задание 2

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

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

```
Командная строка - mongo
> db.unicorns.find({"gender": "m"})
{"_id": ObjectId("6473893dffa4a7536672570c"), "name": "Horny", "loves": [ "carrot", "papaya" ], "weight": 600, "gender": "m", "vampires": 63 }
{"_id": ObjectId("6473893eff44a7536672570e"), "name": "Unicrom", "loves": [ "energon", "redbull" ], "weight": 984, "gender": "m", "vampires": 182 }
{"_id": ObjectId("6473893eff44a7536672570f"), "name": "Rooooooodles", "loves": [ "apple", "weight": 575, "gender": "m", "vampires": 99 }
{"_id": ObjectId("6473893eff44a75366725712"), "name": "Kenny", "loves": [ "grape", "lemon", "weight": 690, "gender": "m", "vampires": 39 }
{"_id": ObjectId("6473893eff44a75366725713"), "name": "Raleigh", "loves": [ "apple", "sugar", "weight": 421, "gender": "m", "vampires": 2 }
{"_id": ObjectId("6473893eff44a75366725715"), "name": "Pilot", "loves": [ "apple", "watermelon", "weight": 650, "gender": "m", "vampires": 54 }
{"_id": ObjectId("647389a8ffa4a75366725717"), "name": "Dunx", "loves": [ "grape", "watermelon", "weight": 704, "gender": "m", "vampires": 165 }
db.unicorns.find({"gender": "f"})
{"_id": ObjectId("6473893eff44a7536672570d"), "name": "Aurora", "loves": [ "carrot", "grape", "weight": 450, "gender": "f", "vampires": 43 }
{"_id": ObjectId("6473893eff44a75366725710"), "name": "Solnara", "loves": [ "apple", "carrot", "chocolate", "weight": 550, "gender": "f", "vampires": 80 }
{"_id": ObjectId("6473893eff44a75366725711"), "name": "Ayna", "loves": [ "strawberry", "lemon", "weight": 733, "gender": "f", "vampires": 40 }
{"_id": ObjectId("6473893eff44a75366725714"), "name": "Leia", "loves": [ "apple", "watermelon", "weight": 601, "gender": "f", "vampires": 33 }
{"_id": ObjectId("6473893eff44a75366725716"), "name": "Nisae", "loves": [ "grape", "carrot", "weight": 540, "gender": "f" }
db.unicorns.find({"gender": "f"}).limit(3)
{"_id": ObjectId("6473893eff44a7536672570d"), "name": "Aurora", "loves": [ "carrot", "grape", "weight": 450, "gender": "f", "vampires": 43 }
{"_id": ObjectId("6473893eff44a75366725710"), "name": "Solnara", "loves": [ "apple", "carrot", "chocolate", "weight": 550, "gender": "f", "vampires": 80 }
{"_id": ObjectId("6473893eff44a75366725711"), "name": "Ayna", "loves": [ "strawberry", "lemon", "weight": 733, "gender": "f", "vampires": 40 }
db.unicorns.find({"gender": "m"}).sort({"name": 1})
{"_id": ObjectId("6473893eff44a7536672570c"), "name": "Dunx", "loves": [ "grape", "watermelon", "weight": 704, "gender": "m", "vampires": 165 }
{"_id": ObjectId("6473893dffa4a7536672570e"), "name": "Horny", "loves": [ "carrot", "papaya", "weight": 600, "gender": "m", "vampires": 63 }
{"_id": ObjectId("6473893eff44a75366725712"), "name": "Kenny", "loves": [ "grape", "lemon", "weight": 690, "gender": "m", "vampires": 39 }
{"_id": ObjectId("6473893eff44a75366725715"), "name": "Pilot", "loves": [ "apple", "watermelon", "weight": 650, "gender": "m", "vampires": 54 }
{"_id": ObjectId("6473893eff44a75366725713"), "name": "Raleigh", "loves": [ "apple", "sugar", "weight": 421, "gender": "m", "vampires": 2 }
{"_id": ObjectId("6473893eff44a7536672570f"), "name": "Rooooooodles", "loves": [ "apple", "weight": 575, "gender": "m", "vampires": 99 }
{"_id": ObjectId("6473893eff44a7536672570e"), "name": "Unicrom", "loves": [ "energon", "redbull", "weight": 984, "gender": "m", "vampires": 182 }
db.unicorns.find({"gender": "f"}).limit(3).sort({"name": 1})
{"_id": ObjectId("6473893eff44a7536672570d"), "name": "Aurora", "loves": [ "carrot", "grape", "weight": 450, "gender": "f", "vampires": 43 }
{"_id": ObjectId("6473893eff44a75366725711"), "name": "Ayna", "loves": [ "strawberry", "lemon", "weight": 733, "gender": "f", "vampires": 40 }
{"_id": ObjectId("6473893eff44a75366725714"), "name": "Leia", "loves": [ "apple", "watermelon", "weight": 601, "gender": "f", "vampires": 33 }
db.unicorns.findOne({"gender": "f", "loves": "carrot"})
{"_id": ObjectId("6473893eff44a7536672570d"), "name": "Aurora", "loves": [ "carrot", "grape" ], "weight": 450, "gender": "f", "vampires": 43 }
db.unicorns.find({"gender": "f", "loves": "carrot"}).limit(1)
{"_id": ObjectId("6473893eff44a7536672570d"), "name": "Aurora", "loves": [ "carrot", "grape", "weight": 450, "gender": "f", "vampires": 43 }
```

Рис. 2: Практическое задание 8.1.2

## Практическое задание 3

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

```
Командная строка - mongo
> db.unicorns.find({"gender": "m"}, {"loves": 0, "vampires": 0})
{"_id": ObjectId("6473893dffa4a7536672570c"), "name": "Horny", "weight": 600, "gender": "m" }
{"_id": ObjectId("6473893eff44a7536672570e"), "name": "Unicrom", "weight": 984, "gender": "m" }
{"_id": ObjectId("6473893eff44a7536672570f"), "name": "Rooooooodles", "weight": 575, "gender": "m" }
{"_id": ObjectId("6473893eff44a75366725712"), "name": "Kenny", "weight": 690, "gender": "m" }
{"_id": ObjectId("6473893eff44a75366725713"), "name": "Raleigh", "weight": 421, "gender": "m" }
{"_id": ObjectId("6473893eff44a75366725715"), "name": "Pilot", "weight": 650, "gender": "m" }
{"_id": ObjectId("647389a8ffa4a75366725717"), "name": "Dunx", "weight": 704, "gender": "m" }
```

Рис. 3: Практическое задание 8.1.3

## Практическое задание 4

Вывести список единорогов в обратном порядке добавления.

```

Командная строка - mongo
> db.unicorns.find().sort({ $natural: -1 })
{ "_id" : ObjectId("6473893eff44a75366725717"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6473893eff44a75366725716"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6473893eff44a75366725715"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6473893eff44a75366725714"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6473893eff44a75366725713"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6473893eff44a75366725712"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6473893eff44a75366725711"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6473893eff44a75366725710"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6473893eff44a7536672570f"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6473893eff44a7536672570e"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6473893eff44a7536672570d"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6473893eff44a7536672570c"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }

```

Рис. 4: Практическое задание 8.1.4

## Практическое задание 5

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

```

Командная строка - mongo
> db.unicorns.find({}, {"loves": {$slice: 1}, "_id": 0})
{ "name" : "Horny", "loves" : [ "carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "name" : "Aurora", "loves" : [ "carrot" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "name" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "name" : "Solnara", "loves" : [ "apple" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "name" : "Ayna", "loves" : [ "strawberry" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "name" : "Leia", "loves" : [ "apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape" ], "weight" : 540, "gender" : "f" }
{ "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }

```

Рис. 5: Практическое задание 8.1.5

## 2.1.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

### Практическое задание 6

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

```

Командная строка - mongo
> db.unicorns.find({"weight": {$gt: 500, $lt: 700}}, {"_id": 0})
{ "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>

```

Рис. 6: Практическое задание 8.1.6

## Практическое задание 7

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

```

Командная строка - mongo
> db.unicorns.find({"gender": "m", "weight": {$gt: 500}, "loves": {$all: ["grape", "lemon"]}}, {"_id": 0})
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
>

```

Рис. 7: Практическое задание 8.1.7

## Практическое задание 8

Найти всех единорогов, не имеющих ключ vampires.

```

Командная строка - mongo
> db.unicorns.find({"vampires": {$exists: false}})
{ "_id" : ObjectId("6473893eff44a75366725716"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>

```

Рис. 8: Практическое задание 8.1.8

## Практическое задание 9

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

```
Командная строка - mongo
> db.unicorns.find({"gender": "m"}, {"loves": {$slice: 1}, "name": true, "_id": 0}).sort({"name": 1})
{ "name" : "Dunx", "loves" : [ "grape" ] }
{ "name" : "Horny", "loves" : [ "carrot" ] }
{ "name" : "Kenny", "loves" : [ "grape" ] }
{ "name" : "Pilot", "loves" : [ "apple" ] }
{ "name" : "Raleigh", "loves" : [ "apple" ] }
{ "name" : "Rooodoodles", "loves" : [ "apple" ] }
{ "name" : "Unicrom", "loves" : [ "energon" ] }
>
```

Рис. 9: Практическое задание 8.1.9

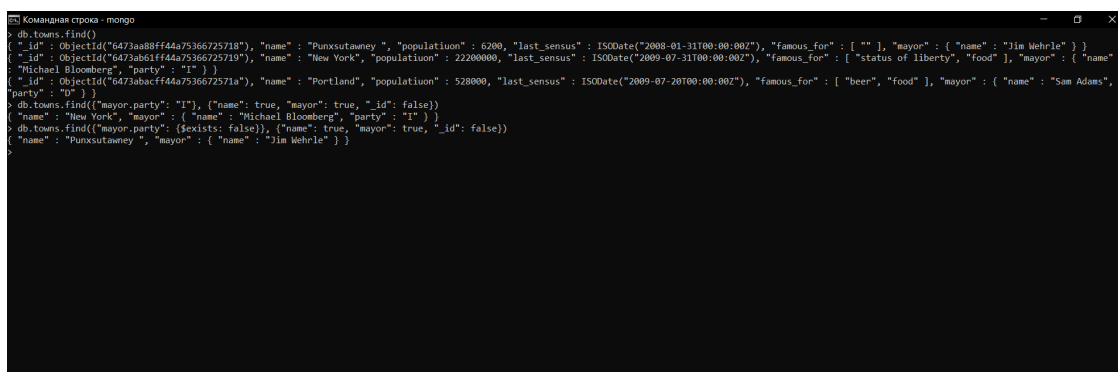


## 2.2 ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB. ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ

### 2.2.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

#### Практическое задание 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.



```
БЭ Командная строка - mongo
> db.towns.find()
{ "_id" : ObjectId("6473aa88ff44a75366725718"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Mehrle" } }
{ "_id" : ObjectId("6473ab61ff44a75366725719"), "name" : "New York", "population" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("6473abacff44a7536672571a"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
> db.towns.find({"mayor.party": "I"}, {"name": true, "mayor": true, "_id": false})
{ "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
> db.towns.find({"mayor.party": {"exists": false}}, {"name": true, "mayor": true, "_id": false})
{ "name" : "Punxsutawney ", "mayor" : { "name" : "Jim Mehrle" } }
```

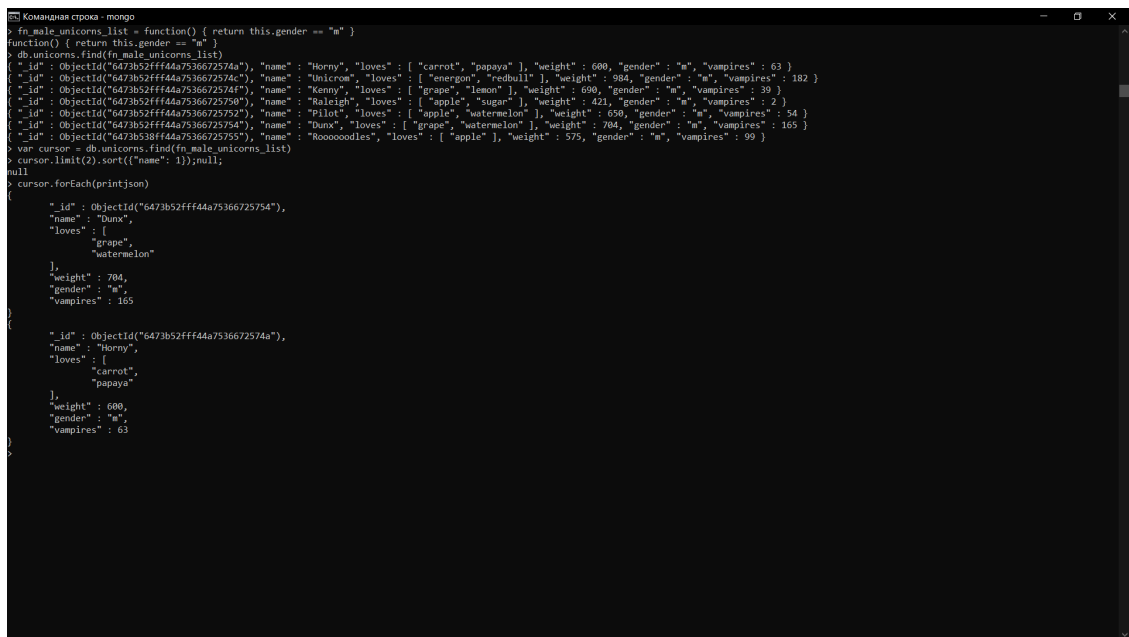
Рис. 10: Практическое задание 8.2.1

### 2.2.2 ИСПОЛЬЗОВАНИЕ JAVASCRIPT. КУРСОРЫ

#### Практическое задание 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*.

#### 4. Содержание коллекции единорогов unicorns.



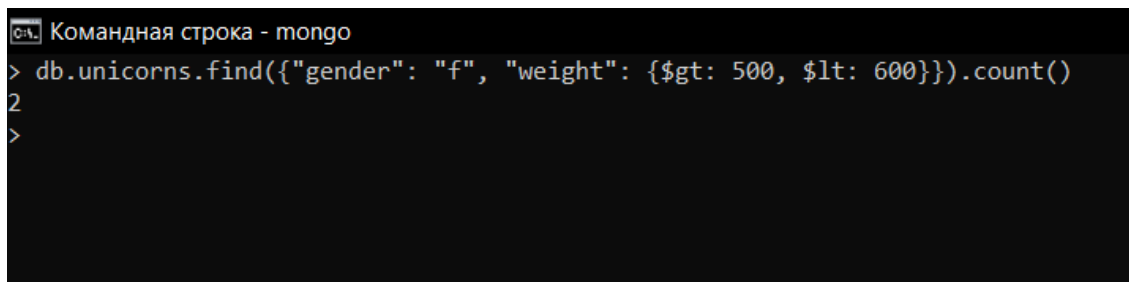
```
Командная строка - mongo
> fn_male_unicorns_list = function() { return this.gender == "m" }
function() { return this.gender == "m" }
> db.unicorns.find(fn_male_unicorns_list)
{ "_id" : ObjectId("6473b52fff44a7536672574a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6473b52fff44a7536672574c"), "name" : "Unicorn", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6473b52fff44a7536672574f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6473b52fff44a75366725750"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6473b52fff44a75366725752"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6473b52fff44a75366725754"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6473b52fff44a75366725755"), "name" : "Roocoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 90 }
> var cursor = db.unicorns.find(fn_male_unicorns_list)
> cursor.limit(2).sort({"name": 1});null;
null
> cursor.forEach(printjson)
{
  "_id" : ObjectId("6473b52fff44a75366725754"),
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
{
  "_id" : ObjectId("6473b52fff44a7536672574a"),
  "name" : "Horny",
  "loves" : [
    "carrot",
    "papaya"
  ],
  "weight" : 600,
  "gender" : "m",
  "vampires" : 63
}
```

Рис. 11: Практическое задание 8.2.2

### 2.2.3 АГРЕГИРОВАННЫЕ ЗАПРОСЫ

#### Практическое задание 3

Вывести количество самок единорогов весом от полутонны до 600 кг.



```
Командная строка - mongo
> db.unicorns.find({"gender": "f", "weight": {$gt: 500, $lt: 600}}).count()
2
>
```

Рис. 12: Практическое задание 8.2.3

#### Практическое задание 4

Вывести список предпочтений.

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

Рис. 13: Практическое задание 8.2.4

## Практическое задание 5

Посчитать количество особей единорогов обоих полов.

```
Командная строка - mongo
> db.unicorns.aggregate({"$group": {"_id": "$gender", "count": {"$sum": 1}}})
{ "_id" : "m", "count" : 7 }
{ "_id" : "f", "count" : 5 }
>
```

Рис. 14: Практическое задание 8.2.5

## 2.2.4 РЕДАКТИРОВАНИЕ ДАННЫХ

### Практическое задание 6

1. Выполнить команду.

## 2. Проверить содержимое коллекции *unicorns*.

```
Командная строка - mongo
> db.unicorns.save({name: 'Barny', loves: ['grape'],
... weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6473b52fff44a7536672574a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6473b52fff44a7536672574b"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6473b52fff44a7536672574c"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6473b52fff44a7536672574d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6473b52fff44a7536672574e"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6473b52fff44a7536672574f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6473b52fff44a75366725750"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6473b52fff44a75366725751"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6473b52fff44a75366725752"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6473b52fff44a75366725753"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6473b52fff44a75366725754"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6473b538ff44a75366725755"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6473b97bff44a75366725756"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Рис. 15: Практическое задание 8.2.6

## Практическое задание 7

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

```
Командная строка - mongo
> db.unicorns.update({"name": "Ayna", "gender": "f"}, {"name": "Ayna", "gender": "f", "weight": 800, "vampires": 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6473b52fff44a7536672574a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6473b52fff44a7536672574b"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6473b52fff44a7536672574c"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6473b52fff44a7536672574d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6473b52fff44a7536672574f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6473b52fff44a75366725750"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6473b52fff44a75366725751"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6473b52fff44a75366725752"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6473b52fff44a75366725753"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6473b52fff44a75366725754"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6473b538ff44a75366725755"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6473b97bff44a75366725756"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
{ "_id" : ObjectId("6473b00ff44a75366725757"), "name" : "Ayna", "gender" : "f", "weight" : 800, "vampires" : 51 }
```

Рис. 16: Практическое задание 8.2.7

## Практическое задание 8

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

## 2. Проверить содержимое коллекции *unicorns*.

```
Командная строка - mongo
> db.unicorns.update({"name": "Raleigh"}, {$set: {"loves": ["redbull"]}})
WriteResult({"nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{"_id" : ObjectId("6473b52fff44a7536672574a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{"_id" : ObjectId("6473b52fff44a7536672574b"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{"_id" : ObjectId("6473b52fff44a7536672574c"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{"_id" : ObjectId("6473b52fff44a7536672574d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("6473b52fff44a7536672574f"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("6473b52fff44a75366725750"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("6473b52fff44a75366725751"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{"_id" : ObjectId("6473b52fff44a75366725752"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{"_id" : ObjectId("6473b52fff44a75366725753"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{"_id" : ObjectId("6473b52fff44a75366725754"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{"_id" : ObjectId("6473b52fff44a75366725755"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{"_id" : ObjectId("6473b538ff44a75366725756"), "name" : "Roooonoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{"_id" : ObjectId("6473b97bfff44a75366725757"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
{"_id" : ObjectId("6473bb00ff44a75366725758"), "name" : "Ayna", "gender" : "f", "weight" : 800, "vampires" : 51 }
```

Рис. 17: Практическое задание 8.2.8

## Практическое задание 9

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

```
Командная строка - mongo
> db.unicorns.update({"gender": "m"}, {$inc: {"vampires": 5}}, {multi: true})
WriteResult({"nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
> db.unicorns.find()
{"_id" : ObjectId("6473b52fff44a7536672574a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{"_id" : ObjectId("6473b52fff44a7536672574b"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{"_id" : ObjectId("6473b52fff44a7536672574c"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{"_id" : ObjectId("6473b52fff44a7536672574d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("6473b52fff44a7536672574f"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("6473b52fff44a75366725750"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("6473b52fff44a75366725751"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{"_id" : ObjectId("6473b52fff44a75366725752"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{"_id" : ObjectId("6473b52fff44a75366725753"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{"_id" : ObjectId("6473b52fff44a75366725754"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{"_id" : ObjectId("6473b52fff44a75366725755"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{"_id" : ObjectId("6473b538ff44a75366725756"), "name" : "Roooonoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{"_id" : ObjectId("6473b97bfff44a75366725757"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
{"_id" : ObjectId("6473bb00ff44a75366725758"), "name" : "Ayna", "gender" : "f", "weight" : 800, "vampires" : 51 }
```

Рис. 18: Практическое задание 8.2.9

## Практическое задание 10

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

## 2. Проверить содержимое коллекции *unicorns*.

```
Командная строка - mongo
> db.towns.update({"name": "Portland"}, {$set: {"mayor.party": 1}})
WriteResult({"nMatched": 1, "nUpserted": 0, "nModified": 1})
> db.towns.find()
{"_id": ObjectId("6473a88ff44a75366725718"), "name": "Punxsutawney", "population": 6200, "last_sensus": ISODate("2008-01-31T00:00:00Z"), "famous_for": [ "" ], "mayor": { "name": "Jim Mehrle" } }
{"_id": ObjectId("6473ab61ff44a75366725719"), "name": "New York", "population": 22200000, "last_sensus": ISODate("2009-07-31T00:00:00Z"), "famous_for": [ "status of liberty", "food" ], "mayor": { "name": "Michael Bloomberg", "party": "R" } }
{"_id": ObjectId("6473abacff44a7536672571a"), "name": "Portland", "population": 528000, "last_sensus": ISODate("2009-07-20T00:00:00Z"), "famous_for": [ "beer", "food" ], "mayor": { "name": "Sam Adams" } }
```

Рис. 19: Практическое задание 8.2.10

## Практическое задание 11

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

```
Командная строка - mongo
> db.unicorns.update({"name": "Pilot"}, {$push: {"loves": "chocolate"}})
WriteResult({"nMatched": 1, "nUpserted": 0, "nModified": 1})
> db.unicorns.find()
{"_id": ObjectId("6473b52fff44a7536672574a"), "name": "Horny", "loves": [ "carrot", "papaya" ], "weight": 600, "gender": "m", "vampires": 68 }
{"_id": ObjectId("6473b52fff44a7536672574b"), "name": "Aurora", "loves": [ "carrot", "grape" ], "weight": 450, "gender": "f", "vampires": 43 }
{"_id": ObjectId("6473b52fff44a7536672574c"), "name": "Unicorn", "loves": [ "energon", "redbull" ], "weight": 984, "gender": "m", "vampires": 187 }
{"_id": ObjectId("6473b52fff44a7536672574d"), "name": "Solnara", "loves": [ "apple", "carrot", "chocolate" ], "weight": 950, "gender": "f", "vampires": 80 }
{"_id": ObjectId("6473b52fff44a7536672574f"), "name": "Kenny", "loves": [ "grape", "lemon" ], "weight": 690, "gender": "m", "vampires": 44 }
{"_id": ObjectId("6473b52fff44a75366725750"), "name": "Raleigh", "loves": [ "redbull" ], "weight": 421, "gender": "m", "vampires": 7 }
{"_id": ObjectId("6473b52fff44a75366725751"), "name": "Leia", "loves": [ "apple", "watermelon" ], "weight": 601, "gender": "f", "vampires": 33 }
{"_id": ObjectId("6473b52fff44a75366725752"), "name": "Pilot", "loves": [ "apple", "watermelon", "chocolate" ], "weight": 650, "gender": "m", "vampires": 59 }
{"_id": ObjectId("6473b52fff44a75366725753"), "name": "Nimue", "loves": [ "grape", "carrot" ], "weight": 540, "gender": "f" }
{"_id": ObjectId("6473b52fff44a75366725754"), "name": "Dunx", "loves": [ "grape", "watermelon" ], "weight": 704, "gender": "m", "vampires": 170 }
{"_id": ObjectId("6473b538ff44a75366725755"), "name": "Rooooooodles", "loves": [ "apple" ], "weight": 575, "gender": "m", "vampires": 104 }
{"_id": ObjectId("6473b97bfff44a75366725756"), "name": "Barny", "loves": [ "grape" ], "weight": 340, "gender": "m", "vampires": 5 }
{"_id": ObjectId("6473bb00ff44a75366725757"), "name": "Ayna", "gender": "f", "weight": 800, "vampires": 51 }
```

Рис. 20: Практическое задание 8.2.11

## Практическое задание 12

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

```

Командная строка - mongo
> db.unicorns.update({"name": "Aurora", "gender": "f"}, {$addToSet: {"loves": {$each: ["sugar", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6473b52ffff44a7536672574a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6473b52ffff44a7536672574b"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6473b52ffff44a7536672574c"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6473b52ffff44a7536672574d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6473b52ffff44a7536672574f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6473b52ffff44a75366725750"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6473b52ffff44a75366725751"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6473b52ffff44a75366725752"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6473b52ffff44a75366725753"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6473b52ffff44a75366725754"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("6473b538ff44a75366725755"), "name" : "Roocooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6473b97bfff44a75366725756"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
{ "_id" : ObjectId("6473bb00ff44a75366725757"), "name" : "Ayna", "gender" : "f", "weight" : 800, "vampires" : 51 }
>

```

Рис. 21: Практическое задание 8.2.12

## 2.2.5 УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

### Практическое задание 13

1. Создайте коллекцию *towns*, включающую следующие документы.
2. Удалите документы с беспартийными мэрами.
3. Проверьте содержание коллекции.
4. Очистите коллекцию.
5. Просмотрите список доступных коллекций.

```

Командная строка - mongo
WriteResult({ "nInserted" : 1 })
> db.towns.find()
{ "_id" : ObjectId("6473c0df44a75366725758"), "name" : "Punxsutawney", "popujatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "phil the groundhog" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("6473c10eff44a75366725759"), "name" : "New York", "popujatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "R" } }
{ "_id" : ObjectId("6473c11aff44a7536672575a"), "name" : "Portland", "popujatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
> db.towns.remove({"mayor.party": {$exists: false}})
WriteResult({ "nRemoved" : 1 })
> db.towns.find()
{ "_id" : ObjectId("6473c10eff44a75366725759"), "name" : "New York", "popujatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "R" } }
{ "_id" : ObjectId("6473c11aff44a7536672575a"), "name" : "Portland", "popujatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
> db.towns.remove({})
WriteResult({ "nRemoved" : 2 })
> show collections
towns
unicorns
>

```

Рис. 22: Практическое задание 8.2.13

## 2.3 ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

### 2.3.1 ССЫЛКИ В БД

#### Практическое задание 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.

```
Командная строка - mongo
> db.zones.insert({"_id": "zone_1", "name": "mountain", "description": "звук журчащих родников и ручьев, шум рек и водопадов, свист ветра и крики горных птиц рождает прекрасную, гармоничную музыку дикой природы"})
WriteResult({"_id": "zone_1", "name": "mountain", "description": "звук журчащих родников и ручьев, шум рек и водопадов, свист ветра и крики горных птиц рождает прекрасную, гармоничную музыку дикой природы"})
> db.zones.insert({"_id": "zone_2", "name": "forest", "description": "Трава мокра от росы, а деревья стряхивают на путника целый дождь. В гуще леса темно. Но вот солнце пробивается сквозь листву и озаряет лесной мир."})
WriteResult({"_id": "zone_2", "name": "forest", "description": "Трава мокра от росы, а деревья стряхивают на путника целый дождь. В гуще леса темно. Но вот солнце пробивается сквозь листву и озаряет лесной мир."})
> db.zones.insert({"_id": "zone_3", "name": "heaven", "description": "Небо это начало вселенной, за ним бескрайние космические дали. Легкие облака на голубом небе кажутся барашками моря настоящего, а черные грозные тучи напоминают бурлящие штормовые валы."})
WriteResult({"_id": "zone_3", "name": "heaven", "description": "Небо это начало вселенной, за ним бескрайние космические дали. Легкие облака на голубом небе кажутся барашками моря настоящего, а черные грозные тучи напоминают бурлящие штормовые валы."})
> db.zones.find()
{"_id": "zone_1", "name": "mountain", "description": "звук журчащих родников и ручьев, шум рек и водопадов, свист ветра и крики горных птиц рождает прекрасную, гармоничную музыку дикой природы"}
{"_id": "zone_2", "name": "forest", "description": "Трава мокра от росы, а деревья стряхивают на путника целый дождь. В гуще леса темно. Но вот солнце пробивается сквозь листву и озаряет лесной мир."}
{"_id": "zone_3", "name": "heaven", "description": "Небо это начало вселенной, за ним бескрайние космические дали. Легкие облака на голубом небе кажутся барашками моря настоящего, а черные грозные тучи напоминают бурлящие штормовые валы."}
> db.unicorns.update("name": "Horny", {$set: {"live_zone": {"$ref": "zones", "$id": "zone_1"}}})
WriteResult({"_id": "Horny", "name": "Horny", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "nModified": 1})
> db.unicorns.update("name": "Aurora", {$set: {"live_zone": {"$ref": "zones", "$id": "zone_1"}}})
WriteResult({"_id": "Aurora", "name": "Aurora", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "nModified": 1})
> db.unicorns.update("name": "Kenny", {$set: {"live_zone": {"$ref": "zones", "$id": "zone_2"}}})
WriteResult({"_id": "Kenny", "name": "Kenny", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "nModified": 1})
> db.unicorns.find()
{"_id": "Horny", "name": "Horny", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "weight": 600, "gender": "m", "vampires": 63}
{"_id": "Aurora", "name": "Aurora", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "weight": 450, "gender": "f", "vampires": 43}
{"_id": "Unicrow", "name": "Unicrow", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "weight": 984, "gender": "m", "vampires": 182}
{"_id": "Solnara", "name": "Solnara", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "weight": 550, "gender": "f", "vampires": 80}
{"_id": "Ayna", "name": "Ayna", "live_zone": {"$ref": "zones", "$id": "zone_1"}, "weight": 733, "gender": "f", "vampires": 40}
{"_id": "Kenny", "name": "Kenny", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 690, "gender": "m", "vampires": 39}
{"_id": "Raleigh", "name": "Raleigh", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 421, "gender": "m", "vampires": 2}
{"_id": "Leia", "name": "Leia", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 601, "gender": "f", "vampires": 33}
{"_id": "Pilot", "name": "Pilot", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 650, "gender": "m", "vampires": 54}
{"_id": "Blame", "name": "Blame", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 540, "gender": "f", "vampires": 165}
{"_id": "Daxx", "name": "Daxx", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 704, "gender": "m", "vampires": 165}
{"_id": "Roocoodles", "name": "Roocoodles", "live_zone": {"$ref": "zones", "$id": "zone_2"}, "weight": 575, "gender": "m", "vampires": 99}
```

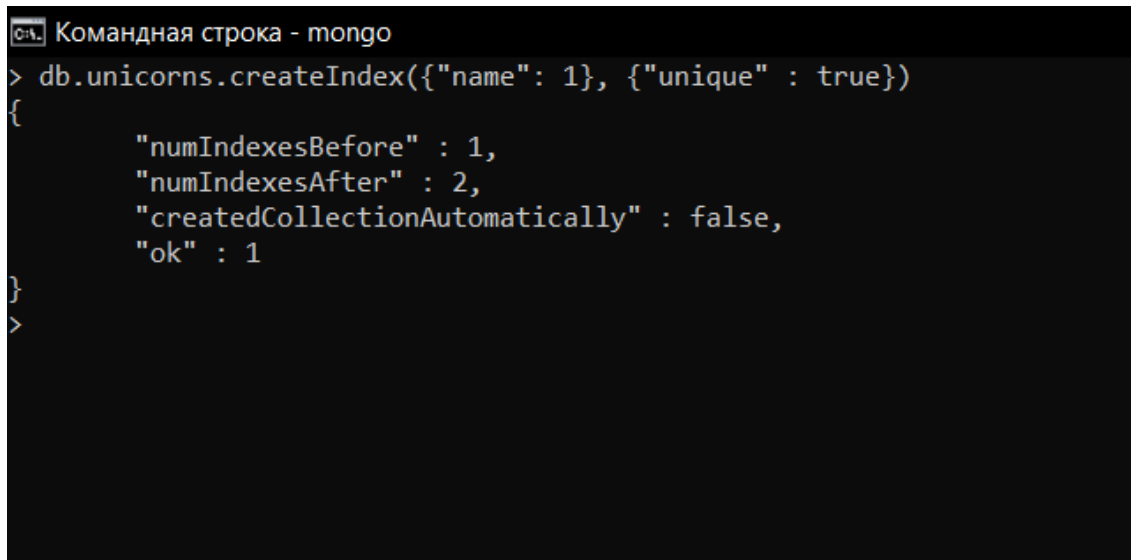
Рис. 23: Практическое задание 8.3.1

### 2.3.2 НАСТРОЙКА ИНДЕКСОВ

#### Практическое задание 2

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



A screenshot of a terminal window titled "Командная строка - mongo". The prompt is ">". The command entered is "db.unicorns.createIndex({"name": 1}, {"unique" : true})". The output is a JSON object: {"numIndexesBefore" : 1, "numIndexesAfter" : 2, "createdCollectionAutomatically" : false, "ok" : 1}.

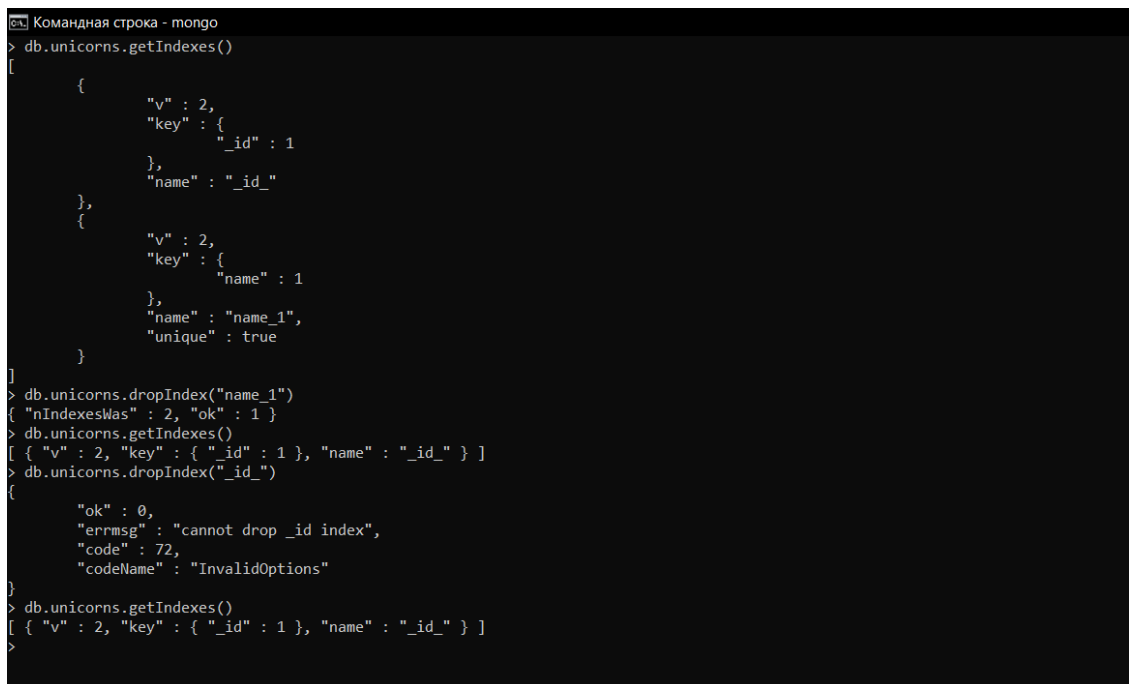
```
Командная строка - mongo
> db.unicorns.createIndex({"name": 1}, {"unique" : true})
{
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "createdCollectionAutomatically" : false,
  "ok" : 1
}
```

Рис. 24: Практическое задание 8.3.2

### 2.3.3 УПРАВЛЕНИЕ ИНДЕКСАМИ

#### Практическое задание 3

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



```
Командная строка - mongo
> db.unicorns.getIndexes()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_"
  },
  {
    "v" : 2,
    "key" : {
      "name" : 1
    },
    "name" : "name_1",
    "unique" : true
  }
]
> db.unicorns.dropIndex("name_1")
{ "nIndexesWas" : 2, "ok" : 1 }
> db.unicorns.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
> db.unicorns.dropIndex("_id_")
{
  "ok" : 0,
  "errmsg" : "cannot drop _id index",
  "code" : 72,
  "codeName" : "InvalidOptions"
}
> db.unicorns.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
>
```

Рис. 25: Практическое задание 8.3.3

### 2.3.4 ПЛАН ЗАПРОСА

#### Практическое задание 4

1. Создайте объемную коллекцию *numbers*, задействовав курсор.
2. Выберите последних четыре документа.
3. Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра *executionTime*)
4. Создайте индекс для ключа *value*.
5. Получите информацию о всех индексах коллекции *numbers*.

6. Выполните запрос 2.
7. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?
8. Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

```

Командная строка - mongo
> for(i = 0; i < 100000; i++) db.numbers.insert({value: i})
WriteResult({ "nInserted" : 1 })
> db.numbers.find().sort({$natural: -1}).limit(4)
{ "_id" : ObjectId("6473d84ff4da73667871e6"), "value" : 99999 }
{ "_id" : ObjectId("6473d84ff4da73667871e5"), "value" : 99998 }
{ "_id" : ObjectId("6473d84ff4da73667871e4"), "value" : 99997 }
{ "_id" : ObjectId("6473d84ff4da73667871e3"), "value" : 99996 }
> db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
{
  "explainVersion" : "1",
  "queryPlanner" : {
    "namespace" : "learn.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      }
    },
    "maxIndexedOrSolutionsReached" : false,
    "maxIndexedAndSolutionsReached" : false,
    "maxScansToExploreReached" : false,
    "winningPlan" : {
      "stage" : "LIMIT",
      "limitAmount" : 4,
      "inputStage" : {
        "stage" : "COLLSCAN",
        "direction" : "backward"
      }
    },
    "rejectedPlans" : [ ]
  },
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 4,
    "executionTimeMillis" : 7,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 4,
    "executionStages" : {
      "stage" : "LIMIT",
      "nReturned" : 4,
      "executionTimeMillisEstimate" : 0,
      "works" : 6,
      "advanced" : 4,
      "needTime" : 1,
      "needYield" : 0,
      "saveState" : 0,
      "restoreState" : 0,
      "isEOF" : 1,
      "limitAmount" : 4,
      "inputStage" : {
        "stage" : "COLLSCAN",
        "nReturned" : 4,
        "executionTimeMillisEstimate" : 0,
        "works" : 5,

```

Рис. 26: Практическое задание 8.3.4. ДО создания индекса

```

Командная строка - mongo
> db.numbers.createIndex({"value": 1})
{
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "createdCollectionAutomatically" : false,
  "ok" : 1
}
> db.numbers.getIndexes()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_"
  },
  {
    "v" : 2,
    "key" : {
      "value" : 1
    },
    "name" : "value_1"
  }
]
> db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
{
  "explainVersion" : "1",
  "queryPlanner" : {
    "namespace" : "learn.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      }
    },
    "maxIndexedOrSolutionsReached" : false,
    "maxIndexedAndSolutionsReached" : false,
    "maxScansToExploreReached" : false,
    "winningPlan" : {
      "stage" : "LIMIT",
      "limitAmount" : 4,
      "inputStage" : {
        "stage" : "COLLSCAN",
        "direction" : "backward"
      }
    },
    "rejectedPlans" : [ ]
  },
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 4,
    "executionTimeMillis" : 22,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 4,

```

Рис. 27: Практическое задание 8.3.4. ПОСЛЕ создания индекса

### 3 Вывод

В процессе выполнения лабораторной работы были освоены практические навыки работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.