

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

## **О Т Ч Е Т**

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

Автор: Бармина Диана Андреевна

Факультет: ИКТ

Группа: K32421

Преподаватель: Говорова М.М.

Дата сдачи: 22.05.23



Санкт-Петербург 2023

## СОДЕРЖАНИЕ

|   |    |
|---|----|
| 1 Описание работы .....   | 3  |
| 2 Коллекции .....   | 3  |
| 3 CRUD-операции в MONGODB. Вставка данных. Выборка данных ..... | 4  |
| 3.1 Вставка документов в коллекцию .....                        | 4  |
| 3.2 Выборка данных из БД .....                                  | 5  |
| 3.3 Логические операторы .....                                  | 8  |
| 4 Запросы к базе данных MONGODB .....                           | 10 |
| 4.1 Запрос к вложенным объектам .....                           | 10 |
| 4.2 Использование JavaScript и курсоров .....                   | 11 |
| 4.3 Агрегированные запросы .....                                | 12 |
| 4.4 Редактирование данных .....                                 | 13 |
| 4.5 Удаление данных из коллекции .....                          | 17 |
| 5 Ссылки и работа с индексами .....                             | 18 |
| 5.1 Ссылки в БД .....   | 18 |
| 5.2 Настройка индексов .....                                    | 19 |
| 5.3 Управление индексами .....                                  | 19 |
| 5.4 План запроса .....  | 20 |
| ЗАКЛЮЧЕНИЕ .....  | 21 |

## 1 Описание работы

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.

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

## 2 Коллекции

Коллекции заполнялись следующим набором данных:

```
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: 'Rooooooodles', 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'});
```

```
{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"}}
```

## 3 CRUD-операции в СУБД MONGODB. Вставка данных. Выборка данных

### 3.1 Вставка документов в коллекцию

Задачи: создать базу данных learn; заполнить коллекцию единорогов unicorns; используя второй способ, вставить в коллекцию единорогов документ; проверить содержимое коллекции.

- 1) Создание коллекции: use learn;
- 2) Выполнение запроса find после вставки данных:

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId("64650afca0a4910557e264f0"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("64650b01a0a4910557e264f1"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("64650b14a0a4910557e264f2"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("64650660a0a4910557e264e7"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("646506c4a0a4910557e264e8"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("646506fda0a4910557e264e9"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("64650730a0a4910557e264ea"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("6465075fa0a4910557e264eb"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("64650787a0a4910557e264ec"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("646507b7a0a4910557e264ed"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646507dba0a4910557e264ee"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("64650857a0a4910557e264ef"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

Рисунок 1 – Результат выполнения вставки документов

## 3.2 Выборка данных из БД

### 1) Практическое задание 8.1.2

Задача 1: сформировать запросы для вывода списка самцов и самок единорогов. Ограничить список самок первыми тремя особями. Отсортировать списки по имени.

Код:

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

Выполнение запроса 3:

```
learn> db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
[
  {
    _id: ObjectId("64650b01a0a4910557e264f1"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("64650b23a0a4910557e264f5"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("64650b32a0a4910557e264f8"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

Рисунок 2 – Выполнение запроса 3

Задача 2: найти всех самок, которые любят carrot, ограничить список первой особью с помощью функций `findOne` и `limit`.

Код:

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

## Выполнение запросов задачи 2:

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId("64650b01a0a4910557e264f1"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId("64650b01a0a4910557e264f1"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

Рисунок 3 – Выполнение запросов задачи 2

### 2) Практическое задание 8.1.3

Задача: модифицировать запрос для вывода списка самцов единорогов, исключив из результата информацию о предпочтениях и поле.

```
Db.unicorns.find({gender: 'm'}, {loves:0, vampires:0})
```

Выполнение запроса:

```
learn> db.unicorns.find({gender: 'm'}, {loves:0, gender:0})
[
  {
    _id: ObjectId("64650afca0a4910557e264f0"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("64650b14a0a4910557e264f2"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("64650b19a0a4910557e264f3"),
    name: 'Rooooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("64650b2aa0a4910557e264f6"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("64650b2ea0a4910557e264f7"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("64650b3aa0a4910557e264f9"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("64650b3ea0a4910557e264fa"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
]
```

Рисунок 4 – Выполнение запроса 8.1.3

### 3) Практическое задание 8.1.4

Задача: вывести список единорогов в обратном порядке добавления.

Для компактности были выведены только имена единорогов.

```
db.unicorns.find({}, {weight:0, loves:0, gender:0, vampires:0}).sort({$natural: -1})
```

Выполнение запроса:

```
learn> db.unicorns.find({}, {weight:0, loves:0, gender:0, vampires:0}).sort({$natural: -1})
[
  { _id: ObjectId("64660a4fa0a4910557e264fb"), name: 'Nimue' },
  { _id: ObjectId("64650b3ea0a4910557e264fa"), name: 'Dunx' },
  { _id: ObjectId("64650b3aa0a4910557e264f9"), name: 'Pilot' },
  { _id: ObjectId("64650b32a0a4910557e264f8"), name: 'Leia' },
  { _id: ObjectId("64650b2ea0a4910557e264f7"), name: 'Raleigh' },
  { _id: ObjectId("64650b2aa0a4910557e264f6"), name: 'Kenny' },
  { _id: ObjectId("64650b23a0a4910557e264f5"), name: 'Ayna' },
  { _id: ObjectId("64650b1ea0a4910557e264f4"), name: 'Solnara' },
  { _id: ObjectId("64650b19a0a4910557e264f3"), name: 'Roooooodles' },
  { _id: ObjectId("64650b14a0a4910557e264f2"), name: 'Unicrom' },
  { _id: ObjectId("64650b01a0a4910557e264f1"), name: 'Aurora' },
  { _id: ObjectId("64650afca0a4910557e264f0"), name: 'Horny' }
```

Рисунок 5 – Выполнение запроса 8.1.4

### 4) Практическое задание 8.1.5

Задача: вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор (для компактности вывода также был исключен vampires).

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

Выполнение запроса:

```
learn> db.unicorns.find({}, {_id: 0, vampires: 0, loves:{$slice:1}})
[
  { name: 'Horny', loves: [ 'carrot' ], weight: 600, gender: 'm' },
  { name: 'Aurora', loves: [ 'carrot' ], weight: 450, gender: 'f' },
  { name: 'Unicrom', loves: [ 'energon' ], weight: 984, gender: 'm' },
  { name: 'Roooooodles', loves: [ 'apple' ], weight: 575, gender: 'm' },
  { name: 'Solnara', loves: [ 'apple' ], weight: 550, gender: 'f' },
  { name: 'Ayna', loves: [ 'strawberry' ], weight: 733, gender: 'f' },
  { name: 'Kenny', loves: [ 'grape' ], weight: 690, gender: 'm' },
  { name: 'Raleigh', loves: [ 'apple' ], weight: 421, gender: 'm' },
  { name: 'Leia', loves: [ 'apple' ], weight: 601, gender: 'f' },
  { name: 'Pilot', loves: [ 'apple' ], weight: 650, gender: 'm' },
  { name: 'Dunx', loves: [ 'grape' ], weight: 704, gender: 'm' },
  { name: 'Nimue', loves: [ 'grape' ], weight: 540, gender: 'f' }
```

Рисунок 6 – Выполнение запроса 8.1.5

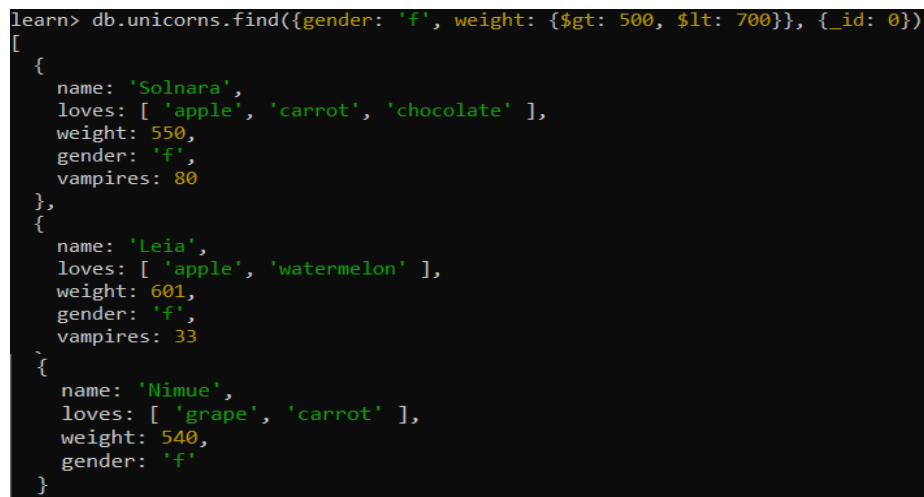
### 3.3 Логические операторы

#### 1) Практическое задание 8.1.6

Задача: вывести список самок единорогов весом от полутонны до 700 кг, исключив вывод идентификатора.

```
db.unicorns.find({gender: 'f', weight: {$gt: 500, $lt: 700}}, {_id: 0})
```

Выполнение запроса:



```
learn> db.unicorns.find({gender: 'f', weight: {$gt: 500, $lt: 700}}, {_id: 0})
[
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

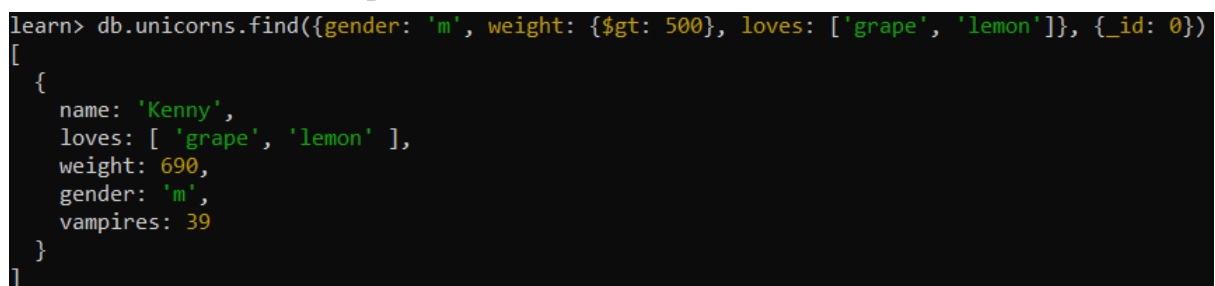
Рисунок 7 – Выполнение запроса 8.1.6

#### 2) Практическое задание 8.1.7

Задача: вывести список самцов единорогов весом от полутонны и предпочитающих grape и lemon, исключив вывод идентификатора.

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

Выполнение запроса:



```
learn> db.unicorns.find({gender: 'm', weight: {$gt: 500}, loves: ['grape', 'lemon']}, {_id: 0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Рисунок 8 – Выполнение запроса 8.1.7



### 3) Практическое задание 8.1.8

Задача: найти всех единорогов, не имеющих ключ vampires.

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

Выполнение запроса:

```
learn> db.unicorns.find({vampires:{$exists:false}})
[
  {
    _id: ObjectId("64660a4fa0a4910557e264fb"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Рисунок 9 – Выполнение запроса 8.1.8

### 4) Практическое задание 8.1.9

Задача: вывести упорядоченный список имен самцов единорогов с информацией об их первом предпочтении.

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

Выполнение запроса:

```
learn> db.unicorns.find({gender: 'm'}, {_id: 0, vampires: 0, weight:0, loves:{$slice:1}}).sort({name:1})
[
  { name: 'Dunx', loves: [ 'grape' ], gender: 'm' },
  { name: 'Horny', loves: [ 'carrot' ], gender: 'm' },
  { name: 'Kenny', loves: [ 'grape' ], gender: 'm' },
  { name: 'Pilot', loves: [ 'apple' ], gender: 'm' },
  { name: 'Raleigh', loves: [ 'apple' ], gender: 'm' },
  { name: 'Roooooodles', loves: [ 'apple' ], gender: 'm' },
  { name: 'Unicrom', loves: [ 'energon' ], gender: 'm' }
]
```

Рисунок 10 – Выполнение запроса 8.1.9

## 4 Запросы к базе данных MONGODB

### 4.1 Запрос к вложенным объектам

Задача: создать коллекцию towns; сформировать два запроса.

Запрос 1: вывести список городов с независимыми мэрами (вывести только название города и информацию о мэре).

```
db.towns.find({"mayor.party": "I"}, {_id:0, population:0, last_sensus:0, famous_for:0})
```

Выполнение запроса 1:

```
learn> db.towns.find({"mayor.party": "I"}, {_id:0, population:0, last_sensus:0, famous_for:0})
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

Рисунок 10 – Выполнение запроса 1 задачи 8.1.9

Запрос 2: вывести список беспартийных мэров (вывести только название города и информацию о мэре).

```
db.towns.find({"mayor.party": {$exists:false}}, {_id:0, population:0, last_sensus:0, famous_for:0})
```

Выполнение запроса 2:

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

Рисунок 11 – Выполнение запроса 2 задачи 8.1.9

## 4.2 Использование JavaScript и курсоров

Задача: сформировать функцию для вывода самцов единорогов; создать курсор для этого списка их первых двух особей с сортировкой в лексикографическом порядке; вывести результат, используя foreach.

```
func = function() {return this.gender=='m';}  
var cursor = db.unicorns.find({"$where": func}).limit(2).sort({name: 1})  
cursor.forEach(function(obj) {print(obj)})
```

Вывод запуска курсора через цикл:

```
learn> func = function() {return this.gender=='m';}  
[Function: func]  
learn> var cursor = db.unicorns.find({"$where": func}).limit(2).sort({name: 1})  
  
learn> cursor.forEach(function(obj) {print(obj)})  
{  
  _id: ObjectId("64650b3ea0a4910557e264fa"),  
  name: 'Dunx',  
  loves: [ 'grape', 'watermelon' ],  
  weight: 704,  
  gender: 'm',  
  vampires: 165  
}  
{  
  _id: ObjectId("64650afca0a4910557e264f0"),  
  name: 'Horny',  
  loves: [ 'carrot', 'papaya' ],  
  weight: 600,  
  gender: 'm',  
  vampires: 63  
}
```

Рисунок 12 – Работа с курсором и forEach

## 4.3 Агрегированные запросы

### 1) Практическое задание 8.2.3

Задача: вывести количество самок весом от полутонны до 600 кг.

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

Выполнение запроса:

```
learn> db.unicorns.find({gender: 'f', weight: {$gt:500, $lt:600}}).count()  
2
```

Рисунок 13 – Выполнение запроса 8.2.3

### 2) Практическое задание 8.2.4

Задача: вывести список предпочтений.

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

Выполнение запроса:

```
learn> db.unicorns.distinct("loves")  
[  
  'apple',      'carrot',  
  'chocolate', 'energon',  
  'grape',      'lemon',  
  'papaya',     'redbull',  
  'strawberry', 'sugar',  
  'watermelon'  
]  
learn> _
```

Рисунок 14 – Выполнение запроса 8.2.4

### 3) Практическое задание 8.2.4

Задача: посчитать количество особей единорогов обоих полов.

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

Выполнение запроса:

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

Рисунок 15 – Выполнение запроса 8.2.4

## 4.4 Редактирование данных

### 1) Практическое задание 8.2.6

Задача: выполнить команду и проверить содержимое коллекции.

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

Данная команда устарела, вместо нее была выполнена другая.

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

Выполнение запроса:

```
learn> db.unicorns.save({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
TypeError: db.unicorns.save is not a function
learn> db.unicorns.insertOne({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedId: ObjectId("646627cfa0a4910557e264ff")
}
learn> db.unicorns.count()
DeprecationWarning: Collection.count() is deprecated. Use countDocuments or estimatedDocumentCount.
13
```

Рисунок 16 – Выполнение запроса 8.2.6

### 2) Практическое задание 8.2.7

Задача: для самки единорога Айна внести изменения в БД: теперь ее вес 800, она убила 51 вампира.

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

Выполнение запроса:

```
learn> db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight: 800, vampires: 51}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId("64650b23a0a4910557e264f5"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

Рисунок 17 – Выполнение запроса 8.2.7

### 3) Практическое задание 8.2.8

Задача: для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.

```
db.unicorns.updateOne({name: 'Raleigh'}, {$set: {loves: ['redbull']}})
```

Выполнение запроса:

```
learn> db.unicorns.updateOne({name: 'Raleigh'}, {$set: {loves: ['redbull']}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Raleigh'})
[
  {
    _id: ObjectId("64650b2ea0a4910557e264f7"),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

Рисунок 18 – Выполнение запроса 8.2.8

### 4) Практическое задание 8.2.9

Задача: всем самцам единорогов увеличить количество убитых вампиров на 5.

```
db.unicorns.updateOne({gender: 'm'}, {$inc: {vampires:5}})
```

Выполнение запроса:

```
learn> db.unicorns.updateOne({gender: 'm'}, {$inc: {vampires:5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: 'm'}, {_id:0, loves:0})
[
  { name: 'Horny', weight: 600, gender: 'm', vampires: 68 },
  { name: 'Unicrom', weight: 984, gender: 'm', vampires: 182 },
  { name: 'Rooooooodles', weight: 575, gender: 'm', vampires: 99 },
  { name: 'Kenny', weight: 690, gender: 'm', vampires: 39 },
  { name: 'Raleigh', weight: 421, gender: 'm', vampires: 2 },
  { name: 'Pilot', weight: 650, gender: 'm', vampires: 54 },
  { name: 'Dunx', weight: 704, gender: 'm', vampires: 165 },
  { name: 'Barney', weight: 340, gender: 'm' }
]
```

Рисунок 19 – Выполнение запроса 8.2.9

### 5) Практическое задание 8.2.10

Задача: изменить информацию о городе Портланд: мэр этого города теперь беспартийный.

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

-Выполнение запроса:

```
learn> db.towns.updateOne({name: 'Portland'}, {$unset: {"mayor.party": 1}})
{ {
  acknowledged: true,650b14a0a4910557e264f2"),
  insertedId: null,,
  matchedCount: 1,gon', 'redbull' ],
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: 'Portland'}, {_id:0, population:0, last_sensus:0, famous_for:0})
[ { name: 'Portland', mayor: { name: 'Sam Adams' } } ]
```

Рисунок 20 – Выполнение запроса 8.2.10

### 6) Практическое задание 8.2.11

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

```
db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
```

Выполнение запроса:

```
learn> db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Pilot'}, {name:1, loves:1})
[
  {
    _id: ObjectId("64650b3aa0a4910557e264f9"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ]
  }
]
learn> _
```

Рисунок 21 – Выполнение запроса 8.2.11

## 7) Практическое задание 8.2.12

Задача: изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.

```
db.unicorns.updateOne({name: 'Aurora'}, {$addToSet: {loves: {$each: ['sugar', 'lemon']}}})
```

Выполнение запроса:

```
learn> db.unicorns.updateOne({name: 'Aurora'}, {$addToSet: {loves: {$each: ['sugar', 'lemon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Aurora'}, {name:1, loves:1})
[
  {
    _id: ObjectId("64650b01a0a4910557e264f1"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ]
  }
]
```

Рисунок 22 – Выполнение запроса 8.2.12



## 4.5 Удаление данных из коллекции

Задача: удалить документы с беспартийными мэрами, проверить содержание коллекции, очистить коллекцию, просмотреть список доступных коллекций.

```
db.towns.deleteMany({"mayor.party": {$exists:false}})
db.towns.find({}, {name:1, mayor:1})
db.towns.drop()
show collections
```

Выполнение запроса:

```
learn> db.towns.deleteMany({"mayor.party": {$exists:false}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find({}, {name:1, mayor:1})
[
  {
    _id: ObjectId("64660fdea0a4910557e264fd"),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId("64661028a0a4910557e264fe"),
    name: 'Portland',
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.drop()
true
learn> show collections
unicorns
learn>
```

Рисунок 23 – Выполнение запроса 8.2.13

## 5 Ссылки и работа с индексами

### 5.1 Ссылки в БД

Задача: создать коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание; включить для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания; проверить содержание коллекции единорогов.

Коллекция полей:

```
learn> db.areas.find()
[
  {
    _id: 'CA',
    name: 'chamomile area',
    description: 'area with chamomiles'
  },
  {
    _id: 'SA',
    name: 'sunflowers area',
    description: 'area with sunlowers'
  }
]
```

Рисунок 24 – Коллекция полей

Пример добавления полей:

```
learn> db.unicorns.updateMany({name: {$in: ['Pilot', 'Ayna']}}, {$set: {area: {$ref: 'areas', $id: 'CA'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

Рисунок 25 – Пример добавления

Единороги с полями:

```
learn> db.unicorns.find({area: {$exists: true}}, {_id:0, name:1, area:1})
[
  { name: 'Aurora', area: DBRef("areas", 'SA') },
  { name: 'Ayna', area: DBRef("areas", 'CA') },
  { name: 'Pilot', area: DBRef("areas", 'CA') },
  { name: 'Nimue', area: DBRef("areas", 'SA') }
]
```

Рисунок 26 – Единороги с полями

## 5.2 Настройка индексов

Задача: проверить, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique.

```
db.unicorns.ensureIndex({"name": 1}, {"unique": true})
```

Выполнение запроса:

```
learn> db.unicorns.ensureIndex({"name": 1}, {"unique": true})
[ 'name_1' ]: 43,
learn>
```

Рисунок 27 – Выполнение запроса 8.3.2

## 5.3 Управление индексами

Задача: получить информацию о всех индексах коллекции единорогов; удалить все индексы, кроме индекса идентификатора, попытаться удалить индекс для идентификатора.

```
db.unicorns.getIndexes()
db.unicorns.dropIndex('name_1')
```

Выполнение запроса:

```
learn> db.unicorns.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
learn> db.unicorns.dropIndex('name_1')
{ nIndexesWas: 2, ok: 1 }
learn> db.unicorns.dropIndex('_id_')
MongoServerError: cannot drop _id index
learn> db.unicorns.getIndexes()
[ { v: 2, key: { _id: 1 }, name: '_id_' } ]
learn> _
```

Рисунок 28 – Выполнение запроса 8.3.3

## 5.4 План запроса

Задача: создать объемную коллекцию; выбрать последних четыре документа; проанализировать план и время выполнения запроса; создать индекс для value; получить информацию о всех индексах; проанализировать план выполнения запроса с индексами; сравнить время выполнения запросов с индексами и без.

```
db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
```

Время запроса без индекса:

```
executionTimeMillis: 1,
```

Рисунок 29 – Время запроса без индекса

Создание индекса:

```
learn> db.numbers.ensureIndex({"value": 1})
[ 'value_1' ]
learn> db.numbers.getIndex()
TypeError: db.numbers.getIndex is not a function
learn> db.numbers.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { value: 1 }, name: 'value_1' }
]
```

Рисунок 30 – Создание индекса

Время запроса с индексом:

```
executionTimeMillis: 0,
```

Рисунок 31 – Время запроса с индексом

Вывод: индексы уменьшают время запросов на очень больших коллекциях, следовательно, запросы с индексами эффективнее.

## **ЗАКЛЮЧЕНИЕ**

В ходе данной лабораторной работы были приобретены навыки работы с CRUD-операциями в базе данных MONGODB.

В результате данной лабораторной работы были созданы коллекции и различные запросы. Некоторые запросы были выполнены с использованием JavaScript. Также было освоено изменение данных в коллекциях в зависимости от целей, удаление документов, ключей и всей коллекции. Были установлены ссылки с одной коллекции на другую. Была проведена работа с созданием, удалением и использованием индексов и был сделан вывод, что запросы с индексами эффективнее запросов без них.