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 №5.2

«РАБОТА С БД В СУБД MONGODB»

Автор: Ивенкова Елизавета Дмитриевна

Группа: K32422

Преподаватель: Говорова М. М.

Санкт-Петербург

2023

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

Практическое задание: привести результаты выполнения практических заданий (номер задания, формулировка, команда, лог (скриншот) результата, вывод (при необходимости)).

Выполнение работы:

Для выполнения работы было предложено воспользоваться текущей версией ПО (6.0.6), однако из-за возникших при установке проблем в работе была использована версия 5.0.17.

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

ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

Практическое задание 8.1.1:

1. *Создайте базу данных learn.*

```
> use learn
switched to db learn
```

2. *Заполните коллекцию единорогов unicorns:*

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

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

```
{name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
```

```
> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
WriteResult({ "nInserted" : 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({ "nInserted" : 1 })
>
```

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

```
> db.unicorns.find({})
{ "_id" : ObjectId("6467bae52b68d0d9e4eaac38"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6467baeb2b68d0d9e4eaac39"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6467baf12b68d0d9e4eaac3a"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6467bafc2b68d0d9e4eaac3b"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6467bb032b68d0d9e4eaac3c"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6467bb0a2b68d0d9e4eaac3d"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733,
 "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6467bb0f2b68d0d9e4eaac3e"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6467bb162b68d0d9e4eaac3f"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "
gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6467bb1c2b68d0d9e4eaac40"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6467bb232b68d0d9e4eaac41"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6467bb292b68d0d9e4eaac42"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6467bb562b68d0d9e4eaac43"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165 }
>
```

ВЫБОРКА ДАННЫХ ИЗ БД

Практическое задание 8.1.2:

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

```
> db.unicorns.find({gender: 'm'}).limit(3).sort({name: 1})
{ "_id" : ObjectId("6467bb562b68d0d9e4eaac43"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6467bae52b68d0d9e4eaac38"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6467bb0f2b68d0d9e4eaac3e"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
>
>
>
> db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
{ "_id" : ObjectId("6467baeb2b68d0d9e4eaac39"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6467bb0a2b68d0d9e4eaac3d"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733,
 "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6467bb1c2b68d0d9e4eaac40"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
>
>
```

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

```
> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
{ "_id" : ObjectId("6467baeb2b68d0d9e4eaac39"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
>
>
> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
        "_id" : ObjectId("6467baeb2b68d0d9e4eaac39"),
        "name" : "Aurora",
        "loves" : [
                "carrot",
                "grape"
        ],
        "weight" : 450,
        "gender" : "f",
        "vampires" : 43
}
```

Практическое задание 8.1.3:

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

```
> db.unicorns.find({gender: 'm'}, {gender:0, loves: 0})
{ "_id" : ObjectId("6467bae52b68d0d9e4eaac38"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("6467baf12b68d0d9e4eaac3a"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
{ "_id" : ObjectId("6467bafc2b68d0d9e4eaac3b"), "name" : "Roooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("6467bb0f2b68d0d9e4eaac3e"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("6467bb162b68d0d9e4eaac3f"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("6467bb232b68d0d9e4eaac41"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("6467bb562b68d0d9e4eaac43"), "name" : "Dunx", "weight" : 704, "vampires" : 165 }
>
```

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find({}).sort({$natural: -1})
{ "_id" : ObjectId("6467bb562b68d0d9e4eaac43"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6467bb292b68d0d9e4eaac42"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6467bb232b68d0d9e4eaac41"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6467bb1c2b68d0d9e4eaac40"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6467bb162b68d0d9e4eaac3f"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "
gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6467bb0f2b68d0d9e4eaac3e"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6467bb0a2b68d0d9e4eaac3d"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733,
 "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6467bb032b68d0d9e4eaac3c"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6467bafc2b68d0d9e4eaac3b"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6467baf12b68d0d9e4eaac3a"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6467baeb2b68d0d9e4eaac39"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6467bae52b68d0d9e4eaac38"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
>
```

Практическое задание 8.1.5:

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

```
> 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" : "Roooooodles", "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 }
>
```

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 8.1.6:

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

```
> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 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" }
>
```

Практическое задание 8.1.7:

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

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

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ `vampires`.

```
> db.unicorns.find({vampires: {$exists: false}})
{ "_id" : ObjectId("6467bb292b68d0d9e4eaac42"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
>
```

Практическое задание 8.1.9:

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

```
> 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" : "Roooooodles", "loves" : [ "apple" ] }
{ "name" : "Unicrom", "loves" : [ "energon" ] }
>
```

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

ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 8.2.1:

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

```
{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"}}
```

```
> db.towns.insert({name: 'Punxsutawney', population: 6200, last_sensus: ISODate('2008-01-31'), famous_for: [''], mayo
r: {name: 'Jim Wehrle'}})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: 'New York', population: 22200000, last_sensus: ISODate('2009-07-31'), famous_for: ['status o
f liberty', 'food'], mayor: {name: 'Michael Bloomberg', party: 'I'}})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: 'Portland', population: 528000, last_sensus: ISODate('2009-07-20'), famous_for: ['beer', 'fo
od'], mayor: {name: 'Sam Adams', party: 'D'}})
WriteResult({ "nInserted" : 1 })
>
```

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

```
> db.towns.find({'mayor.party': 'I'}, {name: 1, mayor: 1, _id: 0})
{ "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
>
```

3. *Сформировать запрос, который возвращает список беспартийных мэров (`party` отсутствует). Вывести только название города и информацию о мэре.*

```
>
> db.towns.find({'mayor.party': {$exists: false}}, {name: 1, mayor: 1, _id: 0})
{ "name" : "Punxsutawney", "mayor" : { "name" : "Jim Wehrle" } }
>
```

КУРСОРЫ

Практическое задание 8.2.2:

1. Сформировать функцию для вывода списка самцов единорогов.

```
> male = function() {return this.gender == 'm'}
function() {return this.gender == 'm'}
> _
```

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

3. *Вывести результат, используя forEach.*

```
> var cursor = db.unicorns.find(male).limit(2).sort({name: 1});
> cursor.forEach( function(obj) {print(obj.name);} );
Dunx
Horny
>
```

4. *Содержание коллекции единорогов* `unicorns`*:*

```
db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});

db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});

db.unicorns.insert({name: 'Roooooodles', 44), loves: ['apple'], weight: 575, gender: 'm', vampires: 99});

db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});

db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});

db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
```

```
db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601,
gender: 'f', vampires: 33});

db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650,
gender: 'm', vampires: 54});

db.unicorns.insert ({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540,
gender: 'f'});

db.unicorns.insert ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704,
gender: 'm', vampires: 165})
```

АГРЕГИРОВАННЫЕ ЗАПРОСЫ

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 8.2.4:

Вывести список предпочтений.

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

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

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

РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 8.2.6:

1. *Выполнить команду:*

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

```
> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
>
```

2. *Проверить содержимое коллекции* `unicorns`.

```
>
> db.unicorns.find({})
{ "_id" : ObjectId("6464c2179ffb8f6d1fa2c2d2"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6464c2789ffb8f6d1fa2c2d3"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6464c2b29ffb8f6d1fa2c2d4"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6464c3cc9ffb8f6d1fa2c2d5"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6464c3db9ffb8f6d1fa2c2d6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6464c3e99ffb8f6d1fa2c2d7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733,
 "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6464c3f79ffb8f6d1fa2c2d8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6464c4079ffb8f6d1fa2c2d9"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "
gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6464c4219ffb8f6d1fa2c2da"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6464c42b9ffb8f6d1fa2c2db"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6464c4369ffb8f6d1fa2c2dc"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6464c47b9ffb8f6d1fa2c2dd"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6466758a9ffb8f6d1fa2c2e1"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "
m" }
>
```

Практическое задание 8.2.7:

1. *Для самки единорога* `Ayna` *внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.*

2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.update({name: 'Ayna'}, {"name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender"
: "f", "vampires" : 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({})
{ "_id" : ObjectId("6464c2179ffb8f6d1fa2c2d2"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6464c2789ffb8f6d1fa2c2d3"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6464c2b29ffb8f6d1fa2c2d4"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6464c3cc9ffb8f6d1fa2c2d5"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6464c3db9ffb8f6d1fa2c2d6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6464c3e99ffb8f6d1fa2c2d7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800,
 "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6464c3f79ffb8f6d1fa2c2d8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6464c4079ffb8f6d1fa2c2d9"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "
gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6464c4219ffb8f6d1fa2c2da"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6464c42b9ffb8f6d1fa2c2db"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6464c4369ffb8f6d1fa2c2dc"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6464c47b9ffb8f6d1fa2c2dd"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6466758a9ffb8f6d1fa2c2e1"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "
m" }
> _
```

Практическое задание 8.2.8:

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

2. *Проверить содержимое коллекции* `unicorns`.

```
>
> db.unicorns.update({name: 'Raleigh'}, {$set: {loves: ['redbull']}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({})
{ "_id" : ObjectId("6464c2179ffb8f6d1fa2c2d2"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6464c2789ffb8f6d1fa2c2d3"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6464c2b29ffb8f6d1fa2c2d4"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6464c3cc9ffb8f6d1fa2c2d5"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6464c3db9ffb8f6d1fa2c2d6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6464c3e99ffb8f6d1fa2c2d7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800,
 "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6464c3f79ffb8f6d1fa2c2d8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6464c4079ffb8f6d1fa2c2d9"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender"
 : "m", "vampires" : 2 }
{ "_id" : ObjectId("6464c4219ffb8f6d1fa2c2da"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6464c42b9ffb8f6d1fa2c2db"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6464c4369ffb8f6d1fa2c2dc"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6464c47b9ffb8f6d1fa2c2dd"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6466758a9ffb8f6d1fa2c2e1"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "
m" }
> _
```

Практическое задание 8.2.9:

1. *Всем самцам единорогов увеличить количество убитых вапмиров на 5.*

2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.update({gender:'m'}, {$inc: {vampires: 5}}, {multi: true})
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
> db.unicorns.find()
{ "_id" : ObjectId("6464c2179ffb8f6d1fa2c2d2"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6464c2789ffb8f6d1fa2c2d3"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6464c2b29ffb8f6d1fa2c2d4"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6464c3cc9ffb8f6d1fa2c2d5"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6464c3db9ffb8f6d1fa2c2d6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6464c3e99ffb8f6d1fa2c2d7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800,
 "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6464c3f79ffb8f6d1fa2c2d8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6464c4079ffb8f6d1fa2c2d9"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender"
 : "m", "vampires" : 7 }
{ "_id" : ObjectId("6464c4219ffb8f6d1fa2c2da"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6464c42b9ffb8f6d1fa2c2db"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6464c4369ffb8f6d1fa2c2dc"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6464c47b9ffb8f6d1fa2c2dd"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("6466758a9ffb8f6d1fa2c2e1"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "
m", "vampires" : 5 }
> _
```

Практическое задание 8.2.10:

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

2. *Проверить содержимое коллекции* `towns`.

```
>
> db.towns.update({name: 'Portland'}, {'$unset': {'mayor.party': 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find({})
{ "_id" : ObjectId("646529859ffb8f6d1fa2c2de"), "name" : "Punxsutawney", "population" : 6200, "last_sensus" : ISODate
("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("646529db9ffb8f6d1fa2c2df"), "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("64652a199ffb8f6d1fa2c2e0"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("
2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams" } }
>
```

Практическое задание 8.2.11:

1. *Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.*

2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.update({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({})
{ "_id" : ObjectId("6464c2179ffb8f6d1fa2c2d2"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6464c2789ffb8f6d1fa2c2d3"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6464c2b29ffb8f6d1fa2c2d4"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6464c3cc9ffb8f6d1fa2c2d5"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6464c3db9ffb8f6d1fa2c2d6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6464c3e99ffb8f6d1fa2c2d7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800,
 "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6464c3f79ffb8f6d1fa2c2d8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6464c4079ffb8f6d1fa2c2d9"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender"
 : "m", "vampires" : 7 }
{ "_id" : ObjectId("6464c4219ffb8f6d1fa2c2da"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6464c42b9ffb8f6d1fa2c2db"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "
weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6464c4369ffb8f6d1fa2c2dc"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6464c47b9ffb8f6d1fa2c2dd"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("6466758a9ffb8f6d1fa2c2e1"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "
m", "vampires" : 5 }
>
```

Практическое задание 8.2.12:

1. *Изменить информацию о самке единорога* `Aurora`*: теперь она любит еще и сахар, и лимоны.*

2. *Проверить содержимое коллекции* `unicorns`*.*

```
> db.unicorns.update({name: 'Aurora'}, {$addToSet: {loves: {$each: ['sugar', 'lemon']}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({})
{ "_id" : ObjectId("6464c2179ffb8f6d1fa2c2d2"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6464c2789ffb8f6d1fa2c2d3"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ],
 "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6464c2b29ffb8f6d1fa2c2d4"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6464c3cc9ffb8f6d1fa2c2d5"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6464c3db9ffb8f6d1fa2c2d6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6464c3e99ffb8f6d1fa2c2d7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800,
 "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6464c3f79ffb8f6d1fa2c2d8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6464c4079ffb8f6d1fa2c2d9"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender"
 : "m", "vampires" : 7 }
{ "_id" : ObjectId("6464c4219ffb8f6d1fa2c2da"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6464c42b9ffb8f6d1fa2c2db"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "
weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6464c4369ffb8f6d1fa2c2dc"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f" }
{ "_id" : ObjectId("6464c47b9ffb8f6d1fa2c2dd"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("6466758a9ffb8f6d1fa2c2e1"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "
m", "vampires" : 5 }
>
```

УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

Практическое задание 8.2.13:

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

```
{name: "Punxsutawney ", population: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil the groundhog"], mayor: {name: "Jim Wehrle"}}

{name: "New York", population: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["status of liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}}

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

```
> db.towns.insert({name: "Punxsutawney ", population: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil th
e groundhog"], mayor: {name: "Jim Wehrle"}})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "New York", population: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["status o
f liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Portland", population: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "fo
od"], mayor: {name: "Sam Adams", party: "D"}})
WriteResult({ "nInserted" : 1 })
>
> db.towns.find({})
{ "_id" : ObjectId("646a3313d6214d38ba757f4c"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" : ISODat
e("2008-01-31T00:00:00Z"), "famous_for" : [ "phil the groundhog" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("646a3320d6214d38ba757f4d"), "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("646a332ed6214d38ba757f4e"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("
2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
>
```

2. *Удалите документы с беспартийными мэрами.*

3. *Проверьте содержание коллекции.*

```
> db.towns.remove({'mayor.party': {$exists: 0}})
WriteResult({ "nRemoved" : 1 })
> db.towns.find({})
{ "_id" : ObjectId("646529db9ffb8f6d1fa2c2df"), "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("64652a199ffb8f6d1fa2c2e0"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("
2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
>
```

4. *Очистите коллекцию.*

5. *Просмотрите список доступных коллекций.*

```
> db.towns.drop()
true
> show collections
unicorns
>
```

ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

ССЫЛКИ В БД

Практическое задание 8.3.1:

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

```
> db.areas.insert('_id': 'rus', 'name': 'Russia', 'description': 'Big country with diverse problems')
uncaught exception: SyntaxError: missing ) after argument list :
@(shell):1:21
> db.areas.insert({'_id': 'rus', 'name': 'Russia', 'description': 'Big country with diverse problems'})
WriteResult({ "nInserted" : 1 })
> db.areas.insert({'_id': 'fpk', 'name': 'Freepik', 'description': 'A website where images with unicorns are stored'}
)
WriteResult({ "nInserted" : 1 })
> db.areas.insert({'_id': 'rnb', 'name': 'Rainbow', 'description': 'The most probable habitat of mystical creatures'}
)
WriteResult({ "nInserted" : 1 })
```

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

3. *Проверьте содержание коллекции едиорогов.*

```
> db.unicorns.update({name: 'Aurora'}, {$set: {area: {$ref: 'areas', $id: 'rnb'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.unicorns.update({name: 'Raleigh'}, {$set: {area: {$ref: 'areas', $id: 'rus'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.unicorns.update({name: 'Dunx'}, {$set: {area: {$ref: 'areas', $id: 'rus'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.unicorns.update({name: 'Ayna'}, {$set: {area: {$ref: 'areas', $id: 'fpk'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.unicorns.update({name: 'Nimue'}, {$set: {area: {$ref: 'areas', $id: 'fpk'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.unicorns.find({})
{ "_id" : ObjectId("64678e059ffb8f6d1fa2c2e3"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "
gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("64678e1c9ffb8f6d1fa2c2e4"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "
gender" : "f", "vampires" : 43, "area" : DBRef("areas", "rnb") }
{ "_id" : ObjectId("64678e319ffb8f6d1fa2c2e5"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 98
4, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("64678e429ffb8f6d1fa2c2e6"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gende
r" : "m", "vampires" : 99 }
{ "_id" : ObjectId("64678e5c9ffb8f6d1fa2c2e7"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "we
ight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("64678e659ffb8f6d1fa2c2e8"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733,
 "gender" : "f", "vampires" : 40, "area" : DBRef("areas", "fpk") }
{ "_id" : ObjectId("64678e6a9ffb8f6d1fa2c2e9"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "ge
nder" : "m", "vampires" : 39 }
{ "_id" : ObjectId("64678e709ffb8f6d1fa2c2ea"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "
gender" : "m", "vampires" : 2, "area" : DBRef("areas", "rus") }
{ "_id" : ObjectId("64678e769ffb8f6d1fa2c2eb"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601,
 "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("64678e7c9ffb8f6d1fa2c2ec"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650
, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("64678e829ffb8f6d1fa2c2ed"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "g
ender" : "f", "area" : DBRef("areas", "fpk") }
{ "_id" : ObjectId("64678e939ffb8f6d1fa2c2ee"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704,
 "gender" : "m", "vampires" : 165, "area" : DBRef("areas", "rus") }
> _
```

4. *Содержание коллекции единорогов* `unicorns`*:*

```
db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});

db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});

db.unicorns.insert({name: 'Roooooodles', 44), loves: ['apple'], weight: 575, gender: 'm', vampires: 99});

db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});

db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});

db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});

db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});

db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});

db.unicorns.insert ({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});

db.unicorns.insert {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
```

НАСТРОЙКА ИНДЕКСОВ

Практическое задание 8.3.2:

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

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

Вывод: для коллекции можно задать индекс для ключа name с флагом unique, так как до этого в коллекции имена единорогов не повторялись.

2. *Содержание коллекции единорогов* `unicorns`*:*

```
db.unicorns.insert({name: 'Horny', dob: new Date(1992,2,13,7,47), loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insert({name: 'Aurora', dob: new Date(1991, 0, 24, 13, 0), loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});

db.unicorns.insert({name: 'Unicrom', dob: new Date(1973, 1, 9, 22, 10), loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});

db.unicorns.insert({name: 'Roooooodles', dob: new Date(1979, 7, 18, 18, 44), loves: ['apple'], weight: 575, gender: 'm', vampires: 99});

db.unicorns.insert({name: 'Solnara', dob: new Date(1985, 6, 4, 2, 1), loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

db.unicorns.insert({name:'Ayna', dob: new Date(1998, 2, 7, 8, 30), loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});

db.unicorns.insert({name:'Kenny', dob: new Date(1997, 6, 1, 10, 42), loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});

db.unicorns.insert({name: 'Raleigh', dob: new Date(2005, 4, 3, 0, 57), loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});

db.unicorns.insert({name: 'Leia', dob: new Date(2001, 9, 8, 14, 53), loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});

db.unicorns.insert({name: 'Pilot', dob: new Date(1997, 2, 1, 5, 3), loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});

db.unicorns.insert ({name: 'Nimue', dob: new Date(1999, 11, 20, 16, 15), loves: ['grape', 'carrot'], weight: 540, gender: 'f'});

db.unicorns.insert ({name: 'Dunx', dob: new Date(1976, 6, 18, 18, 18), loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
```

УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 8.3.3:

1. *Получите информацию о всех индексах коллекции* `unicorns`.

```
> db.unicorns.getIndexes()
[
        {
                "v" : 2,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_"
        },
        {
                "v" : 2,
                "key" : {
                        "name" : 1
                },
                "name" : "name_1",
                "unique" : true
        }
]
>
```

2. *Удалите все индексы, кроме индекса для идентификатора.*

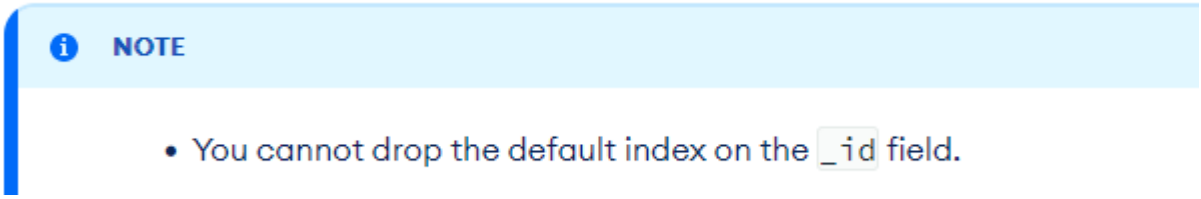
```
> db.unicorns.dropIndex('name_1')
{ "nIndexesWas" : 2, "ok" : 1 }
>
```

Индекс удаляется по своему имени, которое можно посмотреть после выполнения операции getIndexes(), а не по названию поля, которому соответствует индекс. Поэтому в качестве аргумента для dropIndex() используется строка ‘name_1’, соответствующая названию индекса для ключа name.

3. *Попытайтесь удалить индекс для идентификатора.*

```
> db.unicorns.dropIndex('_id_')
{
        "ok" : 0,
        "errmsg" : "cannot drop _id index",
        "code" : 72,
        "codeName" : "InvalidOptions"
}
>
```

Уникальный индекс для идентификатора _id удалить нельзя (см. соответствующее сообщение об ошибке). Об этом же говорит и официальная документация MongoDB:



ПЛАН ЗАПРОСА

Практическое задание 8.3.4:

1. *Создайте объемную коллекцию* `numbers`*, задействовав курсор:*

2. `for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}`

3. *Выберите последних четыре документа.*

```
>
> for (i = 0; i < 100000; i++) {db.numbers.insert({value: i})}
WriteResult({ "nInserted" : 1 })
>
> db.numbers.find({}).skip(99996)
{ "_id" : ObjectId("64679e2e81187664556270e7"), "value" : 99996 }
{ "_id" : ObjectId("64679e2e81187664556270e8"), "value" : 99997 }
{ "_id" : ObjectId("64679e2e81187664556270e9"), "value" : 99998 }
{ "_id" : ObjectId("64679e2e81187664556270ea"), "value" : 99999 }
>
>
```

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

```
>
> db.numbers.explain('executionStats').find({}).skip(99996)
{
        "explainVersion" : "1",
        "queryPlanner" : {
                "namespace" : "test.numbers",
                "indexFilterSet" : false,
                "parsedQuery" : {

                },
                "maxIndexedOrSolutionsReached" : false,
                "maxIndexedAndSolutionsReached" : false,
                "maxScansToExplodeReached" : false,
                "winningPlan" : {
                        "stage" : "SKIP",
                        "skipAmount" : 0,
                        "inputStage" : {
                                "stage" : "COLLSCAN",
                                "direction" : "forward"
                        }
                },
                "rejectedPlans" : [ ]
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 4,
                "executionTimeMillis" : 21,
                "totalKeysExamined" : 0,
                "totalDocsExamined" : 100000,
                "executionStages" : {
```

На выполнение запроса понадобилась **21 миллисекунда**. По примерным подсчетам по секундомеру, на создание коллекции понадобилось сильно больше времени, около 20 секунд.

5. *Создайте индекс для ключа* `value`.

```
> db.numbers.createIndex({value: 1})
{
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "createdCollectionAutomatically" : false,
        "ok" : 1
}
>
```

6. *Получите информацию о всех индексах коллекции* `numbers`.

```
> db.numbers.getIndexes()
[
        {
                "v" : 2,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_"
        },
        {
                "v" : 2,
                "key" : {
                        "value" : 1
                },
                "name" : "value_1"
        }
]
>
```

Индекс поля value автоматически наименован как ‘value_1’.

7. *Выполните запрос 2.*

```
> db.numbers.find({}).skip(99996)
{ "_id" : ObjectId("64679e2e81187664556270e7"), "value" : 99996 }
{ "_id" : ObjectId("64679e2e81187664556270e8"), "value" : 99997 }
{ "_id" : ObjectId("64679e2e81187664556270e9"), "value" : 99998 }
{ "_id" : ObjectId("64679e2e81187664556270ea"), "value" : 99999 }
>
```

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

На выполнение запроса с индексом, установленным на value, ушло **19 миллисекунд** (на 2 миллисекунды меньше, чем без индекса).

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

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

As you create indexes, consider the following behaviors of indexes:

- Each index requires at least 8 kB of data space.
- Adding an index has some negative performance impact for write operations. For collections with high write-to-read ratio, indexes are expensive since each insert must also update any indexes.
- Collections with high read-to-write ratio often benefit from additional indexes. Indexes do not affect un-indexed read operations.
- When active, each index consumes disk space and memory. This usage can be significant and should be tracked for capacity planning, especially for concerns over working set size.

Number of Indexes per Collection

A single collection can have *no more* than 64 indexes.

Выводы: в ходе выполнения данной работы были рассмотрены структура организации и хранения данных в СУБД MongoDB, основные операции над данными: вставка, редактирование, удаление, выборка с учетом условия, использование агрегирующих функций, сортировка, использование ссылок для связи полей и значений разных документов, использование функций и курсоров, а также просмотр статистики о произведенном запросе.

Итого, в БД MongoDB хранятся документы, имеющие уникальный идентификатор _id (генерируется СУБД в случае его неуказания при вставке данных), состоящие из пар ключ-значение (вложенные документы допускаются); документы образовывают коллекции. Было выяснено, что с помощью индексирования ключей в коллекции возможно выполнять запросы более эффективно.