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

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ №8

по теме: работа с БД с СУБД MongoDB
по дисциплине: Проектирование и реализация баз данных

Специальность:
09.03.03 Мобильные и сетевые технологии

Проверил:
Говорова М.М. _____
Дата: «08» июня 2021г.
Оценка _____

Выполнил:
студент группы К3241
Фоменко Иван

Санкт-Петербург 2021 г.

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

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4.4.

ПРАКТИЧЕСКАЯ ЧАСТЬ

8.1.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

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.*

```
> use learn
switched to db learn
> 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 })
```

```
> db.unicorns.insert(document)
WriteResult({ "nInserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60bf30a13737591998b98cfe"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60bf30af3737591998b98cff"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60bf30d53737591998b98d02"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60bf30f03737591998b98d04"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
>
```

8.2.2. 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'}).sort({name: 1})
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

```
> db.unicorns.find({gender:'f'}).limit(3).sort({name: 1});
{ "_id" : ObjectId("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60bf30af3737591998b98cff"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60bf30d53737591998b98d02"), "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("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
> db.unicorns.findOne({gender:'f', loves: "carrot"});
{
        "_id" : ObjectId("60bf307d3737591998b98cfb"),
        "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("60bf305b3737591998b98cfa"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "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("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60bf30f03737591998b98d04"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60bf30d53737591998b98d02"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60bf30af3737591998b98cff"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60bf30a13737591998b98cfe"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "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({}, {_id: 0, loves:{$slice: 1}});
{ "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 }
>
```

8.2.3. 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("60bf30f03737591998b98d04"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>
```

Практическое задание 8.1.9:

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

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

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

8.2.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 8.2.1:

1. Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.insert({name: "New York", populatiuon: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["status of liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}}
... );
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: [""], mayor: {name: "Jim Wehrle" }}
... );
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Portland", populatiuon: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: {name: "Sam Adams", party: "D"}}
... );
WriteResult({ "nInserted" : 1 })
> db.towns.find();
{ "_id" : ObjectId("60bf49b43737591998b98d06"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food"
], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("60bf4a1c3737591998b98d07"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" :
"Jim Wehrle" } }
{ "_id" : ObjectId("60bf4a5e3737591998b98d08"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : {
"name" : "Sam Adams", "party" : "D" } }
>
```

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" } }
>
```

8.2.2 ИСПОЛЬЗОВАНИЕ JAVASCRIPT

8.2.3 КУРСОРЫ

Практическое задание 8.2.2:

1. Сформировать функцию для вывода списка самцов единорогов.

```
> fn = 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.

```
> var cursor = db.unicorns.find(fn); null;
null
> cursor.limit(2).sort({name: 1})
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
>
```

3. Вывести результат, используя forEach.

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

8.2.4 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" : "gender", "count" : 12 }
>
```

8.2.5 РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 8.2.6:

1. Выполнить команд

Проверить содержимое коллекции `users`.

```
> db.unicorns.save({name: 'Barny', loves: ['grape'],
... weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
>
> db.unicorns.find();
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60bf30a13737591998b98cfe"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60bf30af3737591998b98cff"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60bf30d53737591998b98d02"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60bf30f03737591998b98d04"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60bf56bf3737591998b98d09"), "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 `users`.

```
> db.unicorns.update({name: "Ayna"}, {name: "Ayna", loves: ["strawberry", "lemon"], gender: "f", weight: 800, vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Ayna"})
{ "_id" : ObjectId("60bf30af3737591998b98cff"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "gender" : "f", "weight" : 800, "vampires" : 51 }
> _
```

Практическое задание 8.2.8:

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

2. Проверить содержимое коллекции `users`.

```
> db.unicorns.update({name: "Raleigh", gender: "m"}, {$set: {loves: ["redbull"]}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Raleigh"})
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
>
```

Практическое задание 8.2.9:

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

2. Проверить содержимое коллекции `users`.

```
> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true})
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
> db.unicorns.find({gender: "m"})
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("60bf56bf3737591998b98d09"), "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("60bf49b43737591998b98d06"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name"
: "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("60bf4a1c3737591998b98d07"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("60bf4a5e3737591998b98d08"), "name" : "Portland", "populatiuon" : 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 `users`.

```
> db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Pilot"})
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
>
```

Практическое задание 8.2.12:

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

2. Проверить содержимое коллекции `users`*.*

```
> db.unicorns.update({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Aurora"})
{ "_id" : ObjectId("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
>
```

8.2.6 УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

Практическое задание 8.2.13:

1. Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.insert({name: "Punxsutawney ", popujatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil the groundhog"], mayor: {name: "Jim Wehrle" }}
... )
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "New York", popujatiuon: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["status of liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}}
... )
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Portland", popujatiuon: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: {name: "Sam Adams", party: "D"}}
... )
WriteResult({ "nInserted" : 1 })
> db.towns.find()
{ "_id" : ObjectId("60bf5cd63737591998b98d0a"), "name" : "Punxsutawney ", "popujatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "phil the groundhog" ], "mayor" : { "name" : "Jim
 Wehrle" } }
{ "_id" : ObjectId("60bf5cff3737591998b98d0b"), "name" : "New York", "popujatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name"
: "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("60bf5d273737591998b98d0c"), "name" : "Portland", "popujatiuon" : 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.

```
> db.towns.remove({"mayor.party": {$exists: false}})
WriteResult({ "nRemoved" : 1 })
> db.towns.find()
{ "_id" : ObjectId("60bf5cff3737591998b98d0b"), "name" : "New York", "popujatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name"
: "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("60bf5d273737591998b98d0c"), "name" : "Portland", "popujatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams",
"party" : "D" } }
>
```

3. Проверьте содержание коллекции.

4. Очистите коллекцию.

```
> db.towns.remove({})
WriteResult({ "nRemoved" : 2 })
>
```

5. Просмотрите список доступных коллекций.

```
> show collections
towns
unicorns
>
```

8.3 ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

8.3.1 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: "Ice", name: "Iceland", desc: "In the world"})
WriteResult({ "nInserted" : 1 })
> db.areas.insert({_id: "Jungle", name: "Jamaika jungle", desc: "In the world"})
WriteResult({ "nInserted" : 1 })
>
```

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

```
> db.unicorns.update({name: "Horny"}, {$set:{area:{$ref:"areas", $id:"Ice"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Pilot"}, {$set:{area:{$ref:"areas", $id:"Jungle"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Ayna"}, {$set:{area:{$ref:"areas", $id:"Jungle"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

3. Проверьте содержание коллекции единорогов.

```
> db.unicorns.find()
{ "_id" : ObjectId("60bf305b3737591998b98cfa"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68, "area" : DBRef("areas", "Ice") }
{ "_id" : ObjectId("60bf307d3737591998b98cfb"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60bf308a3737591998b98cfc"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60bf30953737591998b98cfd"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("60bf30a13737591998b98cfe"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60bf30af3737591998b98cff"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "gender" : "f", "weight" : 800, "vampires" : 51, "area" : DBRef("areas", "Jungle") }
{ "_id" : ObjectId("60bf30be3737591998b98d00"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("60bf30c93737591998b98d01"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("60bf30d53737591998b98d02"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60bf30e43737591998b98d03"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59, "area" : DBRef("areas", "Jungle") }
{ "_id" : ObjectId("60bf30f03737591998b98d04"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60bf356d3737591998b98d05"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("60bf56bf3737591998b98d09"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
>
```

8.3.2 НАСТРОЙКА ИНДЕКСОВ

Практическое задание 8.3.2:

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

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

8.3.3 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,
                "unique" : true,
                "key" : {
                        "name" : 1
                },
                "name" : "name_1"
        }
]
>
```

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

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

3. Попытайтесь удалить индекс для идентификатора.

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

8.3.4 ПЛАН ЗАПРОСА

Практическое задание 8.3.4:

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

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

2. Выберите последних четыре документа.

```
> db.numbers.find({value: {$gte: 99996}})
{ "_id" : ObjectId("60bf6fdd3737591998bb13a9"), "value" : 99996 }
{ "_id" : ObjectId("60bf6fdd3737591998bb13aa"), "value" : 99997 }
{ "_id" : ObjectId("60bf6fdd3737591998bb13ab"), "value" : 99998 }
{ "_id" : ObjectId("60bf6fdd3737591998bb13ac"), "value" : 99999 }
>
```

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

4. Создайте индекс для ключа `value`.

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

5. Получите информацию о всех индексах коллекции `numbres`.

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

6. Выполните запрос 2.

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

```
> db.numbers.explain("executionStats").find({value: {$gte: 99996}})
{
        "queryPlanner" : {
                "plannerVersion" : 1,
                "namespace" : "learn.numbers",
                "indexFilterSet" : false,
                "parsedQuery" : {
                        "value" : {
                                "$gte" : 99996
                        }
                },
                "winningPlan" : {
                        "stage" : "FETCH",
                        "inputStage" : {
                                "stage" : "IXSCAN",
                                "keyPattern" : {
                                        "value" : 1
                                },
                                "indexName" : "value_1",
                                "isMultiKey" : false,
                                "multiKeyPaths" : {
                                        "value" : [ ]
                                },
                                "isUnique" : true,
                                "isSparse" : false,
                                "isPartial" : false,
                                "indexVersion" : 2,
                                "direction" : "forward",
                                "indexBounds" : {
                                        "value" : [
                                                "[99996.0, inf.0]"
                                        ]
                                }
                        }
                },
                "rejectedPlans" : [ ]
        },
```

```
},
"executionStats" : {
        "executionSuccess" : true,
        "nReturned" : 4,
        "executionTimeMillis" : 17,
        "totalKeysExamined" : 4,
        "totalDocsExamined" : 4,
        "executionStages" : {
                "stage" : "FETCH",
                "nReturned" : 4,
                "executionTimeMillisEstimate" : 0,
                "works" : 5,
                "advanced" : 4,
                "needTime" : 0,
                "needYield" : 0,
                "saveState" : 0,
                "restoreState" : 0,
                "isEOF" : 1,
                "docsExamined" : 4,
                "alreadyHasObj" : 0,
                "inputStage" : {
                        "stage" : "IXSCAN",
                        "nReturned" : 4,
                        "executionTimeMillisEstimate" : 0,
                        "works" : 5,
                        "advanced" : 4,
                        "needTime" : 0,
                        "needYield" : 0,
                        "saveState" : 0,
                        "restoreState" : 0,
                        "isEOF" : 1,
                        "keyPattern" : {
                                "value" : 1
                        },
                        "indexName" : "value_1",
                        "isMultiKey" : false,
                        "multiKeyPaths" : {
                                "value" : [ ]
                        },
                        "isUnique" : true,
                        "isSparse" : false,
                        "isPartial" : false,
                        "indexVersion" : 2,
                        "direction" : "forward",
                        "indexBounds" : {
                                "value" : [
                                        "[99996.0, inf.0]"
                                ]
                        },
                        "keysExamined" : 4,
                        "seeks" : 1,
                        "dupsTested" : 0,
                        "dupsDropped" : 0
                }
        }
```

8. Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективно
После добавления индекса executionTimeMillis изменилось с 60 до 17 , значит с индексом в разы быстрее

Вывод:

Были освоены методы работы с БД в MongoDB