Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
“Национальный исследовательский университет ИТМО”

Факультет инфокоммуникационных технологий

ЛАБОРАТОРНАЯ РАБОТА №5

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

Выполнил студент:
Траоре Мамуду
Группа №K3241
Проверила:
Говорова Марина Михайловна

Санкт-Петербург
2022

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.

Задание 8.1.1

1) Создать базу данных learn и коллекцию “unicorns” .

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

```
> db.createCollection("unicorns")
{ "ok" : 1 }
>
```

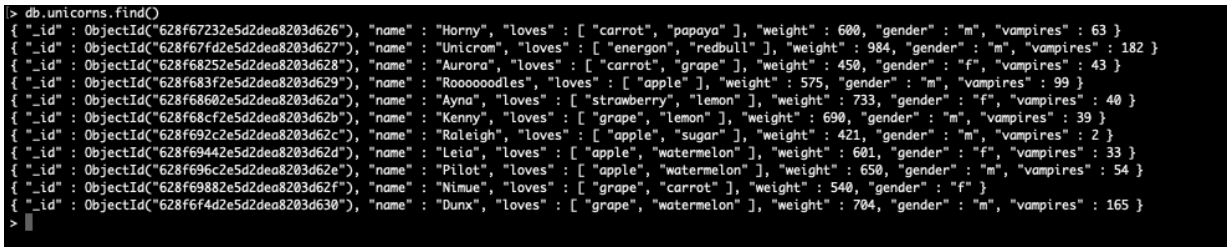
2) Заполнить коллекцию единорогов unicorns:

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

3) Используя второй способ, вставим в коллекцию единорогов документ:

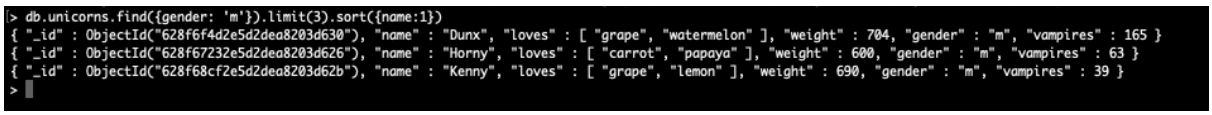
```
> doc = ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
{
        "name" : "Dunx",
        "loves" : [
                "grape",
                "watermelon"
        ],
        "weight" : 704,
        "gender" : "m",
        "vampires" : 165
}
> db.unicorns.insert(doc)
WriteResult({ "nInserted" : 1 })
>
```

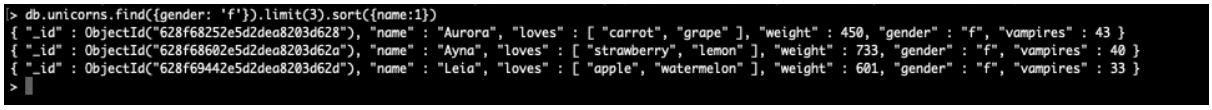
4) Проверим содержимое коллекции с помощью метода find.



Задание 8.1.2

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

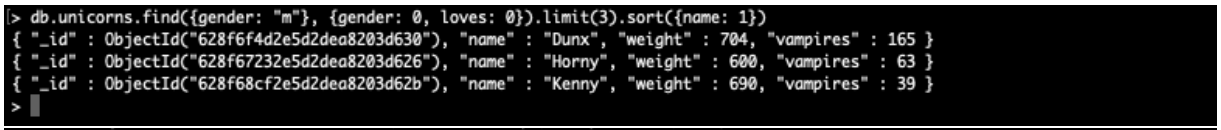


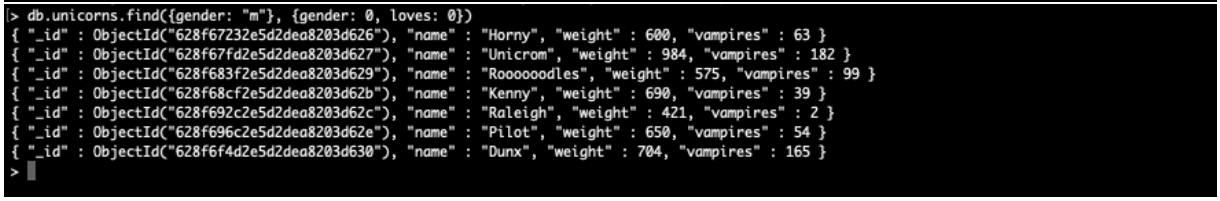


2) Найдем всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций findOne и limit.

Задание 8.1.3

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





Задание 8.1.4

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural: -1})
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f6988e5d2dea8203d62f"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f6944e5d2dea8203d62d"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f68cf2e5d2dea8203d62b"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f6860e5d2dea8203d62a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f683f2e5d2dea8203d629"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f68252e5d2dea8203d628"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f67fd2e5d2dea8203d627"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
>
```

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" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "name" : "Aurora", "loves" : [ "carrot" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "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 }
>
```

Задание 8.1.6

Вывести все занчения где вес больше 500 и меньше 700, а также скрыть id.

```
> db.unicorns.find({gender: "f", weight: {$gte: 500, $lte: 700}}, {_id: 0})
{ "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>
```

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", "v
mpires" : 39 }
```

Задание 8.1.8

Найти всех единорогов, не имеющих ключ vampires.

```
> db.unicorns.find({vampires: {$exists: false}})
{ "_id" : ObjectId("628f69882e5d2dea8203d62f"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>
```

Задание 8.1.9

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

```
> db.unicorns.find({}, {loves: {$slice: 1}}).sort({name: 1})
{ "_id" : ObjectId("628f68252e5d2dea8203d628"), "name" : "Aurora", "loves" : [ "carrot" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f68602e5d2dea8203d62a"), "name" : "Ayna", "loves" : [ "strawberry" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "name" : "Horny", "loves" : [ "carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f68cf2e5d2dea8203d62b"), "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f69442e5d2dea8203d62d"), "name" : "Leia", "loves" : [ "apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f69882e5d2dea8203d62f"), "name" : "Nimue", "loves" : [ "grape" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f683f2e5d2dea8203d629"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f67fd2e5d2dea8203d627"), "name" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
>
```

Задание 8.2.1.

1) Создайте коллекцию towns и заполнить

```
> db.createCollection('towns')
{ "ok" : 1 }
>
```

```
> 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: "Punxsutawwney", populatiuon: 6200, las_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 })
>
```

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"}, {mayor: true, name: true, _id: false})
{ "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: true, _id: false, mayor: true})
{ "name" : "Punxsutawwney", "mayor" : { "name" : "Jim Wehrle" } }
>
```

Задание 8.2.2.

1) Сформировать функцию для вывода списка самцов единорогов.

```
> fn = function () {return this.gender=="m";}
function () {return this.gender=="m";}
>
```

```
> db.unicorns.find(fn)
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f67fd2e5d2dea8203d627"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f683f2e5d2dea8203d629"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f68cf2e5d2dea8203d62b"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
>
```

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

```
> var cursor = db.unicorns.find(fn);null;null;
null
> cursor.limit(2).sort({name: 1})
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "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
>
```

Задание 8.2.3.

```
> db.unicorns.find({gender: "f", weight: {$gte: 500, $lte:600}}).count()
1
>
```

Задание 8.2.4

Вывести список предпочтений.

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

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" : 4 }
>
```

Задание 8.2.6

Выполнена команда из задания.

```
> db.unicorns.save({name: 'Barny', loves: ['grape'],
... weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f67fd2e5d2dea8203d627"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f68252e5d2dea8203d628"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f683f2e5d2dea8203d629"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f68602e5d2dea8203d62a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f68cf2e5d2dea8203d62b"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f69442e5d2dea8203d62d"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f69882e5d2dea8203d62f"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f95e62e5d2dea8203d631"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
>
```

Задание 8.2.7

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

```
> db.unicorns.update({name: "Anyna"}, {name: "Ayna", weight: 800, vampires: 51})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

```
{ "_id" : ObjectId("628f68602e5d2dea8203d62a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
```

Задание 8.2.8

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

```
> db.unicorns.update({name: "Raleigh", gender: "m"}, {$set: {loves: ["redbull"]}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

```
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
```

Задание 8.2.9

Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("628f67fd2e5d2dea8203d627"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f68252e5d2dea8203d628"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f683f2e5d2dea8203d629"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f68602e5d2dea8203d62a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f68cf2e5d2dea8203d62b"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f69442e5d2dea8203d62d"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f69882e5d2dea8203d62f"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f95e62e5d2dea8203d631"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
>
```

Задание 8.2.10

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

```
> db.towns.update({name: "Portland"}, {$unset: {"mayor.party": 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find()
{ "_id" : ObjectId("628ff0172e5d2dea8203d632"), "name" : "Punxsutawmney", "populatiuon" : 6200, "las_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("628ff1d82e5d2dea8203d633"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams" } }
{ "_id" : ObjectId("628ffa182e5d2dea8203d634"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "par
ty" : "I" } }
>
```

Задание 8.2.11

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

```
> db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Pilot"})
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
>
```

Задание 8.2.12

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

```
> db.unicorns.update({name : "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}
}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "_id" : ObjectId("62862a05088b5516cfc543dd"), "name" : "Aurora", "loves" : [ "carrot"
, "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

Задание 8.2.13

Удалите документы с беспартийными мэрами.

```
> db.towns.remove({"mayor.party": {$exists: false}})
WriteResult({ "nRemoved" : 2 })
> db.towns.find()
{ "_id" : ObjectId("628ffa182e5d2dea8203d634"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "par
ty" : "I" } }
> db.towns.remove({})
WriteResult({ "nRemoved" : 1 })
> show collections
towns
unicorns
> db.towns.find()
> db.stats()
{
        "db" : "learn",
        "collections" : 2,
        "views" : 0,
        "objects" : 12,
        "avgObjSize" : 122.33333333333333,
        "dataSize" : 1468,
        "storageSize" : 61440,
        "indexes" : 2,
        "indexSize" : 49152,
        "totalSize" : 110592,
        "scaleFactor" : 1,
        "fsUsedSize" : 100849635328,
        "fsTotalSize" : 121123069952,
        "ok" : 1
}
>
```

Задание 8.3.1

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

```
> db.createCollection("traore")
{ "ok" : 1 }
> db.traore.insert({_id: "mt3545", name: "Mamoudou", contry: "Guinea"})
WriteResult({ "nInserted" : 1 })
> db.traore.insert({_id: "as4555", name: "Axmat", contry: "Tchad"})
WriteResult({ "nInserted" : 1 })
> db.unicorns.update({name: "Horny"}, {$set: {mtr: {$ref: "traore", $id: "mt3545"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Aurora"}, {$set: {mtr: {$ref: "traore", $id: "as4555"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

```
> db.traore.find()
{ "_id" : "mt3545", "name" : "Mamoudou", "contry" : "Guinea" }
{ "_id" : "as4555", "name" : "Axmat", "contry" : "Tchad" }
>
```

Вывод таблицы после редактирования

```
> db.unicorns.find()
{ "_id" : ObjectId("628f67232e5d2dea8203d626"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68, "mtr" : DBRef("traore", "mt3545") }
{ "_id" : ObjectId("628f67fd2e5d2dea8203d627"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f68252e5d2dea8203d628"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43, "mtr" : DBRef("traore", "as4555") }
{ "_id" : ObjectId("628f683f2e5d2dea8203d629"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f68602e5d2dea8203d62a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f68cf2e5d2dea8203d62b"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f692c2e5d2dea8203d62c"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f69442e5d2dea8203d62d"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f696c2e5d2dea8203d62e"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f69882e5d2dea8203d62f"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f6f4d2e5d2dea8203d630"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f95e62e5d2dea8203d631"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
>
```

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})
uncaught exception: TypeError: db.unicorns.ensureIndex is not a function :
@(shell):1:1
>
```

Задание 8.3.3

Получите информацию о всех индексах коллекции `unicorns` .

Удалите все индексы, кроме индекса для идентификатора.

Попытайтесь удалить индекс для идентификатора.

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

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

Задание 8.3.4

1) Создайте объемную коллекцию numbers, задействовав курсор:
   for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}.

```
> db.createCollection("numbers")
{ "ok" : 1 }
> for(i = 0; i < 100000; i++){dbnumbers.insert({value: i})}
uncaught exception: ReferenceError: dbnumbers is not defined :
@(shell):1:29
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
^[[AWriteResult({ "nInserted" : 1 })
> db.numbers.find()
{ "_id" : ObjectId("629013cf2e5d2dea8203d635"), "value" : 0 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d636"), "value" : 1 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d637"), "value" : 2 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d638"), "value" : 3 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d639"), "value" : 4 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d63a"), "value" : 5 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d63b"), "value" : 6 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d63c"), "value" : 7 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d63d"), "value" : 8 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d63e"), "value" : 9 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d63f"), "value" : 10 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d640"), "value" : 11 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d641"), "value" : 12 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d642"), "value" : 13 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d643"), "value" : 14 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d644"), "value" : 15 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d645"), "value" : 16 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d646"), "value" : 17 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d647"), "value" : 18 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d648"), "value" : 19 }
Type "it" for more
> it
{ "_id" : ObjectId("629013cf2e5d2dea8203d649"), "value" : 20 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d64a"), "value" : 21 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d64b"), "value" : 22 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d64c"), "value" : 23 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d64d"), "value" : 24 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d64e"), "value" : 25 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d64f"), "value" : 26 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d650"), "value" : 27 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d651"), "value" : 28 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d652"), "value" : 29 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d653"), "value" : 30 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d654"), "value" : 31 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d655"), "value" : 32 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d656"), "value" : 33 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d657"), "value" : 34 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d658"), "value" : 35 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d659"), "value" : 36 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d65a"), "value" : 37 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d65b"), "value" : 38 }
{ "_id" : ObjectId("629013cf2e5d2dea8203d65c"), "value" : 39 }
Type "it" for more
```

2) Выберите последних четыре документа.

```
> db.numbers.find().sort({value: -1}).limit(4)
{ "_id" : ObjectId("629014062e5d2dea82055cd4"), "value" : 99999 }
{ "_id" : ObjectId("629014062e5d2dea82055cd3"), "value" : 99998 }
{ "_id" : ObjectId("629014062e5d2dea82055cd2"), "value" : 99997 }
{ "_id" : ObjectId("629014062e5d2dea82055cd1"), "value" : 99996 }
> db.numbers.find().skip.(99996).limit(4)
SyntaxError: missing name after . operator :
@(shell):1:23
> db.numbers.find().skip(99996).limit(4)
{ "_id" : ObjectId("629014062e5d2dea82055cd1"), "value" : 99996 }
{ "_id" : ObjectId("629014062e5d2dea82055cd2"), "value" : 99997 }
{ "_id" : ObjectId("629014062e5d2dea82055cd3"), "value" : 99998 }
{ "_id" : ObjectId("629014062e5d2dea82055cd4"), "value" : 99999 }
>
```

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

```
> db.numbers.explain("executionStats").find().skip(99995).limit(5)
```

```
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 5,
                "executionTimeMillis" : 44,
```

4) Создайте индекс для ключа value.

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

5) Получите информацию о всех индексах коллекции numbres.

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

6) Выберите последних четыре документа.

```
> db.numbers.find().skip(99996).limit(4)
{ "_id" : ObjectId("629014062e5d2dea82055cd1"), "value" : 99996 }
{ "_id" : ObjectId("629014062e5d2dea82055cd2"), "value" : 99997 }
{ "_id" : ObjectId("629014062e5d2dea82055cd3"), "value" : 99998 }
{ "_id" : ObjectId("629014062e5d2dea82055cd4"), "value" : 99999 }
>
```

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

```
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 5,
                "executionTimeMillis" : 47,
```

Потребовалось 44 миллисекунд, до создания индекса – 47 миллисекунд. Очевидно, что с индексами запрос работает быстрее.

Вывод:

В данной работе были изучены основные функции MongoDB.