

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 в СУБД MongoDB»**
по дисциплине: Проектирование и реализация баз данных

Специальность: 09.03.03 Мобильные и сетевые технологии

Проверил:
Говорова М.М. _____
Оценка _____

Выполнил:
студент группы К3240
Безруков Андрей

Санкт-Петербург, 2021 г.

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.

Выполнение практического задания:

8.1.1. Используя второй способ, вставьте в коллекцию единорогов документ (после создания БД learn и заполнения коллекции unicorns)

```
> 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 })
```

8.1.2. 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("60b7925153f53b747c1d0255"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0250"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
> db.unicorns.find({gender: "f"}).limit(3).sort({name:1});
{ "_id" : ObjectId("60b6945d53f53b747c1d024b"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024f"), "name" : "Ayna", "loves" : [
"strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0252"), "name" : "Leia", "loves" : [
"apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
```

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

```
> db.unicorns.findOne({gender: "f", loves: "carrot"});
{
  "_id" : ObjectId("60b6945d53f53b747c1d024b"),
  "name" : "Aurora",
  "loves" : [
    "carrot",
    "grape"
  ],
  "weight" : 450,
  "gender" : "f",
  "vampires" : 43
}
> db.unicorns.find({gender: "f", loves: "carrot"}).limit(1);
{ "_id" : ObjectId("60b6945d53f53b747c1d024b"), "name" : "Aurora", "loves" :
[ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

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

```
> db.unicorns.find({gender: "m"}, {gender:0, loves:0}).limit(3).sort({name:1}
);
{ "_id" : ObjectId("60b7925153f53b747c1d0255"), "name" : "Dunx", "weight" : 7
04, "vampires" : 165 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "weight" :
600, "vampires" : 63 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0250"), "name" : "Kenny", "weight" :
690, "vampires" : 39 }
```

8.1.4 Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural: -1});
{ "_id" : ObjectId("60b7925153f53b747c1d0255"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0254"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b6945d53f53b747c1d0253"), "name" : "Pilot", "loves" : [
"apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0252"), "name" : "Leia", "loves" : [
"apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0251"), "name" : "Raleigh", "loves" :
[ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0250"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024f"), "name" : "Ayna", "loves" : [
"strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024e"), "name" : "Solnara", "loves" :
[ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampire
" : 80 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024d"), "name" : "Rooodooles", "love
" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024c"), "name" : "Unicrom", "loves" :
[ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }

{ "_id" : ObjectId("60b6945d53f53b747c1d024b"), "name" : "Aurora", "loves" :
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

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

```
> db.unicorns.find({}, {_id: false, 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" : "Rooooooodles", "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 }
```

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

```
> db.unicorns.find({gender: "f", weight: {$gte: 500, $lte: 700}}, {_id: false});
{ "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" }
```

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: false});
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
```

8.1.8 Найти всех единорогов, не имеющих ключ vampires

```
> db.unicorns.find({vampires: {$exists: false}});
{ "_id" : ObjectId("60b6945d53f53b747c1d0254"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

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

```
> db.unicorns.find({gender: "m"},{name:1, _id: false, 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" : "Rooodles", "loves" : [ "apple" ] }
{ "name" : "Unicrom", "loves" : [ "energon" ] }
```

8.2.1. (после создания коллекции towns)

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:true, mayor: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, mayor:true, _id: false});
{ "name" : "Punxsutawney ", "mayor" : { "name" : "Jim Wehrle" } }
```

8.2.2 1) Сформировать функцию для вывода списка самцов единорогов.

```
> fmale = 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(fmale);null;
null
> cursor.limit(2).sort({name:1});
{ "_id" : ObjectId("60b7925153f53b747c1d0255"), "name" : "Dunx", "loves" : [ "grape" , "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

3) Вывести результат, используя forEach.

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

8.2.3 Вывести количество самок единорогов весом от полутонны до 600 кг

```
> db.unicorns.find({gender: "f", weight: {$gte: 1000/2, $lte:600}}).count();
2
```

8.2.4 Вывести список предпочтений

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

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 }
```

8.2.6 Выполнить команду:

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

Проверить содержимое коллекции unicorns.


```
> db.unicorns.save({name: 'Barney', loves: ['grape'],
... weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.unicorns.find();
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "loves" : [ "carro
t", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024b"), "name" : "Aurora", "loves" : [ "carr
ot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024c"), "name" : "Unicrom", "loves" : [ "ene
rgon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024d"), "name" : "Roooooodles", "loves" : [
"apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024e"), "name" : "Solnara", "loves" : [ "app
le", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024f"), "name" : "Ayna", "loves" : [ "strawb
erry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0250"), "name" : "Kenny", "loves" : [ "grape
", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0251"), "name" : "Raleigh", "loves" : [ "app
le", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0252"), "name" : "Leia", "loves" : [ "apple"
, "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0253"), "name" : "Pilot", "loves" : [ "apple
", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0254"), "name" : "Nimue", "loves" : [ "grape
", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b7925153f53b747c1d0255"), "name" : "Dunx", "loves" : [ "grape"
, "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b7a5fa6d7d3655ac603d5c"), "name" : "Barney", "loves" : [ "grape
" ], "weight" : 340, "gender" : "m" }
```

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

```
> db.unicorns.update({name: 'Ayna'}, {name: "Ayna", loves: ['strawberry', 'lemon'], wei
ght: 800, gender: 'f', vampires: 51});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Ayna"});
{ "_id" : ObjectId("60b7aa396d7d3655ac603d5d"), "name" : "Ayna", "loves" : [ "strawb
erry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
```

8.2.8 Для самца единорога Raleigh внести изменения в БД: теперь он любит редбул.

```
> db.unicorns.update({name: "Raleigh", gender: "m"}, {$set: {loves: ["redbull"]}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Raleigh"});
{ "_id" : ObjectId("60b6945d53f53b747c1d0251"), "name" : "Raleigh", "loves" : [ "red
bull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
```

8.2.9 Всем самцам единорогов увеличить количество убитых вампиров на 5.

```
> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true});
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
```

```
> db.unicorns.find({gender: "m"});
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "loves" : [ "carro
t", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024c"), "name" : "Unicrom", "loves" : [ "ene
rgon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024d"), "name" : "Roooooodles", "loves" : [
"apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0250"), "name" : "Kenny", "loves" : [ "grape
", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0251"), "name" : "Raleigh", "loves" : [ "red
bull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0253"), "name" : "Pilot", "loves" : [ "apple
", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("60b7925153f53b747c1d0255"), "name" : "Dunx", "loves" : [ "grape"
, "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("60b7a5fa6d7d3655ac603d5c"), "name" : "Barney", "loves" : [ "grape"
], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

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("60b7a1086d7d3655ac603d59"), "name" : "Punxsutawney ", "populatiu
on" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ],
"mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("60b7a13d6d7d3655ac603d5a"), "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("60b7a1986d7d3655ac603d5b"), "name" : "Portland", "populatiuon" :
528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "
food" ], "mayor" : { "name" : "Sam Adams" } }
```

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("60b6945d53f53b747c1d0253"), "name" : "Pilot", "loves" : [ "apple
", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
```

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 })
> db.unicorns.find({name: "Aurora"});
{ "_id" : ObjectId("60b6945d53f53b747c1d024b"), "name" : "Aurora", "loves" : [ "carr
ot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

8.2.13 Удалите документы с беспартийными мэрами. Проверьте содержание коллекции. Очистите коллекцию. Просмотрите список доступных коллекций.


```

> db.towns.remove({"mayor.party":{"$exists: false}});
WriteResult({ "nRemoved" : 2 })
> db.towns.find();
{ "_id" : ObjectId("60b7a13d6d7d3655ac603d5a"), "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" } }
> db.towns.remove({});
WriteResult({ "nRemoved" : 1 })
> show collections;
towns
unicorns

```

8.3.1 Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание. Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания. Проверьте содержание коллекции единорогов.

```

> db.zones.insert({_id: "woods", name: "Blessed Woods", char: "Situated in Heaven"})
;
WriteResult({ "nInserted" : 1 })
> db.zones.insert({_id: "rocks", name: "Burning Rocks", char: "Situated in Hell"});
WriteResult({ "nInserted" : 1 })
> db.zones.insert({_id: "garden", name: "Golden Garden", char: "Situated on Earth"})
;
WriteResult({ "nInserted" : 1 })

```

```

> db.unicorns.update({name: "Barney"}, {$set: {zone: {$ref: "zones", $id: "garden"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Dunx"}, {$set: {zone: {$ref: "zones", $id: "rocks"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Pilot"}, {$set: {zone: {$ref: "zones", $id: "woods"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find();
{ "_id" : ObjectId("60b6945d53f53b747c1d024a"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024b"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024c"), "name" : "Unicrom", "loves" : [ "energy", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024d"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024e"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b6945d53f53b747c1d024f"), "weight" : 800, "vampires" : 51 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0250"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0251"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0252"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b6945d53f53b747c1d0253"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59, "zone" : DBRef("zones", "woods") }
{ "_id" : ObjectId("60b6945d53f53b747c1d0254"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b7925153f53b747c1d0255"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170, "zone" : DBRef("zones", "rocks") }
{ "_id" : ObjectId("60b7a5fa6d7d3655ac603d5c"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5, "zone" : DBRef("zones", "garden") }
{ "_id" : ObjectId("60b7aa396d7d3655ac603d5d"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }

```

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

8.3.4 1) Создайте объемную коллекцию numbers, задействовав курсор. 2) Выберите последних четыре документа. 3) Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis) 4) Создайте индекс для ключа value. 5) Получите информацию о всех индексах коллекции numbers. 6) Выполните запрос 2. 7) Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса? 8) Сравните время выполнения запросов с индексом и

без. Дайте ответ на вопрос: какой запрос более эффективен?

```
> db.numbers.find().sort({$natural: -1}).limit(4);
{ "_id" : ObjectId("60b7bb296d7d3655ac61c3fd"), "value" : 99999 }
{ "_id" : ObjectId("60b7bb296d7d3655ac61c3fc"), "value" : 99998 }
{ "_id" : ObjectId("60b7bb296d7d3655ac61c3fb"), "value" : 99997 }
{ "_id" : ObjectId("60b7bb296d7d3655ac61c3fa"), "value" : 99996 }
> db.numbers.explain("executionStats").find({value: {$gte:99996}});
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "learn.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "value" : {
        "$gte" : 99996
      }
    },
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "value" : {
          "$gte" : 99996
        }
      }
    },
    "direction" : "forward"
  },
  "rejectedPlans" : [ ]
},
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 4,
    "executionTimeMillis" : 101,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 100000,
    "executionStages" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "value" : {
          "$gte" : 99996
        }
      }
    }
  },
}
```

```

        "nReturned" : 4,
        "executionTimeMillisEstimate" : 2,
        "works" : 100002,
        "advanced" : 4,
        "needTime" : 99997,
        "needYield" : 0,
        "saveState" : 100,
        "restoreState" : 100,
        "isEOF" : 1,
        "direction" : "forward",
        "docsExamined" : 100000
    }
},
"serverInfo" : {
    "host" : "DESKTOP-HNNVPHI",
    "port" : 27017,
    "version" : "4.4.6",
    "gitVersion" : "72e66213c2c3eab37d9358d5e78ad7f5c1d0d0d7"
},
"ok" : 1
}
> db.numbers.ensureIndex({"value": 1});
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 1,
    "numIndexesAfter" : 2,
    "ok" : 1
}
> 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" : false,
            "isSparse" : false,
            "isPartial" : false,
            "indexVersion" : 2,
            "direction" : "forward",
            "indexBounds" : {
                "value" : [
                    "[99996.0, inf.0]"
                ]
            }
        },
    },
    "rejectedPlans" : [ ]
},
"executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 4,
    "executionTimeMillis" : 62,
    "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,
        "keyPattern" : {
            "value" : 1
        },
        "indexName" : "value_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
            "value" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : false,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
            "value" : [
                "[99996.0, inf.0]"
            ]
        },
        "keysExamined" : 4,
        "seeks" : 1,
        "dupsTested" : 0,
        "dupsDropped" : 0
    }
},
"serverInfo" : {
    "host" : "DESKTOP-HNNVPHI",
    "port" : 27017,
    "version" : "4.4.6",
    "gitVersion" : "72e66213c2c3eab37d9358d5e78ad7f5c1d0d0d7"
},
"ok" : 1
}

```

Скорость запроса увеличилась с 101 мс до 62 мс (примерно на 40%) благодаря созданию индекса на поле “value”, поэтому индексы эффективнее.

Вывод: по итогам работы я познакомился с очередной СУБД MongoDB и научился создавать запросы на выборку данных, вставку, изменение и удаление документов.