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 II курса ИКТ
группы K3243
Царьков Григорий Иванович

Проверил:
Говорова Марина Михайловна

Санкт-Петербург
2022

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

Выполнение:

Практическое задание 1.1:

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.*

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> 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.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
>
```

Практическое задание 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:"f"}).sort({name:1}).limit(3)
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
> db.unicorns.find({gender:"m"}).sort({name:1})
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
>
```

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("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
> db.unicorns.findOne({gender:"f", loves:"carrot"})
{
        "_id" : ObjectId("6294ed6e06f37253c74709da"),
        "name" : "Aurora",
        "loves" : [
                "carrot",
                "grape"
        ],
        "weight" : 450,
        "gender" : "f",
        "vampires" : 43
}
>
```

Практическое задание 1.3

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

```
> db.unicorns.find({gender:"m"}, {loves:0, gender:0}).sort({name:1})
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "weight" : 704, "vampires" : 165 }
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
>
```

Практическое задание 1.4

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural:-1})
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
>
```

Практическое задание 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 }
>
```

Практическое задание 1.6

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

```
> db.unicorns.find({gender: "f", weight:{$gt:500, $lt:700}},{_id:0})
{ "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 1.7

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

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

Практическое задание 1.8

Найти всех единорогов, не имеющих ключ vampires.

```
> db.unicorns.find({vampires:{$exists:false}})
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>
```

Практическое задание 1.9

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

```
> db.unicorns.find({gender:"m"}, {_id:0, name:1, 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" ] }
>
```

Практическое задание 2.1

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

```
> db.towns.insertMany([{name: "Punxsutawney ", populatiuon: 6200,  last_sensus: ISODate("2008-01-31"),
famous_for: [""],  mayor: {    name: "Jim Wehrle"  }}, {name: "New York",  populatiuon: 22200000,  last_
sensus: ISODate("2009-07-31"),  famous_for: ["status of liberty", "food"],  mayor: {    name: "Michael B
loomberg",  party: "I"}}, {name: "Portland",  populatiuon: 528000,  last_sensus: ISODate("2009-07-20"),
 famous_for: ["beer", "food"],  mayor: {    name: "Sam Adams",  party: "D"}} ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("629509e206f37253c74709e5"),
                ObjectId("629509e206f37253c74709e6"),
                ObjectId("629509e206f37253c74709e7")
        ]
}
```

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

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

Практическое задание 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("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "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.*
3) *Вывести результат, используя forEach.*

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

Практическое задание 2.3

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 2.4

Вывести список предпочтений.

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

Практическое задание 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 }
```

Практическое задание 2.6

1) *Выполнить команду:*

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

2) *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6295140006f37253c74709e8"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
>
```

Практическое задание 2.7

1) *Для самки единорога* Ayna *внести изменения в БД: теперь ее вес 800, она убила 51 вампира.*
2) *Проверить содержимое коллекции unicorns.*

```
> db.unicorns.update({name:"Ayna"}, {$set:{weight:800, vampires:51}}, {upsert:false})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6295140006f37253c74709e8"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 2.8

1) *Для самца единорога* Raleigh *внести изменения в БД: теперь он любит рэдбул.*
2) *Проверить содержимое коллекции unicorns.*

```
> db.unicorns.update({name:"Raleigh"}, {$set:{loves: [ "redbull" ]}}, {upsert:false})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6295140006f37253c74709e8"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 2.9

1) *Всем самцам единорогов увеличить количество убитых вампиров на 5.*
2) *Проверить содержимое коллекции* `unicorns.`

```
> db.unicorns.update({gender:"m"}, {$inc: {vampires:5}}, {multi:true})
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
> db.unicorns.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("6295140006f37253c74709e8"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 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("629509e206f37253c74709e5"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("629509e206f37253c74709e6"), "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("629509e206f37253c74709e7"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams" }
}
>
```

Практическое задание 2.11

1) *Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.*
2) *Проверить содержимое коллекции* `unicorns.`

```
> db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("629514006f37253c74709e8"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
>
```

Практическое задание 2.12

1) *Изменить информацию о самке единорога* `Aurora`: *теперь она любит еще и сахар, и лимоны.*
2) *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.update({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("6294ed6e06f37253c74709d9"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6294ed6e06f37253c74709da"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6294ed6e06f37253c74709db"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dc"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6294ed6e06f37253c74709dd"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6294ed6e06f37253c74709de"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6294ed6e06f37253c74709df"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e0"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e1"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e2"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6294ed6e06f37253c74709e3"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6294ede706f37253c74709e4"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("629514006f37253c74709e8"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 2.13

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

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

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

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

2) *Удалите документы с беспартийными мэрами.*
3) *Проверьте содержание коллекции.*

```
> db.towns.insertMany([{name: "Punxsutawney ",
... popujatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: ["phil the groundhog"],
... mayor: {
...     name: "Jim Wehrle"
... }},
... {name: "New York",
... popujatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...     name: "Michael Bloomberg",
... party: "I"}},
... {name: "Portland",
... popujatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...     name: "Sam Adams",
... party: "D"}}]
... )
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("6295364a06f37253c74709e9"),
                ObjectId("6295364a06f37253c74709ea"),
                ObjectId("6295364a06f37253c74709eb")
        ]
}
> db.towns.remove({"mayor.party": {$exists: false}})
WriteResult({ "nRemoved" : 1 })
> db.towns.find()
{ "_id" : ObjectId("6295364a06f37253c74709ea"), "name" : "New York", "popujatiuon" : 22200000, "last_sen
sus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "nam
e" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("6295364a06f37253c74709eb"), "name" : "Portland", "popujatiuon" : 528000, "last_sensu
s" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams
", "party" : "D" } }
>
```

4) *Очистите коллекцию.*
5) *Просмотрите список доступных коллекций.*

```
> db.towns.drop()
true
> show Collections
uncaught exception: Error: don't know how to show [Collections] :
shellHelper.show@src/mongo/shell/utils.js:1211:11
shellHelper@src/mongo/shell/utils.js:838:15
@(shellhelp2):1:1
> show collections
unicorns
>
```

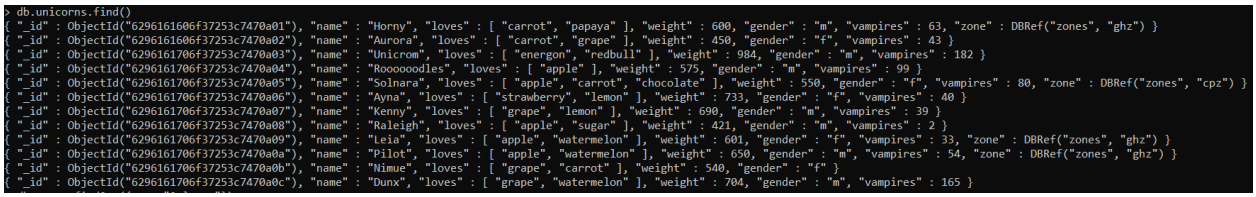
Практическое задание 3.1

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

Содержание коллекции единорогов `unicorns`*:*

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

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

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

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

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

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

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

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

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

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

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

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

```
> db.zones.insert({_id:"ghz", name:"Green Hill", description:"the first of many"})
WriteResult({ "nInserted" : 1 })
> db.unicorns.update({name:"Horny"}, {$set: {zone:{$ref:"zones", $id:"ghz"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name:"Leia"}, {$set: {zone:{$ref:"zones", $id:"ghz"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name:"Pilot"}, {$set: {zone:{$ref:"zones", $id:"ghz"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.zones.insert({_id:"cpz", name:"Chemical Plant", description:"the industrial one"})
WriteResult({ "nInserted" : 1 })
> db.unicorns.update({name:"Solnara"}, {$set: {zone:{$ref:"zones", $id:"cpz"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```



```
> var solnara = db.unicorns.findOne({"_id":ObjectId("6296161706f37253c7470a05")})
> db[solnara.zone.$ref].findOne({_id:solnara.zone.$id})
{
        "_id" : "cpz",
        "name" : "Chemical Plant",
        "description" : "the industrial one"
}
> 
```

Практическое задание 3.2

1) *Проверьте, можно ли задать для коллекции* `unicorns` *индекс для ключа* `name` *с флагом* `unique`.
2) *Содержание коллекции единорогов* `unicorns`*:*

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

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

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

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

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

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

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

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

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

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

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

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

```
> db.unicorns.ensureIndex({"name":1},{"unique":true})
uncaught exception: TypeError: db.unicorns.ensureIndex is not a function :
@(shell):1:1
> db.unicorns.createIndex({"name":1},{"unique":true})
{
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "createdCollectionAutomatically" : false,
        "ok" : 1
}
```

Практическое задание 3.3

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

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

2) *Удалите все индексы, кроме индекса для идентификатора.*
3) *Попытайтесь удалить индекс для идентификатора.*

```
> db.unicorns.dropIndex("name_1")
{ "nIndexesWas" : 2, "ok" : 1 }
> db.unicorns.dropIndex("_id_")
{
        "ok" : 0,
        "errmsg" : "cannot drop _id index",
        "code" : 72,
        "codeName" : "InvalidOptions"
}
```

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

```
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})
... }
WriteResult({ "nInserted" : 1 })
>
> db.numbers.explain("executionStats").find().sort({$natural:-1}).limit(4)
{
        "explainVersion" : "1",
        "queryPlanner" : {
                "namespace" : "learn.numbers",
                "indexFilterSet" : false,
                "parsedQuery" : {

                },
                "maxIndexedOrSolutionsReached" : false,
                "maxIndexedAndSolutionsReached" : false,
                "maxScansToExplodeReached" : false,
                "winningPlan" : {
                        "stage" : "LIMIT",
                        "limitAmount" : 4,
                        "inputStage" : {
                                "stage" : "COLLSCAN",
                                "direction" : "backward"
                        }
                },
                "rejectedPlans" : [ ]
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 4,
                "executionTimeMillis" : 6,
```

Потребовалось 6 миллисекунд на выполнение этого запроса.

```
> db.numbers.find().sort({$natural:-1}).limit(4)
{ "_id" : ObjectId("62961f6406f37253c74890b8"), "value" : 99999 }
{ "_id" : ObjectId("62961f6406f37253c74890b7"), "value" : 99998 }
{ "_id" : ObjectId("62961f6406f37253c74890b6"), "value" : 99997 }
{ "_id" : ObjectId("62961f6406f37253c74890b5"), "value" : 99996 }
```

4) *Создайте индекс для ключа value.*
5) *Получите информацию о всех индексах коллекции numbers.*

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

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?*

Можем увидеть, что время выполнения с индексом стремится к 0 миллисекунд, таким образом, индексирование может существенно ускорить поиск данных.

Вывод: я освоил GUI MongoDB Compass и непосредственно основы самой СУБД MongoDB, выполнив 26 практических заданий на CRUD-операции, курсоры, агрегирование данных, ссылки и индексы. СУБД оказалась относительно удобной для использования благодаря её использованию

документов в стиле JSON и отсутствию необходимости задавать изначальную схему таблицы. Из неудобств можно заметить слегка непривычный для SQL-систем синтаксис и местами не очень точные описания ошибок (один из таких примеров можно видеть в задании 3.2).