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 № 5.2

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. _____
Дата: «__» _____2022 г.
Оценка _____

Выполнил:
студент группы K3240

Балдина Д.Д.

Санкт-Петербург 2022 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.

ПРАКТИЧЕСКОЕ ЗАДАНИЕ

Практическое задание 8.1.1:

1) *Создайте базу данных learn*

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

2) *Заполните 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: '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 })
```

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

```
> 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(dociment)
uncaught exception: ReferenceError: dociment is not defined :
@(shell):1:1
> db.unicorns.insert(document)
WriteResult({ "nInserted" : 1 })
```

4) Проверьте содержимое коллекции с помощью метода find.

```
> db.unicorns.find()
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae695"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae699"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69b"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
>
```

Практическое задание 8.1.2:

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

Для самцов:

```
> db.unicorns.find({gender:"m"}).sort({name:1})
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
>
```

Для самок:

```
> db.unicorns.find({gender:"f"}).sort({name:1}).limit(3)
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae699"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
```

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

С findOne():

```
> db.unicorns.findOne({gender:"f", loves:"carrot"})
{
        "_id" : ObjectId("628b74ae3dae894bf27ae692"),
        "name" : "Aurora",
        "loves" : [
                "carrot",
                "grape"
        ],
        "weight" : 450,
        "gender" : "f",
        "vampires" : 43
}
```

С limit(1):

```
> db.unicorns.find({gender:"f", loves:"carrot"}).limit(1)
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

Практическое задание 8.1.3:

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

```
> db.unicorns.find({gender:"m"}, {loves:0, gender:0}).sort({name:1})
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "weight" : 704, "vampires" : 165 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
```

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

1 способ: так как id добавляются автоматически и идут по порядку, то последний добавленный объект будет с самым большим id, поэтому можем сделать сортировку по убыванию id:

```
> db.unicorns.find().sort({_id:-1})
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69b"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae699"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae695"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

2 способ:

```
> db.unicorns.find().sort({ $natural: -1 })
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69b"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae699"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae695"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

Практическое задание 8.1.5:

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

```
> db.unicorns.find({}, {loves: {$slice : 1},_id:0})
{ "name" : "Horny", "loves" : [ "carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "name" : "Aurora", "loves" : [ "carrot" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "name" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "name" : "Solnara", "loves" : [ "apple" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "name" : "Ayna", "loves" : [ "strawberry" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "name" : "Leia", "loves" : [ "apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape" ], "weight" : 540, "gender" : "f" }
{ "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Практическое задание 8.1.6:

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 8.1.7:

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

```
> db.unicorns.find({gender:"m", weight:{$gt : 500}, loves: {$in:["grape","lemon"]}},{_id : 0})
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ `vampires`.

```
> db.unicorns.find ({vampires: {$exists:false}}, {_id : 0})
{ "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

Практическое задание 8.1.9:

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

```
> db.unicorns.find({gender:"m"}, {loves: {$slice : 1}}).sort({name : 1})
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

Практическое задание 8.2.1:

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

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

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

{name: "Portland",
populatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
```

```
mayor: {
   name: "Sam Adams",
   party: "D"}}
```

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

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

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

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

Практическое 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("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "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
> cursor.sort({name : 1}).limit(2)
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
>
```

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 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 8.2.4:

Вывести список предпочтений.

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

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов

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

Практическое задание 8.2.6:

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("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae695"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampi
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae699"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69b"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628beea63dae894bf27ae6a2"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 8.2.7:

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

```
> db.unicorns.find({name : "Ayna"})
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
> db.unicorns.update({name : "Ayna"}, {name: "Ayna", weight : 800, gender: "f", vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name : "Ayna"})
{ "_id" : ObjectId("628b74ae3dae894bf27ae696"), "name" : "Ayna", "weight" : 800, "gender" : "f", "vampires" : 51 }
```

Практическое задание 8.2.8:

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

```
> db.unicorns.find({name : "Raleigh"})
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
> db.unicorns.update({name : "Raleigh"}, {$set : {loves : ["redbull"]}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name : "Raleigh"})
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
```

Практическое задание 8.2.9:

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

```
> db.unicorns.find({gender : "m"})
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628beea63dae894bf27ae6a2"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
> db.unicorns.updateMany({gender : "m"}, {$inc: {vampires:5}})
{ "acknowledged" : true, "matchedCount" : 8, "modifiedCount" : 8 }
> db.unicorns.find({gender : "m"})
{ "_id" : ObjectId("628b74ae3dae894bf27ae691"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae693"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187
{ "_id" : ObjectId("628b74ae3dae894bf27ae694"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae697"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae698"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("628b756a3dae894bf27ae69c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("628beea63dae894bf27ae6a2"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 8.2.10:

1. *Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.*
2. *Проверить содержимое коллекции* `towns`.

```
> db.towns.find()
{ "_id" : ObjectId("628bd14f3dae894bf27ae69f"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
atus of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("628bd1643dae894bf27ae6a0"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ ""
 ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("628bd1a03dae894bf27ae6a1"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
> db.towns.update({name : "Portland"}, {$unset: {"mayor.party": 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find()
{ "_id" : ObjectId("628bd14f3dae894bf27ae69f"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
atus of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("628bd1643dae894bf27ae6a0"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ ""
 ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("628bd1a03dae894bf27ae6a1"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
", "food" ], "mayor" : { "name" : "Sam Adams" } }
```

Практическое задание 8.2.11:

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

```
> db.unicorns.find({name : "Pilot"}, {loves : 1})
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "loves" : [ "apple", "watermelon" ] }
> db.unicorns.update({name : "Pilot"}, {$push: {loves: "chocolate"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name : "Pilot"}, {loves : 1})
{ "_id" : ObjectId("628b74ae3dae894bf27ae69a"), "loves" : [ "apple", "watermelon", "chocolate" ] }
```

Практическое задание 8.2.12:

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

```
> db.unicorns.find({name : "Aurora"}, {loves : 1})
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "loves" : [ "carrot", "grape" ] }
> db.unicorns.update({name : "Aurora"}, {$addToSet: {loves: {$each : ["sugar", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name : "Aurora"}, {loves : 1})
{ "_id" : ObjectId("628b74ae3dae894bf27ae692"), "loves" : [ "carrot", "grape", "sugar", "lemon" ] }
>
```

Практическое задание 8.2.13:

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

```
> db.towns.find()
{ "_id" : ObjectId("628bd14f3dae894bf27ae69f"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
atus of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("628bd1643dae894bf27ae6a0"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ ""
 ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("628bd1a03dae894bf27ae6a1"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
> db.towns.remove({"mayor.party": {$exists : false}})
WriteResult({ "nRemoved" : 1 })
> db.towns.find()
{ "_id" : ObjectId("628bd14f3dae894bf27ae69f"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
atus of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("628bd1a03dae894bf27ae6a1"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }
```

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

Практическое задание 8.3.1:

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

```
> db.zones.insertMany([{id:"forest", name: "forest", description: "A forest with lots of trees"},
... {id:"b_forest", name: "birch forest", description : "A forest with lots of birch trees"},
... {id: "j_forest", name: "jungle", description: "A forest with lots of jungle trees"},
... {id: "t_forest", name: "taiga", description: "A forest with lots of spruce trees"}
... ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("628c92fc3dae894bf27ae6ae"),
                ObjectId("628c92fc3dae894bf27ae6af"),
                ObjectId("628c92fc3dae894bf27ae6b0"),
                ObjectId("628c92fc3dae894bf27ae6b1")
        ]
}
> db.zones.find()
{ "_id" : ObjectId("628c92fc3dae894bf27ae6ae"), "id" : "forest", "name" : "forest", "description" : "A forest with lots of trees" }
{ "_id" : ObjectId("628c92fc3dae894bf27ae6af"), "id" : "b_forest", "name" : "birch forest", "description" : "A forest with lots of birch trees" }
{ "_id" : ObjectId("628c92fc3dae894bf27ae6b0"), "id" : "j_forest", "name" : "jungle", "description" : "A forest with lots of jungle trees" }
{ "_id" : ObjectId("628c92fc3dae894bf27ae6b1"), "id" : "t_forest", "name" : "taiga", "description" : "A forest with lots of spruce trees" }
```

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

```
> db.unicorns.update({name: "Leia"}, {$set: {zone: {$ref: "zones", $id : "j_forest"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name:"Leia"})
{ "_id" : ObjectId("628c8d083dae894bf27ae6aa"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33, "zone" : DBRe
f("zones", "j_forest") }
```

3) *Проверьте содержание коллекции едиорогов.*

```
> db.unicorns.find()
{ "_id" : ObjectId("628c8d083dae894bf27ae6a3"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6a4"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6a5"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6a6"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6a7"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6a8"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6a9"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6aa"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33, "zone" : DBRe
f("zones", "j_forest") }
{ "_id" : ObjectId("628c8d083dae894bf27ae6ab"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628c8d083dae894bf27ae6ac"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628c8d2c3dae894bf27ae6ad"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

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

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

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

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

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

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

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

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

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

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

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

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

Практическое задание 8.3.2:

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

```
> db.unicorns.createIndex({"name" : 1}, {"unique" : true})
{
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "createdCollectionAutomatically" : false,
        "ok" : 1
}
> db.unicorns.insert({name:"Pilot"})
WriteResult({
        "nInserted" : 0,
        "writeError" : {
                "code" : 11000,
                "errmsg" : "E11000 duplicate key error collection: learn.unicorns index: name_1 dup key: { name: \"Pilot\" }"
        }
})
```

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:'Kenn y', 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: 16
```

Практическое задание 8.3.3:

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

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

2) Удалите все индексы, кроме индекса для идентификатора.

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

3) Попытайтесь удалить индекс для идентификатора

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

Практическое 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++){db.numbers.insert({value: i})}
WriteResult({ "nInserted" : 1 })
```

2) *Выберите последних четыре документа.*

```
> db.numbers.find().sort({$natural : -1}). limit(4)
{ "_id" : ObjectId("628ca080edc60cbe5c339d35"), "value" : 99999 }
{ "_id" : ObjectId("628ca080edc60cbe5c339d34"), "value" : 99998 }
{ "_id" : ObjectId("628ca080edc60cbe5c339d33"), "value" : 99997 }
{ "_id" : ObjectId("628ca080edc60cbe5c339d32"), "value" : 99996 }
>
```

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

```
> db.users.explain("executionStats").find().sort({value:-1}).limit(4)
{
        "explainVersion" : "1",
        "queryPlanner" : {
                "namespace" : "learn.users",
                "indexFilterSet" : false,
                "parsedQuery" : {

                },
                "maxIndexedOrSolutionsReached" : false,
                "maxIndexedAndSolutionsReached" : false,
                "maxScansToExplodeReached" : false,
                "winningPlan" : {
                        "stage" : "EOF"
                },
                "rejectedPlans" : [ ]
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 0,
                "executionTimeMillis" : 1,
                "totalKeysExamined" : 0,
                "totalDocsExamined" : 0,
                "executionStages" : {
                        "stage" : "EOF",
                        "nReturned" : 0,
                        "executionTimeMillisEstimate" : 0,
                        "works" : 1,
                        "advanced" : 0,
                        "needTime" : 0,
```

4) *Создайте индекс для ключа* `value`.

```
> db.numbers.createIndex({"value":1})
{
        "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"
        }
]
```

5) *Получите информацию о всех индексах коллекции* `numbres`.*(см. пункт 4)*

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?*

ВЫВОД: можно заметить, что с индексами запрос сработал быстрее на 1 мс. Это незначительная разница, но на данных бОльших объемов эта разница будет еще более заметной. Поэтому можно утверждать, что с индексами запрос будет работать быстрее и эффективнее.

ВЫВОДЫ

В ходе работы были получены практические навыки работы с CRUD - операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, с ссылками и индексами в базе данных MongoDB.

MongoDB предоставляет мощный CLI интерфейс для выполнения CRUD операций, отличительной особенностью является интеграция полноценного языка программирования: Javascript.