Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт

по лабораторной работе №6 «Введение в СУБД MongoDB. Работа с БД в СУБД MongoDB.»

По дисциплине **«Проектирование и реализация баз данных»**

Автор: Сергеев В. Ю.

Факультет: ИКТ

Группа: К3141

Преподаватель: Говорова М. М.

Санкт-Петербург, 2023

Оглавление

Содержание работы

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.

Оборудование

Компьютерный класс.

Программное обеспечение

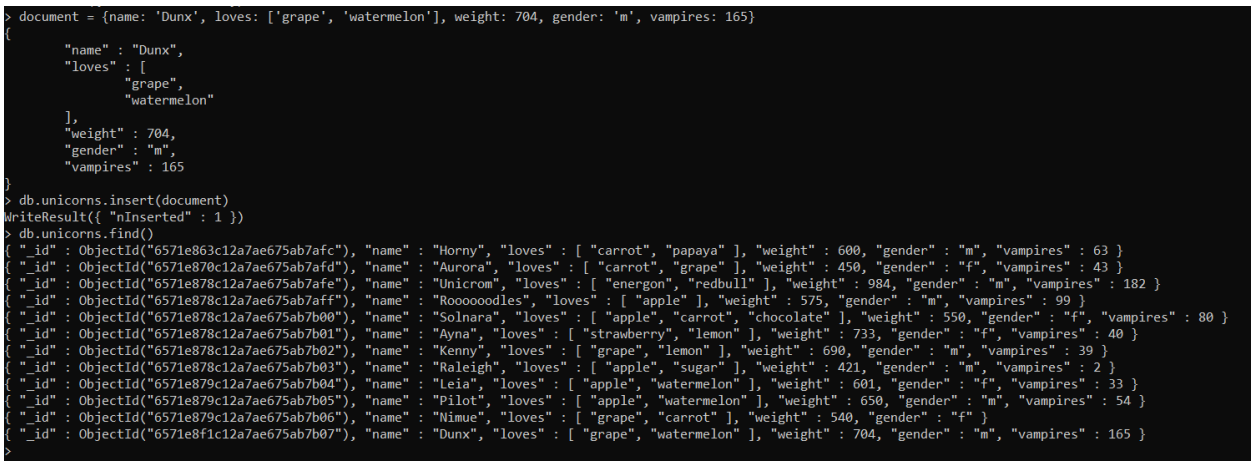
СУБД MongoDB 4+, 6.0.6 (текущая).

Выполнение

Практическое задание 2.1.1

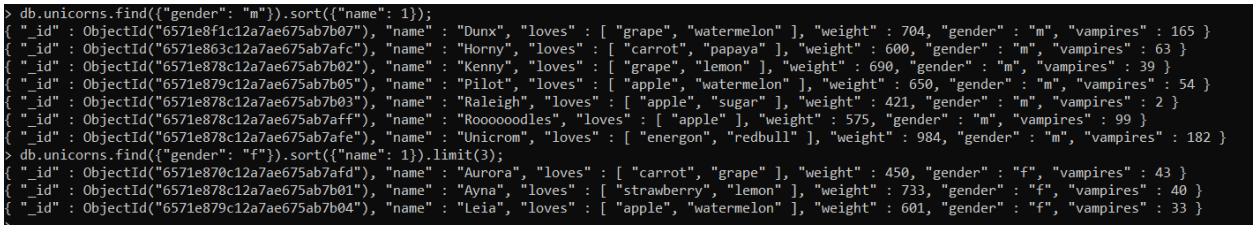
1. Создайте базу данных learn
2. Заполните коллекцию единорогов unicorns
3. Используя второй способ, вставьте в коллекцию единорогов документ
4. Проверьте содержимое коллекции с помощью метода find

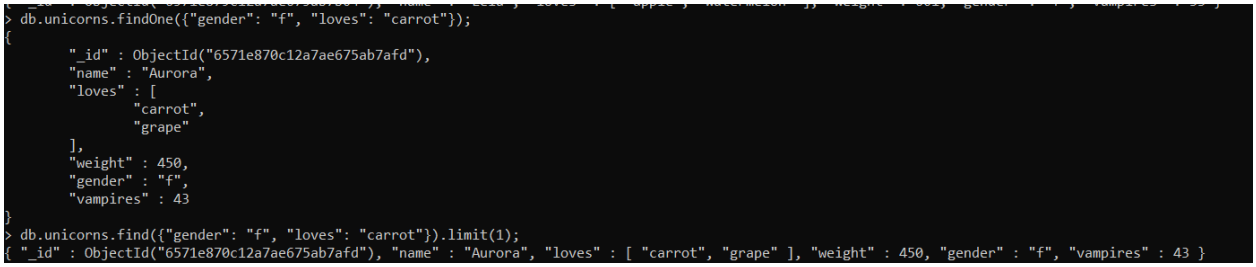
```
> use learn
switched to db learn
> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43})
WriteResult({ "nInserted" : 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 })
```



Практическое задание 2.2.1

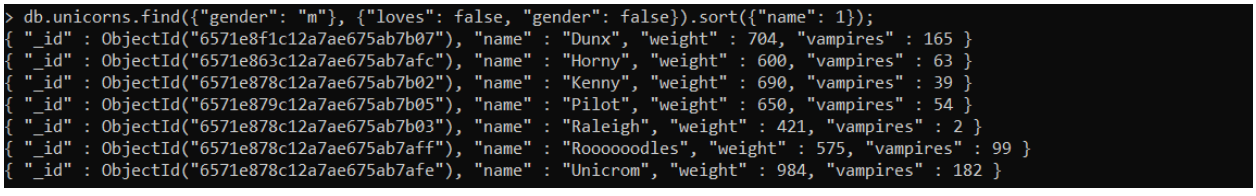
1. Сформулируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.
2. Найдите всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций findOne и limit.





Практическое задание 2.2.2

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



Практическое задание 2.2.3

Вывести список самцов в обратном порядке добавления.

```
> db.unicorns.find({"gender": "m"}).sort({$natural: -1})
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

Практическое задание 2.2(1).3

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор

```
> db.unicorns.find({}, {loves: {$slice: 1}, "_id": false});
{ "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 2.3.1

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

```
> db.unicorns.find({weight: {$gte: 500, $lte: 700}}, {_id: false});
{ "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

Практическое задание 2.3.2

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

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

Практическое задание 2.3.3

Вывести всех единорогов, не имеющих поле vampires

```
> db.unicorns.find({vampires: {$exists: false}});
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

Практическое задание 2.3.4

Вывести упорядоченный список имен единорогов и информацией об их первом предпочтении

```
> db.unicorns.find({gender: "m"}, {_id: false, name: true, 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 3.1.1

1. Создайте коллекцию towns, включающую следующие документы
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.insert({name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...     name: "Jim Wehrle"
... }}
... );
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "New York",
... populatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...     name: "Michael Bloomberg",
... party: "I"}}
... );
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Portland",
... populatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...     name: "Sam Adams",
... party: "D"}}
... );
WriteResult({ "nInserted" : 1 })
```

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

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

Практическое задание 3.1.2

1. Сформировать функцию для вывода списка самцов единорогов
2. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке
3. Вывести результат, используя forEach

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

```
> fn = function() {return this.gender == "m";}
function() {return this.gender == "m";}
> db.unicorns.find(fn);
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
> var cursor = db.unicorns.find(fn).sort({name: 1}).limit(2); null;
null
> cursor.forEach(function(obj){ print(obj.name); })
Dunx
Horny
```

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг

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

Практическое задание 3.2.2

Вывести список предпочтений

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

Практическое задание 3.2.3

Посчитать количество особей обоих полов

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

Практическое задание 3.3.1

1. Выполнить команду
2. Проверить содержимое коллекции unicorns

```
> db.unicorns.save({name: "Barny", loves: ["grape"], weight: 340, gender: "m"});
WriteResult({ "nInserted" : 1 })
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 3.3.2

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

```
> db.unicorns.update({name: "Ayna", gender: "f"}, {$set: {weight: 800, vampires: 51}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 3.3.3

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

```
> db.unicorns.update({name: "Raleigh"}, {$push: {loves: "redbull"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar", "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 3.3.4

1. Всем самцам единорогов увеличить количество убитых вампиров на 5
2. Проверить содержимое коллекции towns

```
> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true});
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar", "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 3.3.5

1. Изменить информацию о городе Портлэнд: мэр этого города теперь беспартийный
2. Проверить содержимое коллекции unicorns

```
> db.towns.update({name: "Portland"}, {$unset: {"mayor.party": 1}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find({}, {name: true, mayor: true});
{ "_id" : ObjectId("65722678c12a7ae675ab7b0c"), "name" : "Punxsutawney ", "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("6572268bc12a7ae675ab7b0d"), "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("6572269bc12a7ae675ab7b0e"), "name" : "Portland", "mayor" : { "name" : "Sam Adams" } }
```

Практическое задание 3.3.6

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

```
> db.unicorns.update({name: "Pilot", gender: "m"}, {$push: {loves: "chocolate"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar", "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 3.3.7

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

```
> db.unicorns.update({name: "Aurora", gender: "f"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar", "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 3.4.1

1. Создайте коллекцию towns, со следующими документами
2. Удалите документы с беспартийными мэрами
3. Проверьте содержимое коллекции
4. Очистите коллекцию
5. Просмотрите список доступных коллекций

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

```
> db.towns.remove({"mayor.party": {$exists: false}});
WriteResult({ "nRemoved" : 1 })
> db.towns.find();
{ "_id" : ObjectId("65722b41c12a7ae675ab7b10"), "name" : "New York", "popujatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for"
: "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("65722b51c12a7ae675ab7b11"), "name" : "Portland", "popujatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" :
"party" : "D" } }
> db.towns.remove({});
WriteResult({ "nRemoved" : 2 })
> show collections
towns
unicorns
> db.towns.find();
>
```

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

Практическое задание 4.1.1

1. Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратно 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. Проверить содержимое коллекции unicorns

```
> db.zones.insert({_id: "earth", name: "Plains", desc: "Normal enviroment for common unicorns."});
WriteResult({ "nInserted" : 1 })
> db.zones.insert({_id: "water", name: "Sea", desc: "Underwater enviroment for water element unicorns."});
WriteResult({ "nInserted" : 1 })
> db.zones.insert({_id: "air", name: "High Mountains", desc: "Place above clouds for unicorns which feel comfortable in the sky."});
WriteResult({ "nInserted" : 1 })
> db.zones.insert({_id: "fire", name: "NetherWorld", desc: "Here is terribly hot and fire element unicorns love it."})
WriteResult({ "nInserted" : 1 })
> db.zones.find();
{ "_id" : "earth", "name" : "Plains", "desc" : "Normal enviroment for common unicorns." }
{ "_id" : "water", "name" : "Sea", "desc" : "Underwater enviroment for water element unicorns." }
{ "_id" : "air", "name" : "High Mountains", "desc" : "Place above clouds for unicorns which feel comfortable in the sky." }
{ "_id" : "fire", "name" : "NetherWorld", "desc" : "Here is terribly hot and fire element unicorns love it." }
>
```

```
> db.unicorns.update({name: "Unicrom"}, {$set: {zone: {$ref: "zones", $id: "fire"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Dunx"}, {$set: {zone: {$ref: "zones", $id: "fire"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Ayna"}, {$set: {zone: {$ref: "zones", $id: "water"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Pilot"}, {$set: {zone: {$ref: "zones", $id: "air"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Horny"}, {$set: {zone: {$ref: "zones", $id: "earth"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Kenny"}, {$set: {zone: {$ref: "zones", $id: "earth"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
> db.unicorns.find();
{ "_id" : ObjectId("6571e863c12a7ae675ab7afc"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68, "zone" : DBRef("zones", "earth") }
{ "_id" : ObjectId("6571e870c12a7ae675ab7afd"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7afe"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187, "zone" : DBRef("zones", "fire") }
{ "_id" : ObjectId("6571e878c12a7ae675ab7aff"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b00"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b01"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51, "zone" : DBRef("zones", "water") }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b02"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44, "zone" : DBRef("zones", "earth") }
{ "_id" : ObjectId("6571e878c12a7ae675ab7b03"), "name" : "Raleigh", "loves" : [ "apple", "sugar", "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b04"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b05"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59, "zone" : DBRef("zones", "air") }
{ "_id" : ObjectId("6571e879c12a7ae675ab7b06"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6571e8f1c12a7ae675ab7b07"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170, "zone" : DBRef("zones", "fire") }
{ "_id" : ObjectId("657220cfc12a7ae675ab7b0b"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 4.2.1

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
}
```

Практическое задание 4.3.1

1. Получите информацию обо всех индексах коллекции unicorns
2. Удалите все индексы, кроме индекса для идентификатора
3. Попытайтесь удалить индекс для идентификатора.

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

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

Практическое задание 4.4.1

1. Создайте объёмную выборку коллекцию numbers, задействовав курсор: `for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}`
2. Выберите последние 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 обо 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?

```
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
WriteResult({ "nInserted" : 1 })
```

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

                },
                "maxIndexedOrSolutionsReached" : false,
                "maxIndexedAndSolutionsReached" : false,
                "maxScansToExplodeReached" : false,
                "winningPlan" : {
                        "stage" : "SORT",
                        "sortPattern" : {
                                "value" : -1
                        },
                        "memLimit" : 104857600,
                        "limitAmount" : 4,
                        "type" : "simple",
                        "inputStage" : {
                                "stage" : "COLLSCAN",
                                "direction" : "forward"
                        }
                },
                "rejectedPlans" : [ ]
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 4,
                "executionTimeMillis" : 100,
                "totalKeysExamined" : 0,
                "totalDocsExamined" : 100000,
                "executionStages" : {
```

первый запрос

```
> 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"
        }
]
```

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

                },
                "maxIndexedOrSolutionsReached" : false,
                "maxIndexedAndSolutionsReached" : false,
                "maxScansToExplodeReached" : false,
                "winningPlan" : {
                        "stage" : "LIMIT",
```

```
        },
        "executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 4,
                "executionTimeMillis" : 2,
                "totalKeysExamined" : 4,
                "totalDocsExamined" : 4,
                "executionStages" : {
```

Второй запрос

Запрос с индексом выполнился в разы быстрее чем без него

Вывод

В данной лабораторной работе я поработал с базами данных в MongoDB. Я овладел 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.