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:
«Базы данных»

Выполнила:
студентка II курса ИКТ
группы К3243
Костень Анна Сергеевна

Проверила:
Говорова Марина Михайловна

Санкт-Петербург
2021

Цель лабораторной 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 5.0.8.

Практическое задание

Задание 8.1.1 Вставка документов в коллекцию

Листинг 1 – Создание БД learn и заполнение коллекции unicorns, вставка документа вторым способом

```
> use learn
switched to db learn
> db.create
db.createCollection(  db.createRole(        db.createUser(        db.createView(
> db.createCollection("unicorns")
{ "ok" : 1 }
> db.unicorns.insert({name: 'Aurora', gender: 'f', weight: 450})
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600,
gender: 'm', vampires: 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 })
```

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

```
> db.unicorns.find()
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" : "f", "weight" : 450 }

{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [ "carrot", "papaya" ],
 "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [ "carrot", "grape" ],
 "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [ "energon", "redbull
" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves" : [ "apple" ], "wei
ght" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [ "apple", "carrot",
"chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "name" : "Ayna", "loves" : [ "strawberry", "lemon"
], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "
weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ],
 "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd347"), "name" : "Leia", "loves" : [ "apple", "watermelon"
], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [ "apple", "watermelon"
 ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [ "grape", "carrot" ],
"weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [ "grape", "watermelon"
], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Рисунок 1 – Результат проверки содержимого коллекции с помощью метода find

8.1.2 Выборка данных из БД

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

Листинг 2 – Список мужских особей, отсортированный в алфавитном порядке

```
> db.unicorns.find({gender: "m"}).sort({name: 1})
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

```
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" : [
"apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

Листинг 3 – Первые три записи в списке женских особей, отсортированном по алфавиту

```
> db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3)
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "name" : "Ayna", "loves" : [
"strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
```

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

Листинг 4 – Все самки, которые любят carrot

```
> db.unicorns.find({"loves": "carrot"}).limit(1)
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
> db.unicorns.find({"loves": "carrot", "gender": "f"})
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [
"apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" :
80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

Листинг 5 – Поиск с помощью функции findOne

```
> db.unicorns.findOne({"loves": "carrot", "gender": "f"})
{
"_id" : ObjectId("628f640afc4ad54c59cbd340"),
"name" : "Aurora",
"loves" : [
"carrot",
"grape"
],
"weight" : 450,
"gender" : "f",
"vampires" : 43
}
```

Листинг 6 – Поиск с помощью limit

```
> db.unicorns.find({"loves": "carrot", "gender": "f"}).limit(1)
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

8.1.3 Модификация запросов

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

Листинг 7 – Модифицированный запрос

```
> db.unicorns.find({"gender": "m"}, {"loves": 0, "gender": 0})
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "weight" :
600, "vampires" : 63 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "weight" :
984, "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "weight"
: 575, "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "weight" :
690, "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "weight" :
421, "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "weight" :
650, "vampires" : 54 }
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "weight" : 704,
"vampires" : 165 }
```

8.1.4

Вывести список единорогов в обратном порядке добавления.

Листинг 8 – Обратный список единорогов

```
> db.unicorns.find().sort({$natural: -1})
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd347"), "name" : "Leia", "loves" : [
"apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" : [
"apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "name" : "Ayna", "loves" : [
"strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [
"apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" :
80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

```
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
```

8.1.5

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

Листинг 9 – Результат выполнения 8.1.5

```
> db.unicorns.find({}, {_id: 0, "loves": {$slice : 1}})
{ "name" : "Aurora", "gender" : "f", "weight" : 450 }
{ "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 }
```

8.1.6

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

Листинг 10 – Список единорогов в заданной весовой категории

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

8.1.7

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

8.1.8 Найти всех единорогов, не имеющих ключ vampires.

Листинг 12 – Единороги без ключа vampires

```
> db.unicorns.find({vampires : {$exists: false}})
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

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

Листинг 13 – Искомый список

```
> db.unicorns.find({}, {"loves" : {$slice : 1}}).sort({name: 1})
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "name" : "Ayna", "loves" : [
"strawberry" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd347"), "name" : "Leia", "loves" : [
"apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" : [
"apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [
"apple" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

Выводы по результатам работы

8.2 ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB.

ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ

8.2.1 Запрос к вложенным объектам

Листинг 14 – Создание и заполнение коллекции towns

```
> db.createCollection("towns")
{ "ok" : 1 }
> 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"}}
uncaught exception: SyntaxError: expected property name, got ')' :
@(shell):1:35
> 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 })
```

Листинг 15 – Список городов с независимыми мэрами (только название города и информация о мэре)

```
> db.towns.find({"mayor.party": "I"}, {_id: 0, "population": 0, "last_sensus":
0, "famous_for": 0})
{ "name" : "New York", "populatiuon" : 22200000, "mayor" : { "name" : "Michael
Bloomberg", "party" : "I" } }
```

Из-за опечатки во время заполнения данных население Нью-Йорка все же отображается и не исключается стандартно.

Листинг 16 – Список беспартийных мэров (информация о мэре и название города)

```
> db.towns.find({"mayor.party": {$exists: false}}, {_id: 0 , population: 0,
last_sensus: 0, famous_for: 0})
```

```
{ "name" : "Punxsutawney ", "populatiuon" : 6200, "mayor" : { "name" : "Jim
Wehrle" } }
```

8.2.2

Листинг 17 – Функция для вывода списка самцов единорогов и результат ее применения

```
> func = function() {return this.gender == "m"}
function() {return this.gender == "m"}
> db.unicorns.find(func)
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" : [
"apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Листинг 18 – Курсор для данного списка из первых двух особей с сортировкой в лексикографическом порядке, вывод результата через forEach

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

8.2.3

Листинг 19 – Количество самок единорогов весом от 500 до 600 кг

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

8.2.4 Вывести список предпочтений

Листинг 20 – Список предпочтений

```
> db.unicorns.distinct("loves")
[
"apple",
"carrot",
"chocolate",
"energon",
```

```
"grape",
"lemon",
"papaya",
"redbull",
"strawberry",
"sugar",
"watermelon"
]
```

8.2.5 Посчитать кол-во особей единорогов обоих полов

Листинг 21 – Количество единорогов

```
> db.unicorns.aggregate({$group:{_id:"$gender", count:{$sum:1}}})
{ "_id" : "f", "count" : 6 }
{ "_id" : "m", "count" : 7 }
```

8.2.6

Листинг 22 – Выполнение команды и проверка содержимого коллекции

```
> db.unicorns.save({name: 'Barny', loves: ['grape'],
... weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [
"apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" :
80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "name" : "Ayna", "loves" : [
"strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" : [
"apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd347"), "name" : "Leia", "loves" : [
"apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f788dff32c7ae5c154823"), "name" : "Barny", "loves" : [
"grape" ], "weight" : 340, "gender" : "m" }
```

8.2.7

Листинг 23 – Обновление данных и его результат

```
> db.unicorns.update({name: "Ayna"}, {name: "Ayna", weight: 800, vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({"name": "Ayna"})
{ "_id" : ObjectId("628f7b5eff32c7ae5c154825"), "name" : "Ayna", "weight" : 800,
"vampires" : 51 }
```

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

Листинг 24 – Обновление данных и его результат

```
> db.unicorns.update({name: "Raleigh"}, {$set: {loves: "RedBull"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({"name": "Raleigh"})
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" :
"RedBull", "weight" : 421, "gender" : "m", "vampires" : 2 }
```

8.2.9 Всем самцам единорогов увеличить кол-во убитых вампиров на 5

Листинг 25 – Обновление данных и его результат

```
> db.unicorns.update({gender: "f"}, {$inc: {vampires: 5}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450, "vampires" : 5 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [
"apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" :
80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "weight" : 800, "vampires" : 51
}
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" :
"RedBull", "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd347"), "name" : "Leia", "loves" : [
"apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f788dff32c7ae5c154823"), "name" : "Barny", "loves" : [
"grape" ], "weight" : 340, "gender" : "m" }
{ "_id" : ObjectId("628f7b35ff32c7ae5c154824"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
{ "_id" : ObjectId("628f7b5eff32c7ae5c154825"), "name" : "Ayna", "weight" : 800,
"vampires" : 51 }
```

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

Листинг 26 – Обновление данных и его результат

```
> db.towns.update({name: "Portland"}, {$unset: {mayor: 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find({name: "Portland"})
{ "_id" : ObjectId("628f71d2ff32c7ae5c154822"), "name" : "Portland",
"populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"),
"famous_for" : [ "beer", "food" ] }
```

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

Листинг 27 – Обновление данных и его результат

```
> db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Pilot"})
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires"
: 54 }
```

8.2.12 Изменить информацию о самке единорога Horny: теперь он любит еще и сахар, и лимоны.

Листинг 28 – Обновление данных и его результат

```
> db.unicorns.update({name: "Horny"}, {$addToSet: {loves: {$each: ["sugar",
"lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Horny"})
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya", "sugar", "lemon" ], "weight" : 600, "gender" : "m",
"vampires" : 63 }
```

8.2.13 Удалить документы с беспартийными мэрами, проверить содержание коллекции, удалить коллекцию и посмотреть список доступных коллекций

Листинг 29 – Выполнение вышеперечисленных действий

```
> db.towns.remove({"mayor.party": null}, true)
WriteResult({ "nRemoved" : 1 })
> db.towns.find()
{ "_id" : ObjectId("628f71bbff32c7ae5c154821"), "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("628f71d2ff32c7ae5c154822"), "name" : "Portland",
"populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"),
"famous_for" : [ "beer", "food" ] }
> db.towns.remove({})
WriteResult({ "nRemoved" : 2 })
> show collections
```

```
towns
unicorns
```

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

Листинг 30 – Выполнение вышеперечисленных действий

```
> db.createCollection("envs")
{ "ok" : 1 }
> db.envs.insert(_id: "w", name: "water"})
uncaught exception: SyntaxError: missing ) after argument list :
@(shell):1:18
> db.envs.insert({_id: "w", name: "water"})
WriteResult({ "nInserted" : 1 })
> db.envs.insert({_id: "e", name: "earth"})
WriteResult({ "nInserted" : 1 })
> db.envs.find()
{ "_id" : "w", "name" : "water" }
{ "_id" : "e", "name" : "earth" }
> db.unicorns.update({name: "Ayna"}, {$set:{env{$ref:"env", id: "w"}}})
uncaught exception: SyntaxError: missing : after property id :
@(shell):1:45
> db.unicorns.update({name: "Ayna"}, {$set:{env:{$ref:"env", id: "w"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Horny"}, {$set:{env:{$ref:"env", id: "e"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("628f6314fc4ad54c59cbd33e"), "name" : "Aurora", "gender" :
"f", "weight" : 450, "vampires" : 5, "loves" : [ "sugar", "lemon" ] }
{ "_id" : ObjectId("628f640afc4ad54c59cbd33f"), "name" : "Horny", "loves" : [
"carrot", "papaya", "sugar", "lemon" ], "weight" : 600, "gender" : "m",
"vampires" : 63, "env" : { "$ref" : "env", "id" : "e" } }
{ "_id" : ObjectId("628f640afc4ad54c59cbd340"), "name" : "Aurora", "loves" : [
"carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd341"), "name" : "Unicrom", "loves" : [
"energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd342"), "name" : "Roooooodles", "loves"
: [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd343"), "name" : "Solnara", "loves" : [
"apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" :
80 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd344"), "weight" : 800, "vampires" : 51
}
{ "_id" : ObjectId("628f640afc4ad54c59cbd345"), "name" : "Kenny", "loves" : [
"grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd346"), "name" : "Raleigh", "loves" :
"RedBull", "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd347"), "name" : "Leia", "loves" : [
"apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd348"), "name" : "Pilot", "loves" : [
"apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires"
: 54 }
{ "_id" : ObjectId("628f640afc4ad54c59cbd349"), "name" : "Nimue", "loves" : [
"grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

```
{ "_id" : ObjectId("628f64a9fc4ad54c59cbd34a"), "name" : "Dunx", "loves" : [
"grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("628f788dff32c7ae5c154823"), "name" : "Barny", "loves" : [
"grape" ], "weight" : 340, "gender" : "m" }
{ "_id" : ObjectId("628f7b35ff32c7ae5c154824"), "name" : "Aurora", "gender" :
"f", "weight" : 450 }
{ "_id" : ObjectId("628f7b5eff32c7ae5c154825"), "name" : "Ayna", "weight" : 800,
"vampires" : 51, "env" : { "$ref" : "env", "id" : "w" } }
```

8.3.2 Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique

Листинг 31 – Получается, нельзя

```
> db.unicorns.ensureIndex({"name": 1}, {"unique": true})
uncaught exception: TypeError: db.unicorns.ensureIndex is not a function :
@(shell):1:1
```

8.3.3 Получить информацию о всех индексах коллекции unicorns, удалить все индексы (кроме индекса для идентификатора), попытаться удалить индекс для идентификатора

Листинг 32 – Попытка не пытка

```
> db.unicorns.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
> db.users.dropIndex("name")
{
"ok" : 0,
"errmsg" : "ns not found learn.users",
"code" : 26,
"codeName" : "NamespaceNotFound"
}
> db.users.dropIndex("key")
{
"ok" : 0,
"errmsg" : "ns not found learn.users",
"code" : 26,
"codeName" : "NamespaceNotFound"
}
> db.users.dropIndex("_id")
{
"ok" : 0,
"errmsg" : "ns not found learn.users",
"code" : 26,
"codeName" : "NamespaceNotFound"
}
```

8.3.4

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

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

Листинг 33 – ананлиз выполнения 2 запроса

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

},
"maxIndexedOrSolutionsReached" : false,
"maxIndexedAndSolutionsReached" : false,
"maxScansToExplodeReached" : false,
"winningPlan" : {
"stage" : "EOF"
},
"rejectedPlans" : [ ]
},
"executionStats" : {
"executionSuccess" : true,
"nReturned" : 0,
"executionTimeMillis" : 2,
```

На выполнение данного запроса понадобилось 2 мс.

5. Создайте индекс для ключа value.

Листинг 35 – Создание индекса

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

6. Получите информацию о всех индексах коллекции numbres.
7. Выполните запрос 2.
8. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

Листинг 36 – Анализ выполнения запроса с индексом

```
> db.numbers.explain("executionStats").find().sort({natural:-1}).limit(4)
{
"explainVersion" : "1",
"queryPlanner" : {
"namespace" : "test.numbers",
```

```
"indexFilterSet" : false,
"parsedQuery" : {

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


Таким образом, с индексом время выполнения данного запроса стремится к 0.

9. Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

Запрос с индексом более эффективен, даже на таком небольшой объеме данных он уменьшил время выполнения с 2 до почти 0 мс.

Вывод: я изучила основы MongoDB, выполнив ряд практических заданий. Синтаксис данной СУБД оказался довольно удобным и простым, несмотря на то, как сильно он отличается от того, что я изучила ранее.