

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное  
учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет прикладной информатики

## **ОТЧЕТ**

**по лабораторной работе № 6 на тему:**

**«Работа с БД в СУБД MongoDB»**

Группа: K3239

ФИО: Никифоров Максим  
Александрович

Проверила: М.М. Говорова

**Цель:** 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+, 8.0.4 (последняя).

## Практическое задание 2.1.1

Создание базы данных и заполнение коллекции

```
}
WriteResult({ "nInserted" : 1 })
{ "_id" : ObjectId("6853d60914ee56412e3ff341"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6853d60914ee56412e3ff342"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6853d60914ee56412e3ff343"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6853d60914ee56412e3ff344"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6853d60914ee56412e3ff345"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6853d60914ee56412e3ff346"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6853d60914ee56412e3ff347"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6853d60914ee56412e3ff348"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6853d60914ee56412e3ff349"), "name" : "Ieia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6853d60914ee56412e3ff34a"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6853d60914ee56412e3ff34b"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6853d60914ee56412e3ff34c"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6853d60914ee56412e3ff34d"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6853d60914ee56412e3ff34e"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6853d60914ee56412e3ff34f"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6853d60914ee56412e3ff350"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6853d60914ee56412e3ff351"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6853d60914ee56412e3ff352"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6853d60914ee56412e3ff353"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6853d60914ee56412e3ff354"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
```

## Практическое задание 2.2.1

Сформируйте запросы для вывода списков самцов и самок единорогов.

Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени. Найдите всех самок, которые любят carrot.

Ограничьте этот список первой особью с помощью функций findOne и limit

Команды MongoDB:

```
db.unicorns.find({ gender: 'm' }).sort({ name: 1 })
```

```
db.unicorns.find({ gender: 'f' }).sort({ name: 1 }).limit(3)
```

```
db.unicorns.find({ gender: 'f', loves: 'carrot' }).limit(1)
```

```
db.unicorns.findOne({ gender: 'f', loves: 'carrot' })
```

```

WriteResult({ "nInserted" : 1 })
  "_id" : ObjectId("6853dd30f57869ce9f55f04d"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
  "_id" : ObjectId("6853dd30f57869ce9f55f042"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
  "_id" : ObjectId("6853dd30f57869ce9f55f048"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
  "_id" : ObjectId("6853dd30f57869ce9f55f04b"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
  "_id" : ObjectId("6853dd30f57869ce9f55f049"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
  "_id" : ObjectId("6853dd30f57869ce9f55f045"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
  "_id" : ObjectId("6853dd30f57869ce9f55f044"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
  "_id" : ObjectId("6853dd30f57869ce9f55f043"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
  "_id" : ObjectId("6853dd30f57869ce9f55f047"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
  "_id" : ObjectId("6853dd30f57869ce9f55f04a"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }

  "_id" : ObjectId("6853dd30f57869ce9f55f043"),
  "name" : "Aurora",
  "loves" : [
    "carrot",
    "grape"
  ],
  "weight" : 450,
  "gender" : "f",
  "vampires" : 43

  "_id" : ObjectId("6853dd30f57869ce9f55f043"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }

```

## Практическое задание 2.2.2

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

Команды MongoDB:

```
db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0})
```

```

{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
{ "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
{ "name" : "Rooodooles", "weight" : 575, "vampires" : 99 }
{ "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "name" : "Dunx", "weight" : 704, "vampires" : 165 }

```

## Практическое задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

Команды MongoDB:

```
db.unicorns.find().sort({$natural: -1})
```

```

}
WriteResult({ "nInserted" : 1 })
{ "_id" : ObjectId("6853dda9f6f9566d87a9f610"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60f"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60e"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60d"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60c"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60b"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f609"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f608"), "name" : "Roocoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f607"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f606"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f605"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }

```

## Практическое задание 2.2.4

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

Команды MongoDB:

```
db.unicorns.find({ }, {loves: { $slice: 1 }, _id: 0})
```

```

WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
{ "name" : "Horny", "loves" : [ "carrot" ] }
{ "name" : "Aurora", "loves" : [ "carrot" ] }
{ "name" : "Unicrom", "loves" : [ "energon" ] }
{ "name" : "Rooooooodles", "loves" : [ "apple" ] }
{ "name" : "Solnara", "loves" : [ "apple" ] }
{ "name" : "Ayna", "loves" : [ "strawberry" ] }
{ "name" : "Kenny", "loves" : [ "grape" ] }
{ "name" : "Raleigh", "loves" : [ "apple" ] }
{ "name" : "Leia", "loves" : [ "apple" ] }
{ "name" : "Pilot", "loves" : [ "apple" ] }
{ "name" : "Nimue", "loves" : [ "grape" ] }
{ "name" : "Dunx", "loves" : [ "grape" ] }

```

### Практическое задание 2.3.1

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

Команды MongoDB:

```
db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 700}}, {_id: 0})
```



```

WriteResult({ "nInserted" : 1 })
{ "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" }
{ "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 2.3.2

Вывести список самцов единорогов весом от полутонны и предпочитающих грейп и лимон, исключив вывод идентификатора.

Команды MongoDB:

```
db.unicorns.find({gender: 'm', weight: {$gte: 500}, loves: {$all: ['grape', 'lemon']}}, {_id: 0})
```

```

{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }

```

### Практическое задание 2.3.3

Найти всех единорогов, не имеющих ключ vampires.

Команды MongoDB:

```
db.unicorns.find({vampires: {$exists: false}})
```

```

WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
{ "_id" : ObjectId("6853de3672aba668918d978d"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }

```

### Практическое задание 2.3.4

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

Команды MongoDB:

```
db.unicorns.find({gender: 'm'}, {name: 1, loves: {$slice: 1}, _id: 0}).sort({name: 1})
```

```
WriteResult({ "nInserted" : 1 })
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
{ "name" : "Dunx", "loves" : [ "grape" ] }
{ "name" : "Horny", "loves" : [ "carrot" ] }
{ "name" : "Kenny", "loves" : [ "grape" ] }
{ "name" : "Pilot", "loves" : [ "apple" ] }
{ "name" : "Raleigh", "loves" : [ "apple" ] }
{ "name" : "Rooooooodles", "loves" : [ "apple" ] }
{ "name" : "Unicrom", "loves" : [ "energon" ] }
```

### Практическое задание 3.1.1

Создайте коллекцию towns и выполните выборки по мэрам с party="I" и без party.

Команды MongoDB:

```
db.towns.insert({...})
```

```
db.towns.find({"mayor.party": "I"}, {name: 1, mayor: 1})
```

```
db.towns.find({"mayor.party": {$exists: false}}, {name: 1, mayor: 1})
```

```
switched to db learn
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
{ "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "name" : "Punxsutawney", "mayor" : { "name" : "Jim Wehrle" } }
```

### Практическое задание 3.1.2

Сформировать функцию для вывода списка самцов единорогов и вывести первых двух.

Команды MongoDB:

```
var cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2);
cursor.forEach(function(unicorn) { print(unicorn.name); });
```

```
function() { return this.gender == "m"; }
{ "_id" : ObjectId("6853df1de7ae6ebf3607810b"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("6853df1de7ae6ebf36078100"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

### Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

Команды MongoDB:

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

```
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
2
```

### Практическое задание 3.2.2

Вывести список предпочтений.



Команды MongoDB:

```
db.unicorns.distinct("loves")
```

```
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
[
  "apple",
  "carrot",
  "chocolate",
  "energon",
  "grape",
  "lemon",
  "papaya",
  "redbull",
  "strawberry",
  "sugar",
  "watermelon"
]
```

### Практическое задание 3.2.3

Посчитать количество особей единорогов обоих полов.

Команды MongoDB:

```
db.unicorns.aggregate([{$group: {_id: "$gender", count: {$sum: 1}}}}])
```

```
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
},
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
[
  "_id" : "m", "count" : 7 }
  "_id" : "f", "count" : 5 }
```

### Практическое задание 3.3.1

Добавить самца Barny.

Команды MongoDB:

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

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

```
WriteResult({ "nInserted" : 1 })
{
  "acknowledged" : true,
  "insertedId" : ObjectId("6853e01b1a88a76cce64dc25")
}
{ "_id" : ObjectId("6853dda9f6f9566d87a9f605"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f606"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6853dda9f6f9566d87a9f607"), "name" : "Unicorn", "loves" : [ "eggplant", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

### Практическое задание 3.3.2

Обновить Айна: вес 800, вампиры 51.

Команды MongoDB:

```
db.unicorns.update({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
```

```
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "_id" : ObjectId("6853dd30f57869ce9f55f047"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 800, "gender" : "f", "vampires" : 51 }
{ "_id" : ObjectId("6853e03b286ae450ac95b49f"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
```

### Практическое задание 3.3.3

Обновить Raleigh: добавить redbull в loves.

Команды MongoDB:

```
db.unicorns.update({name: "Raleigh"}, {$push: {loves: "redbull"}})
```

```
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "_id" : ObjectId("6853e03b286ae450ac95b49f"), "name" : "Raleigh", "loves" : [ "apple", "sugar", "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
```

### Практическое задание 3.3.4

Увеличить количество убитых вампиров у всех самцов на 5.

Команды MongoDB:

```
db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true})
```

```
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nMatched" : 7, "nUpserted" : 0, "nModified" : 7 })
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8dd"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8df"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8e0"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8e3"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8e4"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8e6"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("6853e0d6aa616b4eef13b8e8"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
```

### Практическое задание 3.3.5

Убрать партию у мэра Портланда.

Команды MongoDB:

```
db.towns.update({name: "Portland"}, {$unset: {"mayor.party": 1}})
```

```
riteResult({ "nInserted" : 1 })
riteResult({ "nInserted" : 1 })
riteResult({ "nInserted" : 1 })
riteResult({ "nInserted" : 1 })
riteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "_id" : ObjectId("6853def4134daac4d5b897cf"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "
{ "_id" : ObjectId("6853e0edf547c1474e89cc1f"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "
```

### Практическое задание 3.3.6

Обновить Pilot: добавить chocolate в loves.

Команды MongoDB:

```
db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
```

```
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "_id" : ObjectId("6853dda9f6f9566d87a9f60e"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6853e01b1a88a76cce64dc1e"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("6853e1014afd47a74660cd5f"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
```

### Практическое задание 3.3.7

Обновить Aurora: добавить sugar и lemon в loves.

Команды MongoDB:

```
db.unicorns.update({name: "Aurora"}, {$addToSet: {loves: {$each:
["sugar", "lemon"]}}})
```

```
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "_id" : ObjectId("6853d6176fbf7baf58057a89"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6853dfad08b82314a5c8da13"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("6853e1183ef70ac36ae3ec77"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

### Практическое задание 3.4.1

Удалить беспартийных мэров, очистить коллекцию, просмотреть коллекции.

Команды MongoDB:

```
db.towns.remove({"mayor.party": {$exists: false}})
```

```
db.towns.remove({})
```

```
show collections
```

```
iteResult({ "nRemoved" : 3 })
{"_id" : ObjectId("6853dff30dacb8164bc2d9b0"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
_id" : ObjectId("6853dff30dacb8164bc2d9b1"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
_id" : ObjectId("6853e1330aaa7d58ed3fd4f5"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
_id" : ObjectId("6853e1330aaa7d58ed3fd4f6"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
_id" : ObjectId("6853e1330aaa7d58ed3fd4f8"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "st
_id" : ObjectId("6853e1330aaa7d58ed3fd4f9"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer
iteResult({ "nRemoved" : 6 })
```

### Практическое задание 4.1.1

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.

Команды MongoDB:

```
db.habitats.insert({_id: "lakes", name: "Lakes"});
```

```
db.habitats.insert({_id: "hills", name: "Hills"});
```

```
db.unicorns.update({name: "Horny"}, {$set: {habitat: {$ref: "habitats", $id: "lakes"}}});
```

```
db.unicorns.update({name: "Kenny"}, {$set: {habitat: {$ref: "habitats", $id: "hills"}}});
```

```
db.unicorns.find();
```

```
learn> db.habitats.insert({_id: "lakes", name: "Lakes"});
{ acknowledged: true, insertedIds: { '0': 'lakes' } }
learn> db.habitats.insert({_id: "hills", name: "Hills"});
{ acknowledged: true, insertedIds: { '0': 'hills' } }
```

### Практическое задание 4.2.1

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



Команды MongoDB:

```
db.unicorns.createIndex({name: 1}, {unique: true})
```

```
[learn> db.unicorns.createIndex({name: 1}, {unique: true})  
name_1
```

### Практическое задание 4.3.1

Получите информацию о всех индексах коллекции unicorns. Удалите все индексы, кроме индекса для идентификатора. Попытайтесь удалить индекс для идентификатора.

Команды MongoDB:

```
db.unicorns.getIndexes();  
db.unicorns.dropIndexes();  
db.unicorns.dropIndex("_id_");
```

```
[learn> db.unicorns.getIndexes();  
[  
  { v: 2, key: { _id: 1 }, name: '_id_' },  
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }  
]  
[learn> db.unicorns.dropIndex("name_1");  
{ nIndexesWas: 2, ok: 1 }  
[learn> db.unicorns.dropIndex("_id_");  
MongoServerError[InvalidOptions]: cannot drop _id index  
learn> █
```

### Практическое задание 4.4.1

Создайте объемную коллекцию numbers, задействовав курсор. Выберите последние четыре документа. Проанализируйте план выполнения запроса. Сколько потребовалось времени на выполнение запроса? Создайте индекс для ключа value. Получите информацию о всех индексах коллекции numbers. Выполните запрос 2. Проанализируйте план выполнения запроса с установленным индексом. Сравните время выполнения запросов с индексом и без.

Команды MongoDB:

```
for(i = 0; i < 100000; i++){ db.numbers.insert({ value: i }) }  
db.numbers.find().sort({ $natural: -1 }).limit(4)  
db.numbers.explain("executionStats").find().sort({ $natural: -1 }).limit(4)  
db.numbers.createIndex({ value: 1 })  
db.numbers.getIndexes()  
db.numbers.explain("executionStats").find().sort({ $natural: -1 }).limit(4)
```

Результат с индексом:

```
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BA27D965',
    planCacheShapeHash: 'BA27D965',
    planCacheKey: '7A892B81',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: {
        stage: 'FETCH',
        inputStage: {
          stage: 'IXSCAN',
          keyPattern: { value: 1 },
          indexName: 'value_1',
          isMultiKey: false,
          multiKeyPaths: { value: [ ] },
          isUnique: false,
          isSparse: false,
          isPartial: false,
          indexVersion: 2,
          direction: 'backward',
          indexBounds: { value: [ '[MaxKey, MinKey]' ] }
        }
      }
    },
    rejectedPlans: [ ]
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 0,
    totalKeysExamined: 4,
    totalDocsExamined: 4,
    executionStages: {
      isCached: false,
      stage: 'LIMIT',
      nReturned: 4,
      executionTimeMillisEstimate: 0,
      works: 5,
      advanced: 4,
      needTime: 0,
      needYield: 0,
      ...
    }
  }
}
```

Результат без индекса:

```
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BA27D965',
    planCacheShapeHash: 'BA27D965',
    planCacheKey: '7A892B81',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'SORT',
      sortPattern: { value: -1 },
      memLimit: 33554432,
      limitAmount: 4,
      type: 'simple',
      inputStage: { stage: 'COLLSCAN', direction: 'forward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 3,
    totalKeysExamined: 0,
    totalDocsExamined: 4042,
    executionStages: {
      isCached: false,
      stage: 'SORT',
      nReturned: 4,
      executionTimeMillisEstimate: 2,
      works: 4048,
      advanced: 4,
      needTime: 4043,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      sortPattern: { value: -1 },
      memLimit: 33554432,
      limitAmount: 4,
      type: 'simple',
      totalDataSizeSorted: 260,
      usedDisk: false,
      spills: 0,
      spilledDataStorageSize: 0,
      inputStage: {
        stage: 'COLLSCAN',
        nReturned: 4042,
        executionTimeMillisEstimate: 0
      }
    }
  }
}
```

Без индекса происходит полный просмотр коллекции, который занял  
Итого 4 мс executionTimeMillis снизился с приблизительно 1 до 0.7  
мс.

## Выводы

В ходе выполнения лабораторной работы были подробно изучены и практически освоены основные возможности работы с базой данных MongoDB. В частности, были рассмотрены и реализованы все основные операции CRUD: вставка, выборка, изменение и удаление документов. Это позволило получить представление о том, как осуществляется базовое взаимодействие с коллекциями и документами в MongoDB, а также как можно гибко управлять данными на уровне отдельных записей.

Также была проведена работа с вложенными документами и массивами,

что является одной из ключевых особенностей MongoDB как документо-ориентированной базы данных. Были рассмотрены способы обращения к вложенным полям, фильтрации и обновления данных внутри сложных структур, что позволяет моделировать реальные объекты и их связи максимально естественно и эффективно.