

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 8

по теме: Работа с БД в СУБД MongoDB по дисциплине: Проектирование и реализация баз данных
по дисциплине: Проектирование и реализация баз данных

Специальность:

09.03.03 Мобильные и сетевые технологии

Проверил:
Говорова М.М. _____
Дата: « » Июня 2021г.
Оценка _____

Выполнил(и):
студент(ы) группы
К3241
Касаткин Д.А.

Санкт-Петербург 2021 г.

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.

ПРАКТИЧЕСКАЯ ЧАСТЬ

8.1 CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ

8.1.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

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

```
Activities Terminal
Ср июн 9 22:16:50
daniil@RedmiBook-14: ~
> 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: 'Leila', 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.users.insert(doc)
WriteResult({"nInserted" : 1 })
> db.users.find()
{ "_id" : ObjectId("60c1136b33a783f13bf6481a"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
> db.unicorns.find()
{ "_id" : ObjectId("60c1130133a783f13bf6480f"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60c1130933a783f13bf64810"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60c1130f33a783f13bf64811"), "name" : "Unicron", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60c1131333a783f13bf64812"), "name" : "Roosoooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60c1131733a783f13bf64813"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60c1131a33a783f13bf64814"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60c1131e33a783f13bf64815"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60c1132133a783f13bf64816"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60c1132433a783f13bf64817"), "name" : "Leila", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60c1132833a783f13bf64818"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60c1132b33a783f13bf64819"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

8.2.2. ВЫБОРКА ДАННЫХ ИЗ БД

Практическое задание 8.1.2:

1. Сформируйте запросы для вывода списков самцов и самок единорогов.

Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.

```
> db.unicorns.find((gender:'m')).sort((name:1))
{ "_id" : ObjectId("60c113c233a783f13bf6481b"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60c1130133a783f13bf6480f"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60c1131e33a783f13bf64815"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60c1132833a783f13bf64818"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60c1132133a783f13bf64816"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60c1131333a783f13bf64812"), "name" : "Rooodoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60c1130f33a783f13bf64811"), "name" : "Unicron", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
> db.unicorns.find((gender:'f')).limit(3).sort((name:1))
{ "_id" : ObjectId("60c1130933a783f13bf64810"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60c1131a33a783f13bf64814"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60c1132433a783f13bf64817"), "name" : "Lela", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
```

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

```
> db.unicorns.findOne((gender:'f', loves:'carrot'))
{
  "_id" : ObjectId("60c1130933a783f13bf64810"),
  "name" : "Aurora",
  "loves" : [
    "carrot",
    "grape"
  ],
  "weight" : 450,
  "gender" : "f",
  "vampires" : 43
}
```

Практическое задание 8.1.3:

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

```
> db.unicorns.find((gender:'m'), {gender:0, loves:0})
{ "_id" : ObjectId("60c1130133a783f13bf6480f"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("60c1130f33a783f13bf64811"), "name" : "Unicron", "weight" : 984, "vampires" : 182 }
```

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural:-1})
{ "_id" : ObjectId("60c113c233a783f13bf6481b"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60c1132b33a783f13bf64819"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60c1132833a783f13bf64818"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60c1132433a783f13bf64817"), "name" : "Lela", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60c1132133a783f13bf64816"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60c1131e33a783f13bf64815"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60c1131a33a783f13bf64814"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60c1131333a783f13bf64813"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60c1130933a783f13bf64810"), "name" : "Rooodoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60c1130f33a783f13bf64811"), "name" : "Unicron", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60c1130133a783f13bf6480f"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

Практическое задание 8.1.5:

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор

```
db.unicorns.find({}, {id:0, loves:{$slice:1}})
{ "name": "Horny", "loves": [ "carrot" ], "weight": 600, "gender": "m", "vampires": 63 }
{ "name": "Aurora", "loves": [ "carrot" ], "weight": 450, "gender": "f", "vampires": 43 }
{ "name": "Unicron", "loves": [ "energon" ], "weight": 984, "gender": "m", "vampires": 182 }
{ "name": "Roostoodles", "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": "n", "vampires": 54 }
{ "name": "Nlue", "loves": [ "grape" ], "weight": 540, "gender": "f" }
{ "name": "Dunx", "loves": [ "grape" ], "weight": 704, "gender": "m", "vampires": 165 }
```

8.2.3.

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 8.1.6:

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

```
db.unicorns.find({gender:'f', weight:{$gte:500, $lte:700}}, {id:0})
{ "name": "Solnara", "loves": [ "apple", "carrot", "chocolate" ], "weight": 550, "gender": "f", "vampires": 80 }
{ "name": "Leia", "loves": [ "apple", "watermelon" ], "weight": 601, "gender": "f", "vampires": 33 }
{ "name": "Nlue", "loves": [ "grape", "carrot" ], "weight": 540, "gender": "f" }
> db.users.find({gender:'m', weight:{$gte:500}, loves:{$all:['grape','lemon']}}, {id:0});
> db.unicorns.find({gender:'m', weight:{$gte:500}, loves:{$all:['grape','lemon']}}, {id:0});
{ "name": "Kenny", "loves": [ "grape", "lemon" ], "weight": 690, "gender": "m", "vampires": 39 }
```

Практическое задание 8.1.7:

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

```
danill@RedmiBook-14: ~
{ "name": "Solnara", "loves": [ "apple", "carrot", "chocolate" ], "weight": 550, "gender": "f", "vampires": 80 }
{ "name": "Leia", "loves": [ "apple", "watermelon" ], "weight": 601, "gender": "f", "vampires": 33 }
{ "name": "Nlue", "loves": [ "grape", "carrot" ], "weight": 540, "gender": "f" }
> db.users.find({gender:'m', weight:{$gte:500}, loves:{$all:['grape','lemon']}}, {id:0});
> db.unicorns.find({gender:'m', weight:{$gte:500}, loves:{$all:['grape','lemon']}}, {id:0});
{ "name": "Kenny", "loves": [ "grape", "lemon" ], "weight": 690, "gender": "m", "vampires": 39 }
```

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
db.unicorns.find({vampires:{$exists:true}})
{ "_id": ObjectId("60c1130133a783f13bf6480f"), "name": "Horny", "loves": [ "carrot", "papaya" ], "weight": 600, "gender": "m", "vampires": 63 }
{ "_id": ObjectId("60c1130933a783f13bf64810"), "name": "Aurora", "loves": [ "carrot", "grape" ], "weight": 450, "gender": "f", "vampires": 43 }
{ "_id": ObjectId("60c1130f33a783f13bf64811"), "name": "Unicron", "loves": [ "energon", "redbull" ], "weight": 984, "gender": "m", "vampires": 182 }
{ "_id": ObjectId("60c1131333a783f13bf64812"), "name": "Roostoodles", "loves": [ "apple" ], "weight": 575, "gender": "m", "vampires": 99 }
{ "_id": ObjectId("60c1131733a783f13bf64813"), "name": "Solnara", "loves": [ "apple", "carrot", "chocolate" ], "weight": 550, "gender": "f", "vampires": 80 }
{ "_id": ObjectId("60c1131a33a783f13bf64814"), "name": "Ayna", "loves": [ "strawberry", "lemon" ], "weight": 733, "gender": "f", "vampires": 40 }
{ "_id": ObjectId("60c1131e33a783f13bf64815"), "name": "Kenny", "loves": [ "grape", "lemon" ], "weight": 690, "gender": "m", "vampires": 39 }
{ "_id": ObjectId("60c1132133a783f13bf64816"), "name": "Raleigh", "loves": [ "apple", "sugar" ], "weight": 421, "gender": "m", "vampires": 2 }
{ "_id": ObjectId("60c1132433a783f13bf64817"), "name": "Leia", "loves": [ "apple", "watermelon" ], "weight": 601, "gender": "f", "vampires": 33 }
{ "_id": ObjectId("60c1132833a783f13bf64818"), "name": "Pilot", "loves": [ "apple", "watermelon" ], "weight": 650, "gender": "n", "vampires": 54 }
{ "_id": ObjectId("60c1132c33a783f13bf64819"), "name": "Dunx", "loves": [ "grape", "watermelon" ], "weight": 704, "gender": "m", "vampires": 165 }
```

Практическое задание 8.1.9:

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

```
@(shell):1:73
> db.unicorns.find({gender:'m'}, {name:1,_id:0,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" : "Rooooooodles", "loves" : [ "apple" ] }
{ "name" : "Unicorn", "loves" : [ "energon" ] }
```

8.2 ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB.

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

8.2.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 8.2.1:

1. Создайте коллекцию towns, включающую следующие документы:

```
towns
> db.towns.insert({name: "Punxsutawney ",
... population: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...   name: "Jim Wehrle"
... }}
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "New York",
... population: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...   name: "Michael Bloomberg",
...   party: "I"}
... })
WriteResult({ "nInserted" : 1 })
```

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

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

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

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

8.2.2 ИСПОЛЬЗОВАНИЕ JAVASCRIPT

8.2.3 КУРСОРЫ

Практическое задание 8.2.2:

1. Сформировать функцию для вывода списка самцов единорогов.


```
> foo=function(){return this.gender=='m';}
function(){return this.gender=='m';}
```

2. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.

```
> var cur=db.unicorns.find(foo);null;
null
> cur.limit(2).sort({name:1})
{ "_id" : ObjectId("60c113c233a783f13bf6481b"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60c1130133a783f13bf6480f"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

3. Вывести результат, используя *forEach*.

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

8.2.4 АГРЕГИРОВАННЫЕ ЗАПРОСЫ

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 8.2.4:

Вывести список предпочтений.

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

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

```
> db.users.aggregate([{$group:{_id:"gender", count:{$sum:1}}}]
{ "_id" : "gender", "count" : 12 }
```

8.2.5 РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 8.2.6:

1. Выполнить команду:

2. Проверить содержимое коллекции users.

```
> db.users.save({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.users.find()
{ "_id" : ObjectId("60b69b6ff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b69c8a62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooodoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b69cee62e607a3e9f4b822"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b69d4562e607a3e9f4b828"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b7a7375dc408ebab6796da"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 8.2.7:

1. Для самки единорога Айна внести изменения в БД: теперь ее вес 800, она убила 51вампира.
2. Проверить содержимое коллекции users.

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

Практическое задание 8.2.8:

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

```
> db.unicorns.update({name: 'Raleigh', gender: 'm'}, {$set: {loves: ['redbull']}})
> db.unicorns.find({name: 'Raleigh'})
{ "_id" : ObjectId("60c11f2b7e2c3e34192e4e0e"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
```

Практическое задание 8.2.9:

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

```
> db.unicorns.update({gender: 'm'}, {$inc: {vampires: 5}}, {multi: true})
WriteResult({ "nMatched" : 7, "nUpserted" : 0, "nModified" : 7 })
> db.unicorns.find({gender: 'm'})
{ "_id" : ObjectId("60c11f147e2c3e34192e4e07"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("60c11f1c7e2c3e34192e4e09"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60c11f1f7e2c3e34192e4e0a"), "name" : "Rooodoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("60c11f287e2c3e34192e4e0d"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("60c11f2b7e2c3e34192e4e0e"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("60c11f317e2c3e34192e4e10"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("60c120b57e2c3e34192e4e12"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое задание 8.2.10:

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

```
> db.towns.update({name: 'Portland'}, {$unset: {mayor.party: 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Практическое задание 8.2.11:

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

```
> db.unicorns.find({name: 'Pilot'})
{ "_id" : ObjectId("60c11f317e2c3e34192e4e10"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
```

Практическое задание 8.2.12:

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

```
> db.unicorns.update({name: 'Aurora'}, {$addToSet: {loves: {$_each: {sugar, lemon}}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: 'Aurora'})
{ "_id" : ObjectId("60c11f197e2c3e34192e4e08"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

8.2.6

УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

Практическое задание 8.2.13:

1. Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.insert({name: "Punxsutawney ",
... population: 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",
... population: 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",
... population: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...   name: "Sam Adams",
...   party: "D"}})
WriteResult({ "nInserted" : 1 })
```

2. Удалите документы с беспартийными мэрами.

```
> db.towns.remove({'mayor.party':{$exists:false}})
WriteResult({ "nRemoved" : 3 })
```


8.3 ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

8.3.1 ССЫЛКИ В БД

Практическое задание 8.3.1:

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

```
db.areas.insert({_id:'one', name:'First area', desc : 'Exsist'})
```

```
> db.areas.insert({_id:'two', name:'Second area', desc : 'Exsist'})
```

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

3. Проверьте содержание коллекции единорогов.

```
db.unicorns.update({name:'Leia'},{$set:{area:{ $ref:'areas', $id:'one', $type:'zone'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name:'Solnara'},{$set:{area:{ $ref:'areas', $id:'one'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name:'Kenny'},{$set:{area:{ $ref:'areas', $id:'two'}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("60c11f147e2c3e34192e4e07"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "n", "vampires" : 68 }
{ "_id" : ObjectId("60c11f197e2c3e34192e4e08"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60c11f1c7e2c3e34192e4e09"), "name" : "Unicron", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60c11f1f7e2c3e34192e4e0a"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "n", "vampires" : 104 }
{ "_id" : ObjectId("60c11f227e2c3e34192e4e0b"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80, "area" : DBRef("areas", "one") }
{ "_id" : ObjectId("60c11f257e2c3e34192e4e0c"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "gender" : "f", "weight" : 800, "vampires" : 51 }
{ "_id" : ObjectId("60c11f287e2c3e34192e4e0d"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44, "area" : DBRef("areas", "two") }
{ "_id" : ObjectId("60c11f2b7e2c3e34192e4e0e"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "n", "vampires" : 7 }
{ "_id" : ObjectId("60c11f2e7e2c3e34192e4e0f"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 681, "gender" : "f", "vampires" : 33, "area" : DBRef("areas", "one") }
{ "_id" : ObjectId("60c11f317e2c3e34192e4e10"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "n", "vampires" : 59 }
{ "_id" : ObjectId("60c11f347e2c3e34192e4e11"), "name" : "Nitue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60c120b57e2c3e34192e4e12"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "n", "vampires" : 5 }
{ "_id" : ObjectId("60c122657e2c3e34192e4e13"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60c13938529fa40e365f2896"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "n", "vampires" : 63 }
{ "_id" : ObjectId("60c1393e529fa40e365f2897"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60c1393e529fa40e365f2898"), "name" : "Unicron", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60c13944529fa40e365f2899"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60c13950529fa40e365f289a"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60c1396c529fa40e365f289b"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60c13978529fa40e365f289c"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "n", "vampires" : 2 }
```

8.3.2

НАСТРОЙКА ИНДЕКСОВ

Практическое задание 8.3.2:

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

```
db.unicorns.ensureIndex({'name':1},{unique:true})
{
  "ok" : 0,
  "errmsg" : "Index build failed: d73f4657-63dd-4d11-b298-9bc4a5ac25a1: Collection test.unicorns ( f4706196-91c4-43e8-b09b-b25bfe351720 ) :: caused by :: E11000 duplicate key error col
lection: test.unicorns index: name_1 dup key: { name: \"Aurora\" }",
  "code" : 11000,
  "codeName" : "DuplicateKey",
  "keyPattern" : {
    "name" : 1
  },
  "keyValue" : {
    "name" : "Aurora"
  }
}
```

8.3.3

УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 8.3.3:

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

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

2. Удалите все индексы, кроме индекса для идентификатора.

```
> db.unicorns.dropIndex('name_1')
{
  "ok" : 0,
  "errmsg" : "index not found with name [name_1]",
  "code" : 27,
  "codeName" : "IndexNotFound"
}
```

3. Попробуйте удалить индекс для идентификатора.

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

8.3.4 ПЛАН ЗАПРОСА

Практическое задание 8.3.4:

1. Создайте объемную коллекцию *numbers*, задействовав курсор:

```
for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
```

2. Выберите последних четыре документа.

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

4. Создайте индекс для ключа *value*.

5. Получите информацию о 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?

```
> db.numbers.explain('executionStats').find({value:{$gte:99991}})
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "test.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "value" : {
        "$gte" : 99991
      }
    },
    "winningPlan" : {
      "stage" : "EOF"
    },
    "rejectedPlans" : [ ]
  },
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 0,
    "executionTimeMillis" : 0,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 0,
    "executionStages" : {
      "stage" : "EOF",
      "nReturned" : 0,
      "executionTimeMillisEstimate" : 0,
      "works" : 1,
      "advanced" : 0,
      "needTime" : 0,
      "needYield" : 0,
      "saveState" : 0,
      "restoreState" : 0,
      "isEOF" : 1
    }
  },
  "serverInfo" : {
    "host" : "RedmiBook-14",
    "port" : 27017,
    "version" : "4.4.6",
    "gitVersion" : "72e66213c2c3eab37d9358d5e78ad7f5c1d0d0d7"
  },
  "msg" : ""
}
```

```

> db.numbers.createIndex({'value':1})
{
  "createdCollectionAutomatically" : true,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
> db.numbers.explain('executionStats').find({'value':{'$gte:99991}})
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "test.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "value" : {
        "$gte" : 99991
      }
    },
    "winningPlan" : {
      "stage" : "FETCH",
      "inputStage" : {
        "stage" : "IXSCAN",
        "keyPattern" : {
          "value" : 1
        },
        "indexName" : "value_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
          "value" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : false,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
          "value" : [

```

Activities Terminal 4т июн 10 01:20:40 danil@RedmiBook-14: ~

```

    "value" : [
      "99991.0, inf.0]"
    ]
  },
  "rejectedPlans" : [ ]
},
"executionStats" : {
  "executionSuccess" : true,
  "nReturned" : 0,
  "executionTimeMillis" : 1,
  "totalKeysExamined" : 0,
  "totalDocsExamined" : 0,
  "executionStages" : {
    "stage" : "FETCH",
    "nReturned" : 0,
    "executionTimeMillisEstimate" : 0,
    "works" : 1,
    "advanced" : 0,
    "needTime" : 0,
    "needYield" : 0,
    "saveState" : 0,
    "restoreState" : 0,
    "isEOF" : 1,
    "docsExamined" : 0,
    "alreadyHasObj" : 0,
    "inputStage" : {
      "stage" : "IXSCAN",

```

```
Activities Terminal Чт июн 10 01:20:43 en 99%
daniil@RedmiBook-14: ~

    "executionTimeMillisEstimate" : 0,
    "works" : 1,
    "advanced" : 0,
    "needTime" : 0,
    "needYield" : 0,
    "saveState" : 0,
    "restoreState" : 0,
    "isEOF" : 1,
    "keyPattern" : {
      "value" : 1
    },
    "indexName" : "value_1",
    "isMultiKey" : false,
    "multiKeyPaths" : {
      "value" : [ ]
    },
    "isUnique" : false,
    "isSparse" : false,
    "isPartial" : false,
    "indexVersion" : 2,
    "direction" : "forward",
    "indexBounds" : {
      "value" : [
        "[99991.0, inf.0]"
      ]
    },
    "keysExamined" : 0,
    "seeks" : 1,
    "dupsTested" : 0,
    "dupsDropped" : 0
  },
  "serverInfo" : {
    "host" : "RedmiBook-14",
    "port" : 27017,
    "version" : "4.4.6",
    "gitVersion" : "72e66213c2c3eab37d9358d5e78ad7f5c1d0d0d7"
  },
  "ok" : 1
}
```

После создания индекса value значение executionTimeMillis изменилось с 11 до 1. Это значит, что для ускорения работы стоит задавать индексы.

Вывод:

В данной лабораторной работе мы овладели 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.