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

Отчет

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

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

Автор: Кузнецов А.Г.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

Цель работы

Овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

Выполнение

2.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

Практическое задание 2.1.1:

1) Создайте базу данных learn.

2) Заполните коллекцию единорогов unicorns:

```
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b50a6ee5c28b3066b5c3') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b50f6ee5c28b3066b5c4') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b5146ee5c28b3066b5c5') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b5186ee5c28b3066b5c6') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b51e6ee5c28b3066b5c7') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b5506ee5c28b3066b5c8') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b5576ee5c28b3066b5c9') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b55c6ee5c28b3066b5ca') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b56b6ee5c28b3066b5cb') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b5706ee5c28b3066b5cc') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b5756ee5c28b3066b5cd') }
}
```

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

```
learn> document=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
{
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
learn> db.unicorns.insert(document)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583b6446ee5c28b3066b5ce') }
}
learn>
```

4) Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('6583b50a6ee5c28b3066b5c3'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6583b5146ee5c28b3066b5c5'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6583b5186ee5c28b3066b5c6'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6583b51e6ee5c28b3066b5c7'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('6583b5506ee5c28b3066b5c8'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
  },
  {
    _id: ObjectId('6583b5576ee5c28b3066b5c9'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6583b55c6ee5c28b3066b5ca'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6583b56b6ee5c28b3066b5cb'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('6583b5706ee5c28b3066b5cc'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6583b5756ee5c28b3066b5cd'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6583b6446ee5c28b3066b5ce'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
learn>
```

2.2 ВЫБОРКА ДАННЫХ ИЗ БД

Практическое задание 2.2.1:

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

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    _id: ObjectId('6583b6446ee5c28b3066b5ce'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('6583b50a6ee5c28b3066b5c3'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6583b5576ee5c28b3066b5c9'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6583b5706ee5c28b3066b5cc'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6583b55c6ee5c28b3066b5ca'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6583b5186ee5c28b3066b5c6'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6583b5146ee5c28b3066b5c5'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
```

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6583b5506ee5c28b3066b5c8'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6583b56b6ee5c28b3066b5cb'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
    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.

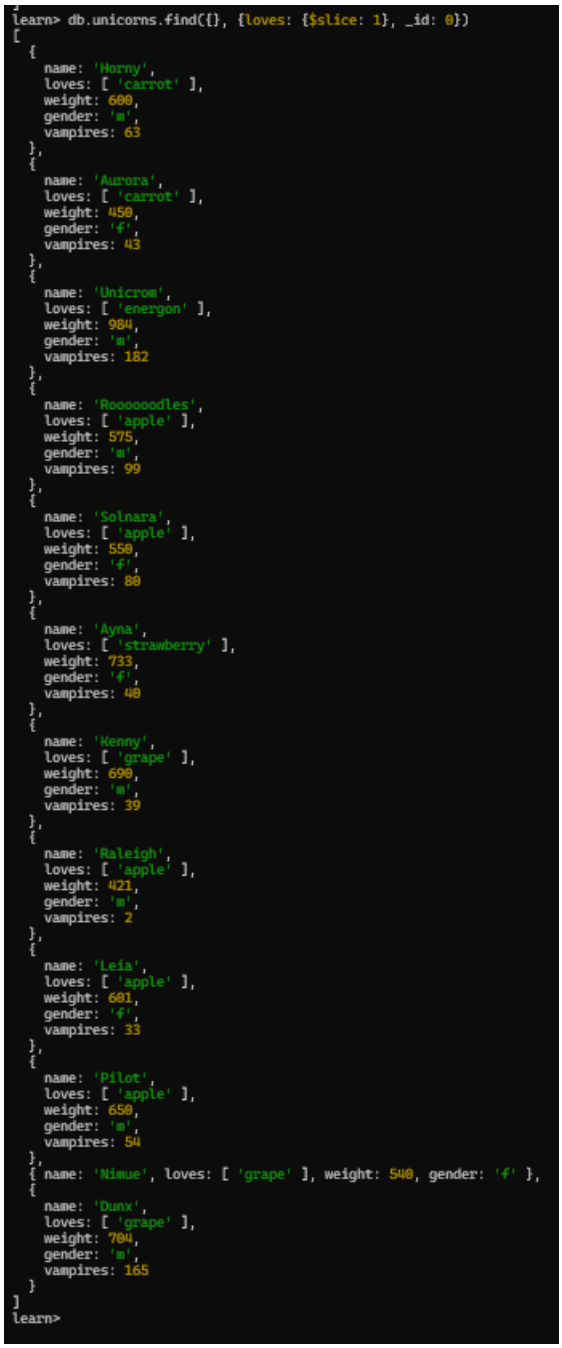
```
learn> db.unicorns.find({gender: "m"}, {loves:0, gender:0}).sort({name: 1})
[
  {
    _id: ObjectId('657f837d2442d8bc060bc4bb'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('657f82842442d8bc060bc4b0'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b6'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b9'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b7'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b3'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b2'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
]
learn>
```

Практическое задание 2.2.3:
Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({ $natural: -1 })
[
  {
    _id: ObjectId('6583b6446ee5c28b3066b5ce'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('6583b5756ee5c28b3066b5cd'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6583b5706ee5c28b3066b5cc'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6583b56b6ee5c28b3066b5cb'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('6583b55c6ee5c28b3066b5ca'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6583b5576ee5c28b3066b5c9'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6583b5506ee5c28b3066b5c8'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6583b51e6ee5c28b3066b5c7'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('6583b5186ee5c28b3066b5c6'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6583b5146ee5c28b3066b5c5'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6583b50a6ee5c28b3066b5c3'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
```

Практическое задание 2.1.4

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



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

2.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 2.3.1

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

```
learn> 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'
  }
]
```

Практическое задание 2.3.2

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

```
learn> db.unicorns.find({gender: 'm', weight: {$gte: 500}, loves: {$all: ['lemon', 'grape']}}, {_id: 0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Практическое задание 2.3.3

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId('6583b5756ee5c28b3066b5cd'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.4

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

```
learn> db.unicorns.find({gender: 'm'}, {name: 1, loves: {$slice: 1}, _id: 0}).sort({name: 1})
[
  { name: 'Dunx', loves: [ 'grape' ] },
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Roooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
```

3.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 3.1.1

1) Создайте коллекцию towns, включающую следующие документы

```
{name: "Punxsutawney ",
populatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: [""],
mayor: {
   name: "Jim Wehrle"
   }}
{name: "New York",
populatiuon: 22200000,
last_sensus: ISODate("2009-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
   name: "Michael Bloomberg",
   party: "I"}}
{name: "Portland",
populatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
   party: "D"}}
```

```
learn> db.towns.insert({name: "Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: [""], mayor: {name: "Jim Wehrle"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583fc2b6ee5c28b3066b5d0') }
}
learn> 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"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583fc7b6ee5c28b3066b5d1') }
}
learn> db.towns.insert({name: "Portland", populatiuon: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: { name: "Sam Adams", party: "D"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6583fd6b6ee5c28b3066b5d2') }
}
learn> db.towns.find()
[
  {
    _id: ObjectId('6583fc2b6ee5c28b3066b5d0'),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('6583fc7b6ee5c28b3066b5d1'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('6583fd6b6ee5c28b3066b5d2'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

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

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

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

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

Практическое задание 3.1.2

4) Сформировать функцию для вывода списка самцов единорогов.

```
learn> function findMaleUnicorns() { return db.unicorns.find({gender: 'm'}); }
[Function: findMaleUnicorns]
```

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

```
learn> var cursor = findMaleUnicorns().sort({name: 1}).limit(2);
```

6) Вывести результат, используя forEach.

```
learn> cursor.forEach(function(unicorn) { print(unicorn.name, unicorn.gender); });
Dunx m
Horny m
```

3.2 АГРЕГИРОВАННЫЕ ЗАПРОСЫ

Практическое задание 3.2.1
Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2
Вывести список предпочтений.

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

Практическое задание 3.2.3
Посчитать количество особей единорогов обоих полов

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

3.3 РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 3.3.1

1. Выполнить команду:

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

```
learn> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
TypeError: db.unicorns.save is not a function
```

```
learn> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedId: ObjectId('658405a16ee5c28b3066b5d3')
}
```

2. Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId('658405a16ee5c28b3066b5d3'),
  name: 'Barny',
  loves: [ 'grape' ],
  weight: 340,
  gender: 'm'
}
```

Практическое задание 3.3.2

1. Для самки единорога Ayna внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.

```
learn> db.unicorns.update({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId('6583b5506ee5c28b3066b5c8'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
```

Практическое задание 3.3.3

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

```
learn> db.unicorns.updateOne({name: "Raleigh", gender: 'm'}, {$set: {loves: ["redbull"]}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId('657f82852442d8bc060bc4b7'),
  name: 'Raleigh',
  loves: 'redbull',
  weight: 421,
  gender: 'm',
  vampires: 2
},
```

Практическое задание 3.3.4

1. Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
learn> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({gender: 'm'})
[
  {
    _id: ObjectId('6583b50a6ee5c28b3066b5c3'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('6583b5146ee5c28b3066b5c5'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('6583b5186ee5c28b3066b5c6'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('6583b5576ee5c28b3066b5c9'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
```

```
  {
    _id: ObjectId('6583b5576ee5c28b3066b5c9'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('6583b55c6ee5c28b3066b5ca'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('6583b5706ee5c28b3066b5cc'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  },
  {
    _id: ObjectId('6583b6446ee5c28b3066b5ce'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170
  },
  {
    _id: ObjectId('658405a16ee5c28b3066b5d3'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 5
  }
]
```

Практическое задание 3.3.5

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

```
learn> db.towns.updateOne({name: "Portland"}, {$set: {"mayor.party": "I"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции towns.

```
learn> db.towns.find()
[
  {
    _id: ObjectId('6583fc2b6ee5c28b3066b5d0'),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('6583fc7b6ee5c28b3066b5d1'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('6583fd6b6ee5c28b3066b5d2'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'I' }
  }
]
```

Практическое задание 3.3.6

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

```
learn> db.unicorns.updateOne({name: "Pilot"}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('6583b5706ee5c28b3066b5cc'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

Практическое задание 3.3.7

1. Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.

```
learn> db.unicorns.updateOne({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemons"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

3.4 УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

Практическое задание 3.4.1

1. Создайте коллекцию towns, включающую следующие документы:

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

2. Удалите документы с беспартийными мэрами.

```
learn> db.towns.deleteMany({"mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
```

3. Проверьте содержание коллекции.

```
learn> db.towns.find()
[
  {
    _id: ObjectId('6583fc7b6ee5c28b3066b5d1'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('6583fd6b6ee5c28b3066b5d2'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'I' }
  }
]
learn>
```

4. Очистите коллекцию.

```
learn> db.towns.deleteMany({})
{ acknowledged: true, deletedCount: 2 }
```

5. Просмотрите список доступных коллекций.

```
learn> show collections
towns
unicorns
```

4.1 ССЫЛКИ В БД

Практическое задание 4.1.1

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

```
learn> db.habitat.insertMany([
... {
... _id:"mf",
... name: "Mysterious forest",
... description: "A creepy and cold mysterious forest"
... },
... {
... _id: "rc",
... name: "Rainbow clouds",
... description: "There are clouds here and everything is rainbow"
... }
... ]);
{ acknowledged: true, insertedIds: { '0': 'mf', '1': 'rc' } }
```

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

```
learn> db.unicorns.updateMany(
... {name: {$regex: /^A/i}},
... {$set: {habitat: {$ref: "habitat", $id: "mf"}}}
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

```
learn> db.unicorns.updateMany(
... {name: {$in: ["Barny", "Pilot"]}},
... {$set: {habitat: {$ref: "habitat", $id: "rc"}}}
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

3. Проверьте содержание коллекции едиорогов.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('6583b50a6ee5c28b3066b5c3'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('6583b50f6ee5c28b3066b5c4'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat: DBRef('habitat', 'mf')
  },
  {
    _id: ObjectId('6583b5146ee5c28b3066b5c5'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('6583b5186ee5c28b3066b5c6'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('6583b51e6ee5c28b3066b5c7'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('6583b5506ee5c28b3066b5c8'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51,
    habitat: DBRef('habitat', 'mf')
  },
  {
    _id: ObjectId('6583b5576ee5c28b3066b5c9'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('6583b55c6ee5c28b3066b5ca'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('6583b56b6ee5c28b3066b5cb'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('6583b5706ee5c28b3066b5cc'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59,
    habitat: DBRef('habitat', 'rc')
  },
  {
    _id: ObjectId('6583b5756ee5c28b3066b5cd'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6583b6446ee5c28b3066b5ce'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170
  },
  {
    _id: ObjectId('658405a16ee5c28b3066b5d3'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 5,
    habitat: DBRef('habitat', 'rc')
  }
```

4.2 НАСТРОЙКА ИНДЕКСОВ

Практическое задание 4.2.1

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

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

4.3 УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 4.3.1

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

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

2. *Удалите все индексы, кроме индекса для идентификатора.*

```
learn> db.unicorns.dropIndex("name_1")
{ nIndexesWas: 2, ok: 1 }
```

3. *Попытайтесь удалить индекс для идентификатора.*

```
learn> db.unicorns.dropIndex("_id_")
MongoServerError: cannot drop _id index
```

4.4 ПЛАН ЗАПРОСА

Практическое задание 4.4.1

1. Создайте объемную коллекцию numbers, задействовав курсор:

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

```
learn> for (let i = 0; i < 100000; i++) {
... db.numbers.insert({value: i});
... }
;
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65841fe66ee5c28b30683c73') }
}
learn> ;
```

2. Выберите последних четыре документа.

```
learn> db.numbers.find().sort({ _id: -1 }).limit(4)
[
  { _id: ObjectId('65841fe66ee5c28b30683c73'), value: 99999 },
  { _id: ObjectId('65841fe66ee5c28b30683c72'), value: 99998 },
  { _id: ObjectId('65841fe66ee5c28b30683c71'), value: 99997 },
  { _id: ObjectId('65841fe66ee5c28b30683c70'), value: 99996 }
]
```

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

```
learn> db.numbers.find().sort({ _id: -1 }).limit(4).explain("executionStats")
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 13,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'limit',
    planNodeId: 3,
    nReturned: 4,
    executionTimeMillisEstimate: 0,
    opens: 1,
    closes: 1,
    saveState: 0,
    restoreState: 0,
    isEOF: 1,
    limit: 4,
```

4. Создайте индекс для ключа value.

```
learn> db.numbers.createIndex({ value: 1 });
value_1
```

5. Получите информацию о всех индексах коллекции numbres.

```
learn> db.numbers.getIndexes();
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { value: 1 }, name: 'value_1' }
]
```

6. Выполните запрос 2.

```
learn> db.numbers.find().sort({ _id: -1 }).limit(4)
[
  { _id: ObjectId('65841fe66ee5c28b30683c73'), value: 99999 },
  { _id: ObjectId('65841fe66ee5c28b30683c72'), value: 99998 },
  { _id: ObjectId('65841fe66ee5c28b30683c71'), value: 99997 },
  { _id: ObjectId('65841fe66ee5c28b30683c70'), value: 99996 }
]
```

7. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

```
learn> db.numbers.find().sort({ value: -1 }).limit(4).explain("executionStats");
{
 executionStats: {
   executionSuccess: true,
   nReturned: 4,
   executionTimeMillis: 7,
   totalKeysExamined: 4,
   totalDocsExamined: 4,
   executionStages: {
     stage: 'limit',
     planNodeId: 3,
     nReturned: 4,
     executionTimeMillisEstimate: 0,
     opens: 1,
     closes: 1,
     saveState: 0,
     restoreState: 0,
     isEOF: 1,
     limit: 4,
```

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

Время выполнения без индекса: 13 мс

Время выполнения с индексом: 9 мс

Время выполнения с индексом на ~70% быстрее, чем время выполнения без индекса.

Вывод

В ходе лабораторной работы удалось 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.