Университет ИТМО

Лабораторная работа № 6

«Работа с БД в СУБД MongoDB»

Выполнила: Анисимова Ксения Сергеевна
Группа: K3241
Преподаватель: Говорова Марина Михайловна

Санкт – Петербург
2024

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

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4+, 6.0.6 (текущая).

Практическое задание: Практические задания 2.1.1 – 4.4.1

Ход работы:

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});
elon'], weight: 650, gender: 'm', vampires: 54});
db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d3c') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d3d') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d3e') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d3f') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d40') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d41') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d42') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d43') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d44') }
}
```

```
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268a65d912a7acae9d45') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8268b65d912a7acae9d46') }
}
```

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('65a8298465d912a7acae9d48') }
}
```

4. Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d3c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3e'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d40'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
{
  _id: ObjectId('65a8268a65d912a7acae9d41'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
{
  _id: ObjectId('65a8268a65d912a7acae9d42'),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 39
},
{
  _id: ObjectId('65a8268a65d912a7acae9d43'),
  name: 'Raleigh',
  loves: [ 'apple', 'sugar' ],
  weight: 421,
  gender: 'm',
  vampires: 2
},
{
  _id: ObjectId('65a8268a65d912a7acae9d44'),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33
},
{
  _id: ObjectId('65a8268a65d912a7acae9d45'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon' ],
  weight: 650,
  gender: 'm',
  vampires: 54
},
{
  _id: ObjectId('65a8268b65d912a7acae9d46'),
  name: 'Nimue',
  loves: [ 'grape', 'carrot' ],
  weight: 540,
  gender: 'f'
},
{
  _id: ObjectId('65a8298465d912a7acae9d48'),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
```

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.

Для самцов:

```
learn> db.unicorns.find({gender:'m'}).limit(3).sort({name:1})
[
  {
    _id: ObjectId('65a8298465d912a7acae9d48'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d42'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Самки:

```
learn> db.unicorns.find({gender:'f'}).limit(3).sort({name:1})
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d3d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d41'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d44'),
    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.find({gender:'f'},{_id:0, name:1, loves: 'carrot'})
[
  { name: 'Aurora', loves: 'carrot' },
  { name: 'Solnara', loves: 'carrot' },
  { name: 'Ayna', loves: 'carrot' },
  { name: 'Leia', loves: 'carrot' },
  { name: 'Nimue', loves: 'carrot' }
]
learn> db.unicorns.find({gender:'f'},{_id:0, name:1, loves: 'carrot'}).limit(1)
[ { name: 'Aurora', loves: 'carrot' } ]
```

Практическое задание 2.2.2:

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

```
learn> db.unicorns.find({gender:'m'},{loves:0, gender:0}).sort({name:1})
[
  {
    _id: ObjectId('65a8298465d912a7acae9d48'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3c'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d42'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d45'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d43'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3f'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3e'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
]
```

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({ $natural: -1 })
[
  {
    _id: ObjectId('65a8298465d912a7acae9d48'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65a8268b65d912a7acae9d46'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d45'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d44'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d43'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d42'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d41'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d40'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3e'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
```

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
   }
 ]
```

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 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.

```
learn> db.unicorns.find({vampires:{$exists:false}})
[
  {
    _id: ObjectId('65a8268b65d912a7acae9d46'),
    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'},{loves:{$slice:1}}
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d3c'),
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3e'),
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d42'),
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d43'),
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d45'),
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65a8298465d912a7acae9d48'),
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

Практическое задание 3.1.1:

Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.insertMany([{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"}}
... ]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65a8357b65d912a7acae9d4b'),
    '1': ObjectId('65a8357b65d912a7acae9d4c'),
    '2': ObjectId('65a8357b65d912a7acae9d4d')
  }
}
```

1. 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: 1})
[
  {
    _id: ObjectId('65a8357b65d912a7acae9d4c'),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

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

```
learn> db.towns.find({'mayor.party' : {$exists : 1}}, {name: 1, mayor: 1, _id: false})
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  { name: 'Portland', mayor: { name: 'Sam Adams', party: 'D' } }
]
```

Практическое задание 3.1.2:

1. Сформировать функцию для вывода списка самцов единорогов.

```
[learn> function unicornsMaleList() {return db.unicorns.find({gender: 'm'})}
 [Function: unicornsMaleList]
[learn> unicornsMaleList()
 [
   {
     _id: ObjectId('65a8268a65d912a7acae9d3c'),
     name: 'Horny',
     loves: [ 'carrot', 'papaya' ],
     weight: 600,
     gender: 'm',
     vampires: 63
   },
   {
     _id: ObjectId('65a8268a65d912a7acae9d3e'),
     name: 'Unicrom',
     loves: [ 'energon', 'redbull' ],
     weight: 984,
     gender: 'm',
     vampires: 182
   },
   {
     _id: ObjectId('65a8268a65d912a7acae9d3f'),
     name: 'Roooooodles',
     loves: [ 'apple' ],
     weight: 575,
     gender: 'm',
     vampires: 99
   },
   {
     _id: ObjectId('65a8268a65d912a7acae9d42'),
     name: 'Kenny',
     loves: [ 'grape', 'lemon' ],
     weight: 690,
     gender: 'm',
     vampires: 39
   },
   {
     _id: ObjectId('65a8268a65d912a7acae9d43'),
     name: 'Raleigh',
     loves: [ 'apple', 'sugar' ],
     weight: 421,
     gender: 'm',
     vampires: 2
   },
   {
     _id: ObjectId('65a8268a65d912a7acae9d45'),
     name: 'Pilot',
     loves: [ 'apple', 'watermelon' ],
     weight: 650,
     gender: 'm',
     vampires: 54
   },
   {
     _id: ObjectId('65a8298465d912a7acae9d48'),
     name: 'Dunx',
     loves: [ 'grape', 'watermelon' ],
     weight: 704,
     gender: 'm',
     vampires: 165
   }
 ]
```

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

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

3. Вывести результат, используя forEach.

```
[learn> cursor.forEach(function(obj){print(obj.name)})
 Dunx
 Horny
```

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: 'm', count: 7 }, { _id: 'f', count: 5 } ]
```

Практическое задание 3.3.1:

1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции* `unicorns`.

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

Команда save устарела.

Практическое задание 3.3.2:

Для самки единорога `Ayna` *внести изменения в БД: теперь ее вес 800, она убила 51 вапмира. Проверить содержимое коллекции* `unicorns`.

```
learn> db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight:800, vampires: 51}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Ayna'})
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d41'),
    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.* 2. *Проверить содержимое коллекции* `unicorns`.

```
learn> db.unicorns.updateOne({name: 'Raleigh'}, {$set: {'loves': ['redbull']}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Raleigh'})
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d43'),
    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.*
2. *Проверить содержимое коллекции* `unicorns`.

```
learn> db.unicorns.find({gender:'m'}, {_id:0, name:1, vampires:1})
[
  { name: 'Horny', vampires: 68 },
  { name: 'Unicrom', vampires: 187 },
  { name: 'Roooooodles', vampires: 104 },
  { name: 'Kenny', vampires: 44 },
  { name: 'Raleigh', vampires: 7 },
  { name: 'Pilot', vampires: 59 },
  { name: 'Dunx', vampires: 170 }
]
```

Практическое задание 3.3.5:

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

```
learn> db.towns.updateOne({name:'Portland'},{$unset:{'mayor.party':1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name:'Portland'})
[
  {
    _id: ObjectId('65a8357b65d912a7acae9d4d'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

Практическое задание 3.3.6:

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

```
learn> db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Pilot'})
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d45'),
    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.*
2. *Проверить содержимое коллекции* `unicorns`.

```
learn> db.unicorns.updateOne({name: 'Aurora'}, {$addToSet: {loves: {$each: ['sugar', 'le
mon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Aurora'})
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d3d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

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"}}
learn> db.towns.insertMany([{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"}}]
... );
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65a83f3c65d912a7acae9d4e'),
    '1': ObjectId('65a83f3c65d912a7acae9d4f'),
    '2': ObjectId('65a83f3c65d912a7acae9d50')
  }
}
```

2. Удалите документы с беспартийными мэрами.

```
learn> db.towns.deleteMany({'mayor.party':{$exists: 0}})
{ acknowledged: true, deletedCount: 3 }
```

3. Проверьте содержание коллекции.

```
learn> db.towns.find()
[
  {
    _id: ObjectId('65a8357b65d912a7acae9d4c'),
    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('65a83f3c65d912a7acae9d50'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

4. Очистите коллекцию.

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

5. Просмотрите список доступных коллекций.

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

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.createCollection('area')
{ ok: 1 }
learn> db.area.insertMany([{_id:'forest', name: 'The forest', description: 'Area with lu
shious trees and a rich wildlife dependant on them.'}, {_id: 'mountain', name: 'The moun
tain', description: 'A tall mountain on the west coast of some Japaneese island near Mal
ayia'}, {_id:'meadow', name: 'The Green meadow', description:'A wide area filled fith gr
een herbs and an abundance of colourful flowers.'}]);
{
  acknowledged: true,
  insertedIds: { '0': 'forest', '1': 'mountain', '2': 'meadow' }
}
```

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

```
learn> db.unicorns.updateOne({_id: ObjectId('65a8268a65d912a7acae9d3d')}, {$set: {locat
ion:{$ref :'area', $id:'mountain'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({_id: ObjectId('65a8268a65d912a7acae9d45')},{$set: {locati
n:{$ref :'area', $id:'meadow'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('65a8268a65d912a7acae9d3c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    location: DBRef('area', 'mountain')
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3e'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d3f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d40'),
    name: 'Solnara',
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d43'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d44'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('65a8268a65d912a7acae9d45'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59,
    location: DBRef('area', 'meadow')
  },
  {
    _id: ObjectId('65a8268b65d912a7acae9d46'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('65a8298465d912a7acae9d48'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170
  }
]
```

Практическое задание 4.2.1:

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

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

Практическое задание 4.3.1:

1. Получите информацию обо 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.dropIndexes()
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
```

3. Попытайтесь удалить индекс для идентификатора.

```
learn> db.unicorns.dropIndexes('_id_')
MongoServerError: cannot drop _id index
```

Практическое задание 4.4.1:

1. Создайте объемную коллекцию `numbers`*, задействовав курсор:* `for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}`

2. Сколько потребовалось времени на выполнение запроса? (по значению параметра `executionTimeMillis`*)*

3. Проанализируйте план выполнения запроса

4. Создайте индекс для ключа `value`*.*

5. Получите информацию о всех индексах коллекции `numbres`*.*

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?

```
learn> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bu
rite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65a8491583fdc40527f8014e') }
}
learn> db.numbers.find().sort({value:-1}).limit(4)
[
  { _id: ObjectId('65a8491583fdc40527f8014e'), value: 99999 },
  { _id: ObjectId('65a8479965d912a7acb023f0'), value: 99999 },
  { _id: ObjectId('65a8479965d912a7acb023ef'), value: 99998 },
  { _id: ObjectId('65a8491583fdc40527f8014d'), value: 99998 }
]
```

Без индекса:

```
learn> db.numbers.find().sort({value:-1}).limit(4).explain('executionStats').executionStat
s.executionTimeMillis
135
```

С индексом:

```
learn> db.numbers.createIndex({value:-1})
value_-1
learn> db.numbers.find().sort({value:-1}).limit(4).explain('executionStats').executionStat
s.executionTimeMillis
2
```

Можно сделать вывод, что с индексом быстрее. Т.к. индекс позволяет, не просматривая каждый документ, эффективно находить и сортировать данные.

Вывод:

В ходе выполнения данной лабораторной работы мы получили практические навыки работы с CRUD-операциями в MongoDB. Мы изучили основные команды для выполнения этих операций в коллекциях.