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: Пузенко А.А.

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:

Практическое задание 2.1.1:

1. Создайте базу данных learn.
2. Заполните коллекцию единорогов unicorns:

```
27017> use learn
switched to db learn
learn> db.createCollection('unicorns')
{ ok: 1 }
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594a6c0d602a4f3f2573c') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594c3c0d602a4f3f2573d') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594c9c0d602a4f3f2573e') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594d0c0d602a4f3f2573f') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594d6c0d602a4f3f25740') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594e1c0d602a4f3f25741') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594e6c0d602a4f3f25742') }
}
```

```
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594eac0d602a4f3f25743') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594efc0d602a4f3f25744') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594f5c0d602a4f3f25745') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65e594fbc0d602a4f3f25746') }
}
```

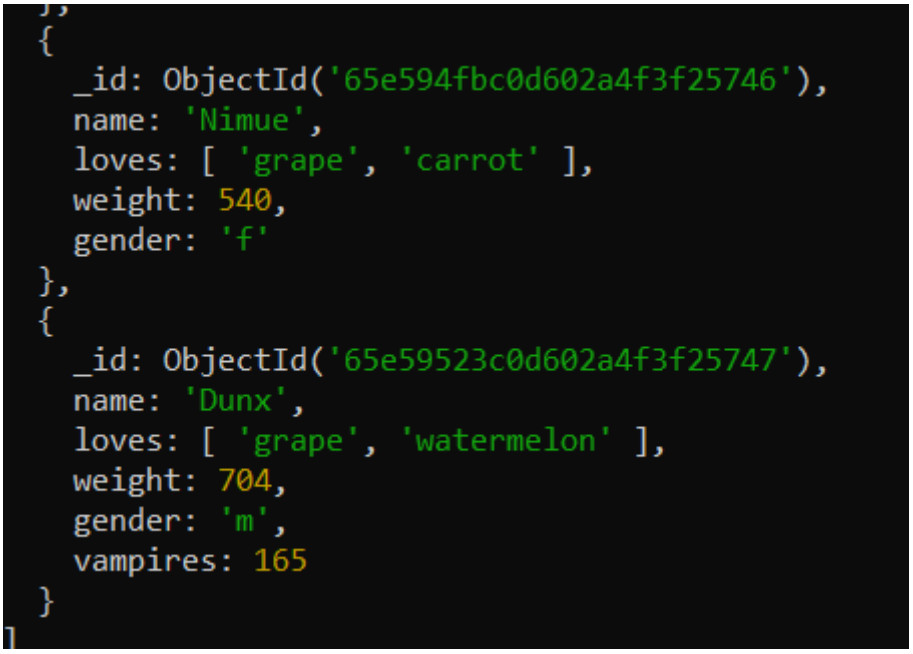
3. Используя второй способ, вставьте в коллекцию единорогов документ:

```
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('65e59523c0d602a4f3f25747') }
}
```

4. Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('65e594a6c0d602a4f3f2573c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65e594c3c0d602a4f3f2573d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65e594c9c0d602a4f3f2573e'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('65e594d0c0d602a4f3f2573f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65e594d6c0d602a4f3f25740'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
```

```
{
  _id: ObjectId('65e594e1c0d602a4f3f25741'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
{
  _id: ObjectId('65e594e6c0d602a4f3f25742'),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 39
},
{
  _id: ObjectId('65e594eac0d602a4f3f25743'),
  name: 'Raleigh',
  loves: [ 'apple', 'sugar' ],
  weight: 421,
  gender: 'm',
  vampires: 2
},
{
  _id: ObjectId('65e594efc0d602a4f3f25744'),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33
},
{
  _id: ObjectId('65e594f5c0d602a4f3f25745'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon' ],
  weight: 650,
  gender: 'm',
  vampires: 54
},
```

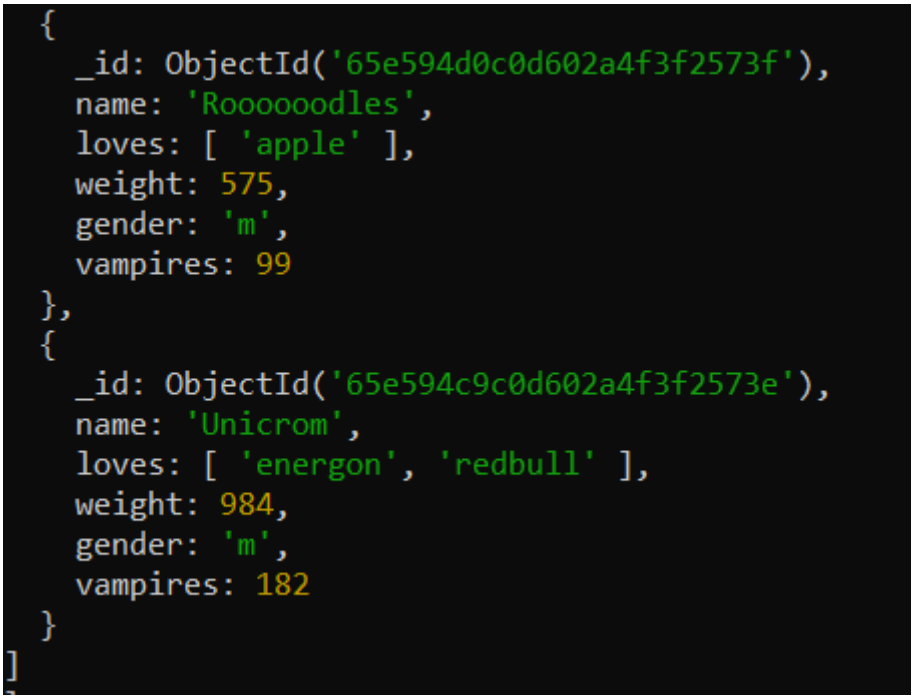


Практическое задание 2.2.1:

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

Самцы

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    _id: ObjectId('65e59523c0d602a4f3f25747'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65e594a6c0d602a4f3f2573c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65e594e6c0d602a4f3f25742'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65e594f5c0d602a4f3f25745'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
learn>
  },
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
```


{
_id: ObjectId('65e594d0c0d602a4f3f2573f'),
name: 'Roooooodles',
loves: ['apple'],
weight: 575,
gender: 'm',
vampires: 99
},
{
_id: ObjectId('65e594c9c0d602a4f3f2573e'),
name: 'Unicrom',
loves: ['energon', 'redbull'],
weight: 984,
gender: 'm',
vampires: 182
}
]

Самки

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1})
[
  {
    _id: ObjectId('65e594c3c0d602a4f3f2573d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65e594e1c0d602a4f3f25741'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65e594efc0d602a4f3f25744'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('65e594fbc0d602a4f3f25746'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('65e594d6c0d602a4f3f25740'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
]
```

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

findOne

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId('65e594c3c0d602a4f3f2573d'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

limit

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId('65e594c3c0d602a4f3f2573d'),
    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('65e59523c0d602a4f3f25747'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('65e594a6c0d602a4f3f2573c'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('65e594e6c0d602a4f3f25742'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('65e594f5c0d602a4f3f25745'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('65e594d0c0d602a4f3f2573f'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('65e594c9c0d602a4f3f2573e'),
    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('65e59523c0d602a4f3f25747'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65e594fbc0d602a4f3f25746'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('65e594f5c0d602a4f3f25745'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65e594efc0d602a4f3f25744'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
```

```
{
  _id: ObjectId('65e594e6c0d602a4f3f25742'),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 39
},
{
  _id: ObjectId('65e594e1c0d602a4f3f25741'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
{
  _id: ObjectId('65e594d6c0d602a4f3f25740'),
  name: 'Solnara',
  loves: [ 'apple', 'carrot', 'chocolate' ],
  weight: 550,
  gender: 'f',
  vampires: 80
},
{
  _id: ObjectId('65e594d0c0d602a4f3f2573f'),
  name: 'Roooooodles',
  loves: [ 'apple' ],
  weight: 575,
  gender: 'm',
  vampires: 99
},
{
  _id: ObjectId('65e594c9c0d602a4f3f2573e'),
  name: 'Unicrom',
  loves: [ 'energon', 'redbull' ],
  weight: 984,
  gender: 'm',
  vampires: 182
},
```

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 2.3.2:

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}})

learn>
```

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}}).sort({name: 1})
[
  {
    _id: ObjectId('65e59523c0d602a4f3f25747'),
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65e594a6c0d602a4f3f2573c'),
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65e594e6c0d602a4f3f25742'),
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65e594f5c0d602a4f3f25745'),
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
```

Практическое задание 3.1.1:

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('65e59dad0a118997386767ce'),
    '1': ObjectId('65e59dad0a118997386767cf'),
    '2': ObjectId('65e59dad0a118997386767d0')
  }
}
```

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' } } ]
```

Практическое задание 3.1.2:

1. Сформировать функцию для вывода списка самцов единорогов.

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

3. Вывести результат, используя forEach.

```
learn> function print_male_uni() {var cursor = db.unicorns.find({gender: 'm'}); cursor.sort({name: 1}).limit(2); cursor.forEach(function(unicorn) {print(unicorn.name);});}
[Function: print_male_uni]
learn> print_male_uni
[Function: print_male_uni]
learn> print_male_uni()
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.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedId: ObjectId('65e5a26b0a118997386767d1')
}
learn> db.unicorns.find({name: 'Barny'})
[
  {
    _id: ObjectId('65e5a26b0a118997386767d1'),
    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 вампира.*
2. *Проверить содержимое коллекции* `unicorns`.

```
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId('65e594e1c0d602a4f3f25741'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
]
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
}
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId('65e594e1c0d602a4f3f25741'),
    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.find({name: 'Raleigh'})
[
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

```
learn> db.unicorns.update({name: 'Raleigh'}, {$set: {loves: ['redbull']}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Raleigh'})
[
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

Практическое задание 3.3.4:

1. Всем самцам единорогов увеличить количество убитых вампиров 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('65e594a6c0d602a4f3f2573c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('65e594c9c0d602a4f3f2573e'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('65e594d0c0d602a4f3f2573f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('65e594e6c0d602a4f3f25742'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('65e594eac0d602a4f3f25743'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
```

```
{
  _id: ObjectId('65e594f5c0d602a4f3f25745'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon' ],
  weight: 650,
  gender: 'm',
  vampires: 59
},
{
  _id: ObjectId('65e59523c0d602a4f3f25747'),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 170
},
{
  _id: ObjectId('65e5a26b0a118997386767d1'),
  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.*
2. *Проверить содержимое коллекции* `towns`.

```
learn> db.towns.updateMany({name: 'Portland'}, {$set: {'mayor.party': null}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: 'Portland'})
[
  {
    _id: ObjectId('65e59dad0a118997386767d0'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: null }
  }
]
```

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.find({name: 'Pilot'})
[
  {
    _id: ObjectId('65e594f5c0d602a4f3f25745'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn> db.unicorns.update({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId('65e594f5c0d602a4f3f25745'),
    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.find({name: 'Aurora'})
[
  {
    _id: ObjectId('65e594c3c0d602a4f3f2573d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

```
learn> db.unicorns.updateOne({name: 'Aurora'}, {$addToSet: {loves: {$each: ['sugar', 'lemons']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId('65e594c3c0d602a4f3f2573d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 3.4.1:

4. Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.find()
[
  {
    _id: ObjectId('65ee59493cfc3e0d34b783cd'),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('65ee59493cfc3e0d34b783ce'),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('65ee59493cfc3e0d34b783cf'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

5. Удалите документы с беспартийными мэрами.

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

6. Проверьте содержание коллекции.

```
learn> db.towns.find()
[
  {
    _id: ObjectId('65ee59493cfc3e0d34b783ce'),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('65ee59493cfc3e0d34b783cf'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

7. Очистите коллекцию.

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

8. Просмотрите список доступных коллекций

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

Практическое задание 4.1.1:

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

```
learn> db.createCollection('areas')
{ ok: 1 }
```

```
learn> db.areas.insertMany([{_id: 'RU', name: 'Russian Federation', description: 'Country spanning Eastern Europe and North Asia'}, {_id: 'FR', name: 'France', description: 'country located primarily in Western Europe'}, {_id: 'JP', name: 'Japan', description: 'island country in East Asia'}])
{
  acknowledged: true,
  insertedIds: { '0': 'RU', '1': 'FR', '2': 'JP' }
}
learn> db.areas.find()
[
  {
    _id: 'RU',
    name: 'Russian Federation',
    description: 'Country spanning Eastern Europe and North Asia'
  },
  {
    _id: 'FR',
    name: 'France',
    description: 'country located primarily in Western Europe'
  },
  {
    _id: 'JP',
    name: 'Japan',
    description: 'island country in East Asia'
  }
]
```

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

```
learn> db.unicorns.update({name: 'Horny'}, {$set: {area: {$ref: 'areas', $id: 'RU'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Unicrom'}, {$set: {area: {$ref: 'areas', $id: 'FR'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
learn> db.unicorns.update({name: 'Solnara'}, {$set: {area: {$ref: 'areas', $id: 'JP'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

9. Проверьте содержание коллекции едиорогов.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('65e594a6c0d602a4f3f2573c'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    area: DBRef('areas', 'RU')
  },
  {
    _id: ObjectId('65e594c3c0d602a4f3f2573d'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65e594c9c0d602a4f3f2573e'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187,
    area: DBRef('areas', 'FR')
  },
  {
    _id: ObjectId('65e594d0c0d602a4f3f2573f'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
```

```
  {
    _id: ObjectId('65e594d6c0d602a4f3f25740'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80,
    area: DBRef('areas', 'JP')
  },
```

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' ]
```

Практическое задание 4.3.1:

11. 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 }
]
```

12. Удалите все индексы, кроме индекса для идентификатора.

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

13. Попытайтесь удалить индекс для идентификатора.

```
learn> db.unicorns.dropIndex('_id_')
MongoServerError[InvalidOptions]: 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})}`

```
learn> db.createCollection('numbers')
{ ok: 1 }
```

```
learn> for (i=0; i < 100000; i++) {db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65ee635f3cfc3e0d34ba9eed') }
}
```

2. *Выберите последних четыре документа.*

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eed'), value: 99999 },
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eec'), value: 99998 },
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eeb'), value: 99997 },
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eea'), value: 99996 }
]
```

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

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 169,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
```

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

```
learn> db.numbers.ensureIndex({'value': 1}, {'unique': true})
[ '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', unique: true }
]
```

6. *Выполните запрос 2.*

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eed'), value: 99999 },
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eec'), value: 99998 },
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eeb'), value: 99997 },
  { _id: ObjectId('65ee635f3cfc3e0d34ba9eea'), 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: 3,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
  executionStages: {
```

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

Без использование индекса executionTimeMillis было равно 169, с индексом 3. Второй запрос более эффективен

Вывод:

В ходе лабораторной работы, я освоил навыки работы с СУБД MongoDB. В ходе практических заданий применены функции на выборку данных, а также были созданы запросы для удаления, замены и вставки данных разными способами.