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: Гусейнова М. Э.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

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

Оборудование: компьютерный класс.

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

Выполнение

Практическое задание 2.1.1:

1) Создайте базу данных learn.

2) Заполните коллекцию единорогов unicorns:

```
test> 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('65b2445dc3fabc87a482bfcd') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
elon'], weight: 650, gender: 'm', vampires: 54});
db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfce') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfcf') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd0') }
[}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
[{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd1') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd2') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd3') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd4') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd5') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd6') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b2445dc3fabc87a482bfd7') }
}
```

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('65b244bdc3fabc87a482bfd8') }
}
```

4) Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfce'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcf'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd1'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd2'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
```

```
{
  _id: ObjectId('65b2445dc3fabc87a482bfd4'),
  name: 'Raleigh',
  loves: [ 'apple', 'sugar' ],
  weight: 421,
  gender: 'm',
  vampires: 2
},
{
  _id: ObjectId('65b2445dc3fabc87a482bfd5'),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33
},
{
  _id: ObjectId('65b2445dc3fabc87a482bfd6'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon' ],
  weight: 650,
  gender: 'm',
  vampires: 54
},
{
  _id: ObjectId('65b2445dc3fabc87a482bfd7'),
  name: 'Nimue',
  loves: [ 'grape', 'carrot' ],
  weight: 540,
  gender: 'f'
},
{
  _id: ObjectId('65b244bdc3fabc87a482bfd8'),
  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.

Список самцов:

```
learn> db.unicorns.find({gender: "m"}).sort({name: 1})
[
  {
    _id: ObjectId('65b244bdc3fabc87a482bfd8'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcf'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
learn>
```

Список самок:

```
learn> db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfce'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd2'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd5'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

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('65b2445dc3fabc87a482bfce'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn>
```

Limit:

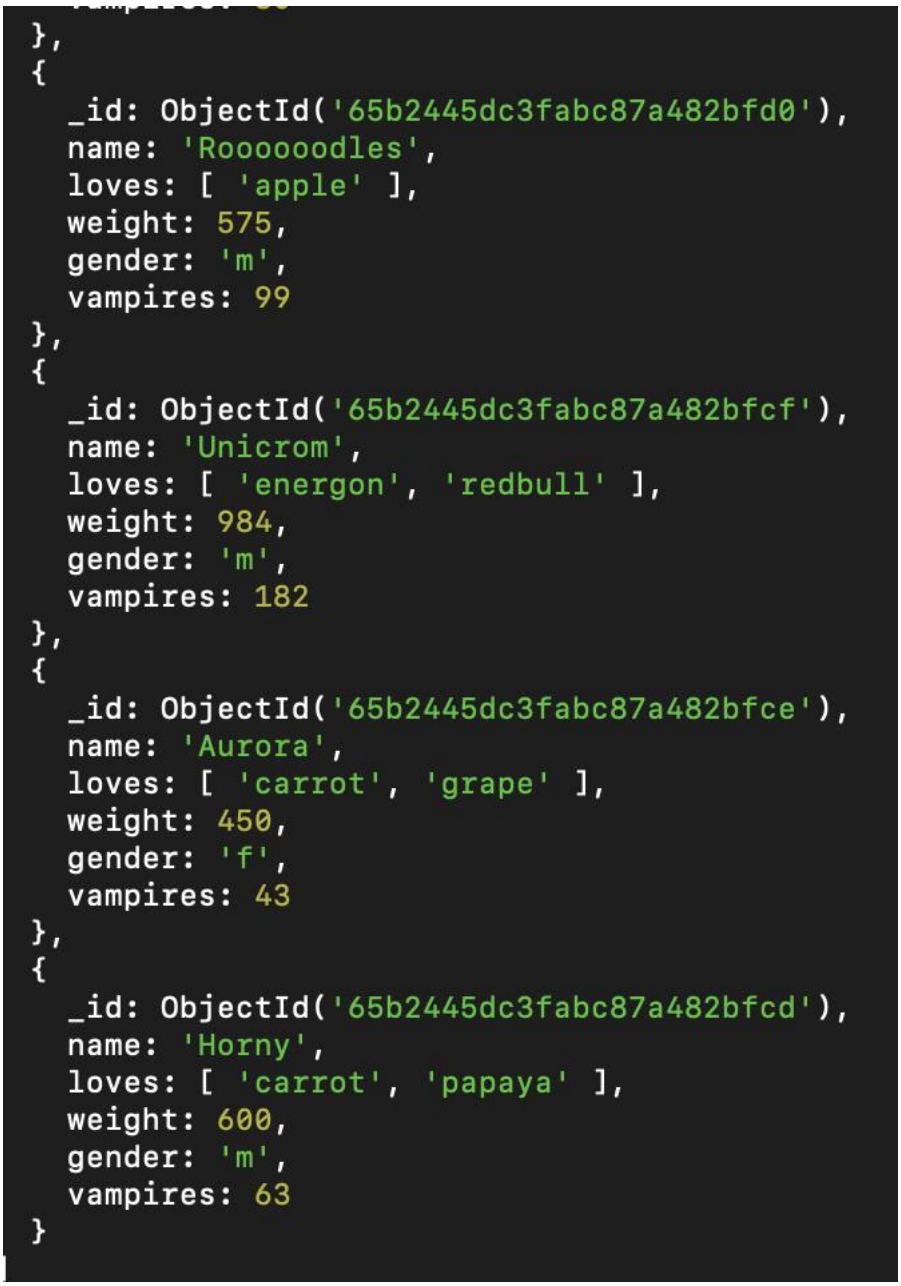
Практическое задание 2.2.2:

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('65b244bdc3fabc87a482bfd8'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcd'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd0'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcf'),
    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('65b244bdc3fabc87a482bfd8'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd7'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd5'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd2'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd1'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
```



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,
```

```
  },
  {
    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>
```

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'
  }
]
learn>
```

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: ['lemom', 'grape']}}, {_id: 0})

learn>
```

Практическое задание 2.3.3

```
learn> db.unicorns.find({vampires: {$exists:false}})
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd7'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
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('65b244bdc3fabc87a482bfd8'),
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcd'),
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcf'),
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
learn>
```

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.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('65b25571c3fabc87a482bfd9'),
    '1': ObjectId('65b25571c3fabc87a482bfda'),
    '2': ObjectId('65b25571c3fabc87a482bfdb')
  }
}
learn>
```

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

3) Сформировать функцию для вывода списка самцов единорогов.

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

5) Вывести результат, используя forEach.

```
learn> function MaleUnicorns()
... {var cursor = db.unicorns.find({gender: "m"}); null;
... cursor.sort({name: 1}).limit(2);null;
... cursor.forEach(function(unicorn){
... print(unicorn.name);
... });}
[Function: MaleUnicorns]
learn> MaleUnicorns()
Dunx
Horny

learn>
```

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
learn> db.unicorns.find({gender:'m', weight: {$gte: 500, $lte: 600}}).count()
(node:27289) [MONGODB DRIVER] Warning: cursor.count is deprecated and will be removed in the next major versio
n, please use `collection.estimatedDocumentCount` or `collection.countDocuments` instead
2
learn>
```

Практическое задание 3.2.2

Вывести список предпочтений.

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

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 } ]
learn>
```

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

2) Проверить содержимое коллекции unicorns.

Практическое задание 3.3.2

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
}
learn>
```

```
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd2'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  },
```

Практическое задание 3.3.3

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

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

```
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: 'redbull',
    weight: 421,
    gender: 'm',
    vampires: 2
  },
```

Практическое задание 3.3.4

Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
learn> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi:true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 7,
  modifiedCount: 7,
  upsertedCount: 0
}
learn>
```

```
learn> db.unicorns.find({gender: "m"})
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcf'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: 'redbull',
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  },
  {
    _id: ObjectId('65b244bdc3fabc87a482bfd8'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170
  }
]
learn>
```

Практическое задание 3.3.5

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

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

```
learn> db.town.find({name:"Portland"})

learn>
```

Практическое задание 3.3.6

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

```
learn> db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn>
```

Практическое задание 3.3.7

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

```
learn> db.unicorns.update({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfce'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

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.

1. *Проверьте содержание коллекции.*
2. *Очистите коллекцию.*
3. *Просмотрите список доступных коллекций.*

```
learn> db.towns.remove({"mayor.party": "I"})
DeprecationWarning: Collection.remove() is depre
te.
{ acknowledged: true, deletedCount: 1 }
learn>
```

```
learn> db.towns.find()
[
  {
    _id: ObjectId('65b25571c3fabc87a482bfd9'),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('65b25571c3fabc87a482bfdb'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  },
  {
```

```
learn> db.towns.remove({})
{ acknowledged: true, deletedCount: 3 }
learn> show collections
towns
unicorns
learn>
```

Практическое задание 4.1.1

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

2) Включите для нескольких единорогов в документы ссылку на зону обитания, использую второй способ автоматического связывания.
3) Проверьте содержание коллекции едиорогов.

```
learn> db.habitat.insertMany([ {_id: "hv", name: "Heavens", description: "Heavens"}, {_id: "mc", name: "Magic
cave", description: "Magic cave"}, {_id: "wk", name: "Water Kingdom", description: "Water kingdom"}
... ])
{
  acknowledged: true,
  insertedIds: { '0': 'hv', '1': 'mc', '2': 'wk' }
}
learn>
```

```
learn> db.unicorns.update({gender: "m"}, {$set: {habitat:{$ref:"habitat", $id:"wk"}}}, {multi:true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 7,
  modifiedCount: 6,
  upsertedCount: 0
}
learn>
```

```
learn> db.unicorns.find({gender: "m"})
[
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habitat: DBRef('habitat', 'wk')
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfcf'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187,
    habitat: DBRef('habitat', 'wk')
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104,
    habitat: DBRef('habitat', 'wk')
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44,
    habitat: DBRef('habitat', 'wk')
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd4'),
    name: 'Raleigh',
    loves: 'redbull',
    weight: 421,
    gender: 'm',
    vampires: 7,
    habitat: DBRef('habitat', 'wk')
  },
  {
    _id: ObjectId('65b2445dc3fabc87a482bfd6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59,
    habitat: DBRef('habitat', 'wk')
  },
  {
    _id: ObjectId('65b244bdc3fabc87a482bfd8'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
```

Практическое задание 4.2.1

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

```
db.unicorns.insert({name: 'Horny', dob: new Date(1992,2,13,7,47),
loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insert({name: 'Aurora', dob: new Date(1991, 0, 24, 13,
0), loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires:
43});

db.unicorns.insert({name: 'Unicrom', dob: new Date(1973, 1, 9, 22,
10), loves: ['energon', 'redbull'], weight: 984, gender: 'm',
vampires: 182});

db.unicorns.insert({name: 'Roooooodles', dob: new Date(1979, 7, 18,
18, 44), loves: ['apple'], weight: 575, gender: 'm', vampires: 99});

db.unicorns.insert({name: 'Solnara', dob: new Date(1985, 6, 4, 2, 1),
loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f',
vampires:80});

db.unicorns.insert({name:'Ayna', dob: new Date(1998, 2, 7, 8, 30),
loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires:
40});

db.unicorns.insert({name:'Kenny', dob: new Date(1997, 6, 1, 10, 42),
loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});

db.unicorns.insert({name: 'Raleigh', dob: new Date(2005, 4, 3, 0, 57),
loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});

db.unicorns.insert({name: 'Leia', dob: new Date(2001, 9, 8, 14, 53),
loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires:
33});

db.unicorns.insert({name: 'Pilot', dob: new Date(1997, 2, 1, 5, 3),
loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires:
54});

db.unicorns.insert ({name: 'Nimue', dob: new Date(1999, 11, 20, 16,
15), loves: ['grape', 'carrot'], weight: 540, gender: 'f'});

db.unicorns.insert ({name: 'Dunx', dob: new Date(1976, 6, 18, 18, 18),
loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires:
165});
```

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

Практическое задание 4.3.1

1. *Получите информацию о всех индексах коллекции* `unicorns`.
2. *Удалите все индексы, кроме индекса для идентификатора.*
3. *Попытайтесь удалить индекс для идентификатора.*

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

```
learn> db.unicorns.dropIndexes()
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
learn>
```

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

Практическое задание 4.4.1

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

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

2. *Выберите последних четыре документа.*
3. *Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра* `executionTimeMillis`*)*
4. *Создайте индекс для ключа* `value`*.*
5. *Получите информацию о всех индексах коллекции* `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})}
for(i = 0; i < 100000; i++){db.numbers.insert({value: i})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65b325c6c3fabc87a484467c') }
}
```

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('65b325c6c3fabc87a484467c'), value: 99999 },
  { _id: ObjectId('65b325c6c3fabc87a484467b'), value: 99998 },
  { _id: ObjectId('65b325c6c3fabc87a484467a'), value: 99997 },
  { _id: ObjectId('65b325c6c3fabc87a4844679'), value: 99996 }
]
learn>
```

```
learn> db.numbers.find().sort({value: -1}).limit(4).explain("executionStats")
{
  explainVersion: '2',
  queryPlanner: {
    namespace: 'learn.numbers',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: '70AEDD67',
```

```
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 60,
    totalKeysExamined: 0,
    totalDocsExamined: 100000,
```

```
learn> db.numbers.ensureIndex({ "value": 1 }, {"unique": true} )
[ 'value_1' ]
learn>
```

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

```
learn> db.numbers.find().sort({value: -1}).limit(4).explain("executionStats")
{
  explainVersion: '2',
```

```
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 6,
    totalKeysExamined: 4,
    totalDocsExamined: 4,
```

Время выполнения без индекса: 60 мс

Время выполнения с индексом: 4 мс
Время выполнения с индексом превосходит в 15 раз.

Вывод

В ходе лабораторной работы была освоена работа с СУБД MongoDB. Были проведены практические работы с CRUD-операциями, вложенными объектами, агрегациями, изменениями данных, ссылками и индексами.