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: Чернышев М.П.

Факультет: ИКТ

Группа: К3241 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.

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

CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ
2.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

Практическое задание 2.1.1:

1) Создать бд lerarn

```
use learn
```

```
> use learn
< switched to db learn
```

2) Заполнить коллекцию единорогов unicorns:

```
db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender:
'm', vampires: 63}); db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'],
weight: 450, gender:
'f', vampires: 43});
db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984,
gender: 'm', vampires: 182});
db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm',
vampires: 99});
db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'],
weight:550, gender:'f', vampires:80}); db.unicorns.insert({name:'Ayna', loves:
['strawberry', 'lemon'], weight: 733, gender:
'f', vampires: 40});
db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender:
'm', vampires: 39}); db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'],
weight: 421, gender:
'm', vampires: 2}); db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'],
weight: 601, gender:
'f', vampires: 33}); db.unicorns.insert({name: 'Pilot', loves: ['apple',
'watermelon'], weight: 650, gender:
'm', vampires: 54}); db.unicorns.insert({name: 'Nimue', loves: ['grape',
'carrot'], weight: 540, gender: 'f'});
```

```
> 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("65844cc08b3311aa43918291")
    }
  }
> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
  db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("65844ce18b3311aa43918293")
    }
  }
> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
  db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
  db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("65844d368b3311aa43918296")
    }
  }
```

```
> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
  db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
  db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
  db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
  db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("65844d4d8b3311aa4391829b")
    }
  }
```

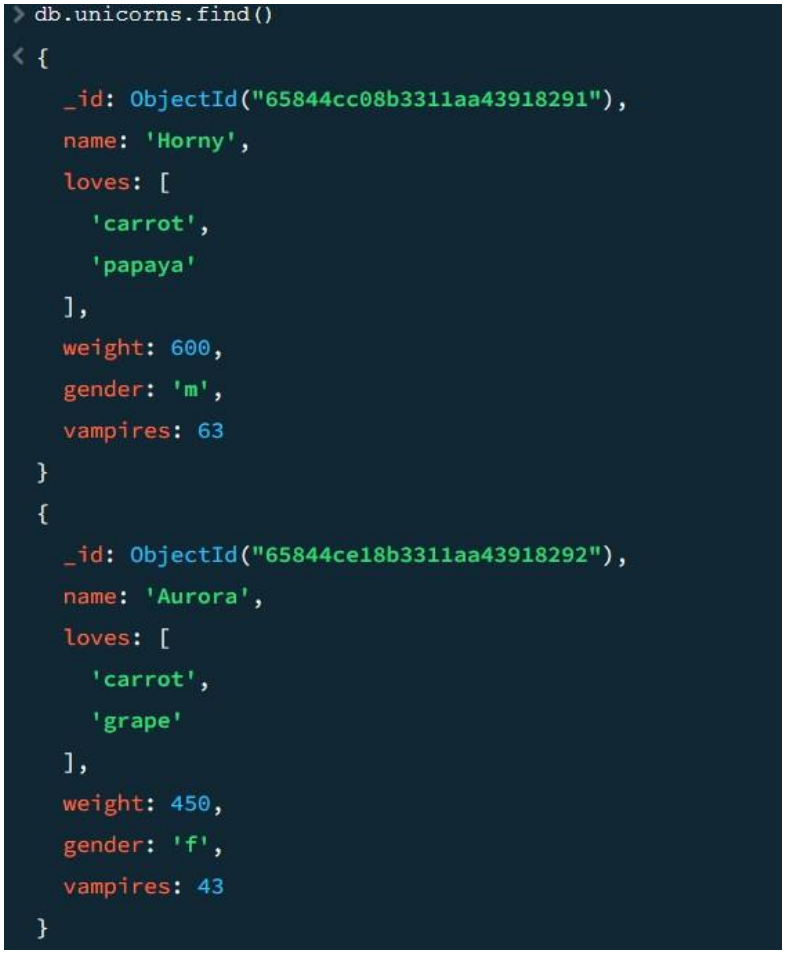
3) Используя второй способ, вставить в коллекцию единорогов документ:

```
document = ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm',
vampires: 165})
```

```
> document = ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
< {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
```

4) Проверить содержимое:

```
db.unicorns.find()
```



```
{
  _id: ObjectId("65844ce18b3311aa43918293"),
  name: 'Unicrom',
  loves: [
    'energon',
    'redbull'
  ],
  weight: 984,
  gender: 'm',
  vampires: 182
}
{
  _id: ObjectId("65844d368b3311aa43918294"),
  name: 'Roooooodles',
  loves: [
    'apple'
  ],
  weight: 575,
  gender: 'm',
  vampires: 99
}
```

```
{
  _id: ObjectId("65844d368b3311aa43918295"),
  name: 'Solnara',
  loves: [
    'apple',
    'carrot',
    'chocolate'
  ],
  weight: 550,
  gender: 'f',
  vampires: 80
}
{
  _id: ObjectId("65844d368b3311aa43918296"),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 733,
  gender: 'f',
  vampires: 40
}
```

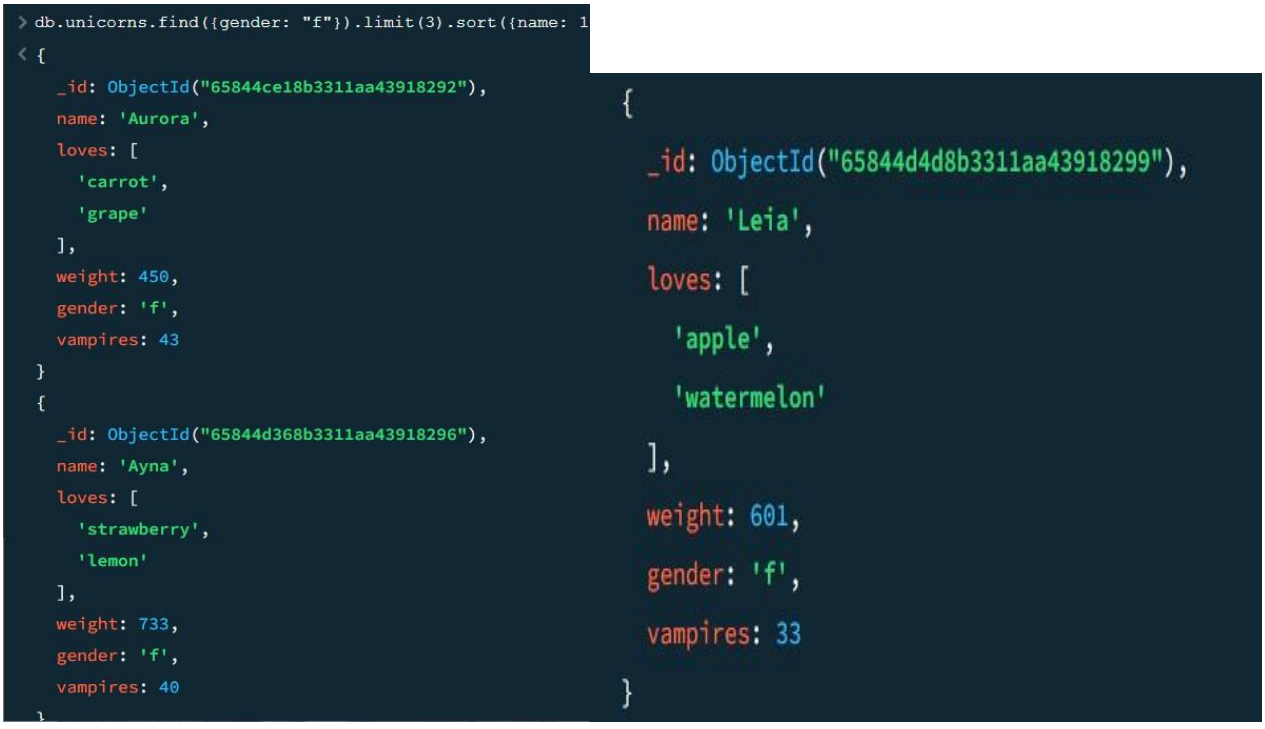
```
{
  _id: ObjectId("65844d4d8b3311aa43918297"),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 39
}
{
  _id: ObjectId("65844d4d8b3311aa43918298"),
  name: 'Raleigh',
  loves: [
    'apple',
    'sugar'
  ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
```

```
 {
   _id: ObjectId("65844d4d8b3311aa43918299"),
   name: 'Leia',
   loves: [
     'apple',
     'watermelon'
   ],
   weight: 601,
   gender: 'f',
   vampires: 33
 }
 {
   _id: ObjectId("65844d4d8b3311aa4391829a"),
   name: 'Pilot',
   loves: [
     'apple',
     'watermelon'
   ],
   weight: 650,
   gender: 'm',
   vampires: 54
 }
```

Практическое задание 2.2.1

1) Сформировать запросы для вывода списков самцов и самок единорогов. Ограничить список самок первыми тремя особями. Отсортировать списки по имени.

```
db.unicorns.find({gender: "f"}).limit(3).sort({name: 1})
```




```
db.unicorns.find({gender : "m"}).sort({name: 1})
```

```
{
  _id: ObjectId("65844cc08b3311aa43918291"),
  name: 'Horny',
  loves: [
    'carrot',
    'papaya'
  ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
{
  _id: ObjectId("65844d4d8b3311aa43918297"),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 39
}
```

```
{
  _id: ObjectId("65844d4d8b3311aa4391829a"),
  name: 'Pilot',
  loves: [
    'apple',
    'watermelon'
  ],
  weight: 650,
  gender: 'm',
  vampires: 54
}
{
  _id: ObjectId("65844d4d8b3311aa43918298"),
  name: 'Raleigh',
  loves: [
    'apple',
    'sugar'
  ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
```

```
{
  _id: ObjectId("65844d368b3311aa43918294"),
  name: 'Roooooodles',
  loves: [
    'apple'
  ],
  weight: 575,
  gender: 'm',
  vampires: 99
}
{
  _id: ObjectId("65844ce18b3311aa43918293"),
  name: 'Unicrom',
  loves: [
    'energon',
    'redbull'
  ],
  weight: 984,
  gender: 'm',
  vampires: 182
}
```

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

```
db.unicorns.find({loves: "carrot"})
```

```
> db.unicorns.find({loves: "carrot"})
< {
    _id: ObjectId("65844cc08b3311aa43918291"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId("65844ce18b3311aa43918292"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("65844d368b3311aa43918295"),
    name: 'Solnara',
    loves: [
      'apple',
      'carrot',
      'chocolate'
    ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
  {
    _id: ObjectId("65844d4d8b3311aa4391829b"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
```

```
db.unicorns.findOne({loves: "carrot"})
```

```
> db.unicorns.findOne({loves: "carrot"})
< {
    _id: ObjectId("65844cc08b3311aa43918291"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
```

```
db.unicorns.find({loves: "carrot"}).limit(1)
```

```
> db.unicorns.find({loves: "carrot"}).limit(1)
< {
    _id: ObjectId("65844cc08b3311aa43918291"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
```

Практическое задание 2.2.2:

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

```
db.unicorns.find({gender : "m"}, {loves : 0, gender : 0}).sort({name: 1})
```

```
> db.unicorns.find({gender : "m"}, {loves : 0, gender : 0}).sort({name: 1})
< {
    _id: ObjectId("65844cc08b3311aa43918291"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
  {
    _id: ObjectId("65844d4d8b3311aa43918297"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  }
  {
    _id: ObjectId("65844d4d8b3311aa4391829a"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  }
  {
    _id: ObjectId("65844d4d8b3311aa43918298"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  }
```

```
{
  _id: ObjectId("65844d368b3311aa43918294"),
  name: 'Roooooodles',
  weight: 575,
  vampires: 99
}
{
  _id: ObjectId("65844ce18b3311aa43918293"),
  name: 'Unicrom',
  weight: 984,
  vampires: 182
}
```

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

```
db.unicorns.find().sort({ $natural: -1 })
```

```
> db.unicorns.find().sort({ $natural: -1 })
< {
    _id: ObjectId("65844d4d8b3311aa4391829b"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("65844d4d8b3311aa4391829a"),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
  {
    _id: ObjectId("65844d4d8b3311aa43918299"),
    name: 'Leia',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
  {
    _id: ObjectId("65844d4d8b3311aa43918298"),
    name: 'Raleigh',
    loves: [
      'apple',
      'sugar'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```


{
_id: ObjectId("65844d4d8b3311aa43918297"),
name: 'Kenny',
loves: [
'grape',
'lemon'
],
weight: 690,
gender: 'm',
vampires: 39
}
{
_id: ObjectId("65844d368b3311aa43918296"),
name: 'Ayna',
loves: [
'strawberry',
'lemon'
],
weight: 733,
gender: 'f',
vampires: 40
}
{
_id: ObjectId("65844d368b3311aa43918295"),
name: 'Solnara',
loves: [
'apple',
'carrot',
'chocolate'
],
weight: 550,
gender: 'f',
vampires: 80
}
{
_id: ObjectId("65844d368b3311aa43918294"),
name: 'Roooooodles',
loves: [
'apple'
],
weight: 575,
gender: 'm',
vampires: 99
}

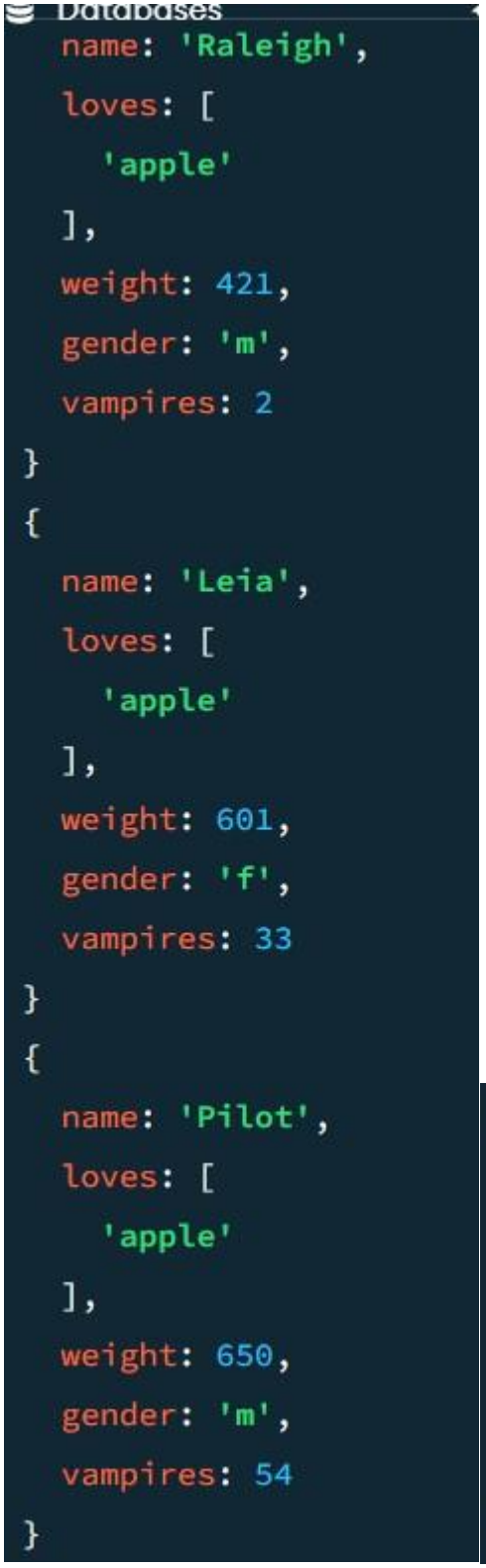
Практическое задание 2.1.4:

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

```
db.unicorns.find({}, {loves : {$slice : 1}, _id : 0}
```

```
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
}
```



2.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 2.3.1:

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

```
db.unicorns.find({gender : "f", weight : {$gte : 500, $lte : 700}}, {_id: 0})
```

```
> 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.

```
db.unicorns.find({gender : "m", weight : {$gte : 500}, loves : {$all :['grape',
'lemon']}}, {_id : 0})
```

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

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ vampires.

```
db.unicorns.find({vampires : {$exists:false}})
```

```
> db.unicorns.find({vampires : {$exists:false}})
< {
    _id: ObjectId("65844d4d8b3311aa4391829b"),
    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.

```
db.unicorns.find({gender : "m"}, {loves: {$slice : 1}}).sort({name : 1})
```

```
> db.unicorns.find({gender : 'm'}, {loves : { $slice : 1}}).sort({name : 1})
< {
    _id: ObjectId('6596a3f09dbc95074933cbe7'),
    name: 'Horny',
    loves: [
      'carrot'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId('6596a3f09dbc95074933cbed'),
    name: 'Kenny',
    loves: [
      'grape'
    ],
    weight: 690,
    gender: 'm',
    vampires: 39
```

3.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 3.1.1:

Создайте коллекцию towns

```
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"}}
])
```

```
> 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("658556238b3311aa4391829c"),
```

```
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("658556238b3311aa4391829c"),
      '1': ObjectId("658556238b3311aa4391829d"),
      '2': ObjectId("658556238b3311aa4391829e")
    }
  }
```

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

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

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

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

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

ИСПОЛЬЗОВАНИЕ JAVASCRIPT

Практическое задание 3.1.2:

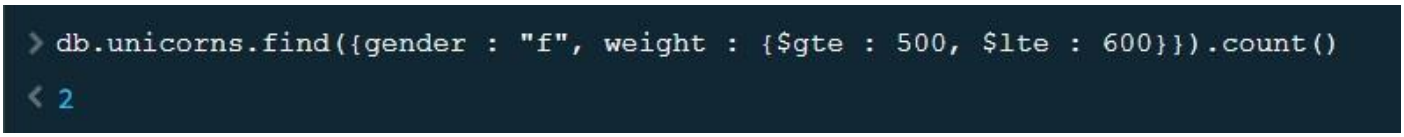
```
func = function() {return db.unicorns.find({gender: 'm'}).limit(2).sort({name:1})}
var cursor = func();null;
cursor.forEach(function(obj){
    print(obj.name);
});
```

3.2 АГРЕГИРОВАННЫЕ ЗАПРОСЫ

Практическое задание 3.2.1: Вывести

количество самок единорогов весом от полутонны до 600 кг.

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



Практическое задание 3.2.2:

Вывести список предпочтений

```
db.unicorns.distinct("loves")
```

```
> 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.

```
db.unicorns.aggregate({"$group":{_id:"$gender", count : {$sum: 1}}})
```

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

3.3 РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 3.3.1:

Команда save устарела:

Updates an existing document or inserts a new document, depending on its `document` parameter.

> **NOTE**
>
> Starting in MongoDB 4.2, the `db.collection.save()` method is deprecated. Use `db.collection.insertOne()` or `db.collection.replaceOne()` instead.

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

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.

```
db.unicorns.updateOne({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
```

```
> db.unicorns.updateOne({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name : "Ayna"})
< {
    _id: ObjectId("65844d368b3311aa43918296"),
    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.
Проверить содержимое коллекции unicorns.

```
db.unicorns.update({name : "Raleigh"}, {$set : {"loves" : ["redbull"]}})
```

```
> db.unicorns.update({name : "Raleigh"}, {$set : {"loves" : ["redbull"]}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name : "Raleigh"})
< {
    _id: ObjectId("65844d4d8b3311aa43918298"),
    name: 'Raleigh',
    loves: [
      'redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```

Практическое задание 3.3.4:
Всем самцам единорогов увеличить количество убитых вапмиров на 5.
Проверить содержимое коллекции unicorns.

```
db.unicorns.updateMany({gender : "m"}, {$inc : {vampires : 5}})
```

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

```
> db.unicorns.find({gender : "m"})
< {
    _id: ObjectId("65844ce18b3311aa43918293"),
    name: 'Unicrom',
    loves: [
      'energon',
      'redbull'
    ],
    weight: 984,
    gender: 'm',
    vampires: 187
  }
  {
    _id: ObjectId("65844d368b3311aa43918294"),
    name: 'Roooooodles',
    loves: [
      'apple'
    ],
    weight: 575,
    gender: 'm',
    vampires: 104
  }
```

```
{
  _id: ObjectId("65844d4d8b3311aa43918297"),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 44
}
{
  _id: ObjectId("65844d4d8b3311aa43918298"),
  name: 'Raleigh',
  loves: [
    'redbull'
  ],
  weight: 421,
  gender: 'm',
  vampires: 7
}
```

```
{
  _id: ObjectId("65844d4d8b3311aa4391829a"),
  name: 'Pilot',
  loves: [
    'apple',
    'watermelon'
  ],
  weight: 650,
  gender: 'm',
  vampires: 59
}
{
  _id: ObjectId("6585d684cab8f5245897b12d"),
  name: 'Dunx',
  loves: [
    'grape',
    'watermelon'
  ],
  weight: 704,
  gender: 'm',
  vampires: 170
}
```



```
{
  _id: ObjectId("6585df5bcab8f5245897b12e"),
  name: 'Horny',
  loves: [
    'carrot',
    'papaya'
  ],
  weight: 600,
  gender: 'm',
  vampires: 68
}
```


Практическое задание 3.3.5:
Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.
Проверить содержимое коллекции towns.

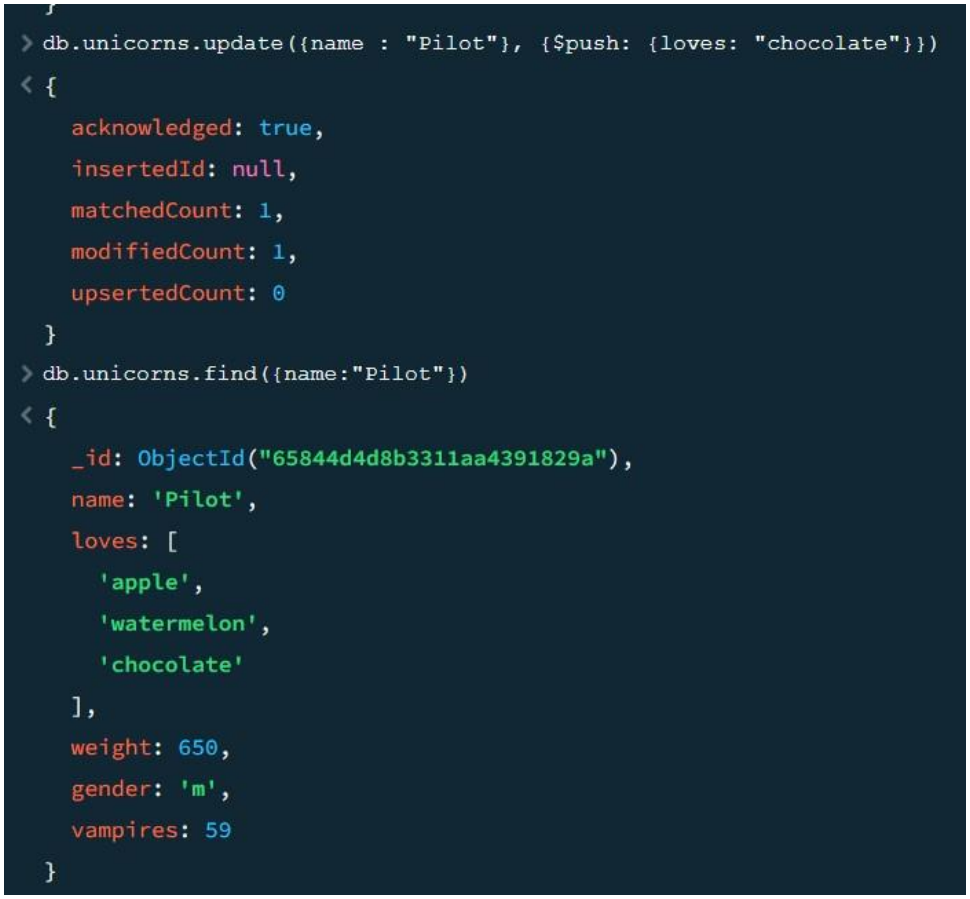
```
db.towns.update({name : "Portland"}, {$unset: {"mayor.party" : 1}})
```

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

```
> db.towns.find({name:"Portland"})
< {
    _id: ObjectId("658556238b3311aa4391829e"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams'
    }
  }
```

Практическое задание 3.3.6:
Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.
Проверить содержимое коллекции unicorns.

```
db.unicorns.update({name : "Pilot"}, {$push: {loves: "chocolate"}})
```



Практическое задание 3.3.7:
Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.
Проверить содержимое коллекции unicorns.

```
db.unicorns.update({name : "Aurora"},{$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
```

```
> db.unicorns.update({name : "Aurora"},
  {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name:"Aurora"})
< {
    _id: ObjectId("65844ce18b3311aa43918292"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

3.4 УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

Практическое задание 3.4.1:
Удалите документы с беспартийными мэрами.
Проверьте содержание коллекции.
Очистите коллекцию.
Просмотрите список доступных коллекций.

```
db.towns.deleteMany({"mayor.party" : {$exists : false}})
```

```
}
> db.towns.deleteMany({"mayor.party" : {$exists : false}})
< {
    acknowledged: true,
    deletedCount: 1
  }
> db.towns.find()
< {
    _id: ObjectId("658556238b3311aa4391829d"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: 2009-07-31T00:00:00.000Z,
    famous_for: [
      'status of liberty',
      'food'
    ],
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
  {
    _id: ObjectId("6585eaf7cab8f5245897b130"),
```

```
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams',
      party: 'D'
    }
  }
```

```
db.towns.deleteMany({})
```

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

4.1 ССЫЛКИ В БД

Практическое задание 4.1.1:

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

```
db.zones.insertMany([ {_id: "forest", name: "Magic Forest", description: "Enchanted forest where mystical unicorns roam freely"}, {_id: "meadow", name: "Sparkling Meadow", description: "Beautiful meadow with lush grass and vibrant flowers, a favorite grazing spot for unicorns"}, {_id: "mountain", name: "Celestial Mountain", description: "Majestic mountain range inhabited by wise and ancient unicorns"}, {_id: "lake", name: "Crystal Lake", description: "Serene lake surrounded by shimmering trees, a peaceful retreat for reflective unicorns"} ]);
```

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

```
db.unicorns.updateOne({_id: ObjectId("65844d4d8b3311aa4391829a")}, {$set: {habitat:{$ref : "zones", $id:"forest"}}})
```


```
db.unicorns.updateOne({_id: ObjectId("65844d4d8b3311aa43918297")}, {$set: {habitat:{$ref : "zones", $id:"mountain"}}})
```

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
db.unicorns.updateOne({_id: ObjectId("6585d684cab8f5245897b12d")}, {$set:
{habitat:{$ref : "zones", $id:"lake"}}})
```

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Проверьте содержание коллекции единорогов

```
db.unicorns.find({name: "Horny"})
{
  _id: ObjectId('6596a3f09dbc95074933cbe7'),
  name: 'Horny',
  loves: [
    'carrot',
    'papaya'
  ],
  weight: 600,
  gender: 'm',
  vampires: 73,
  habitat: DBRef('zones', 'forest')
}
```

4.2 НАСТРОЙКА ИНДЕКСОВ Практическое задание 4.2.1:
Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique.

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

Практическое задание 4.3.1:
Получите информацию о всех индексах коллекции unicorns .

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

Удалите все индексы, кроме индекса для идентификатора.

```
db.unicorns.dropIndexes()
```

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

Попытайтесь удалить индекс для идентификатора.

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

4.4 ПЛАН ЗАПРОСА

Практическое задание 4.4.1:
Создайте объемную коллекцию numbers, задействовав курсор:

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

```
> db.createCollection("num")
< { ok: 1 }
> for(i = 0; i < 100000;i++){db.num.insert({value : i})}
< DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('659925887b95069573d6d0bf')
    }
  }
```

Выберите последних четыре документа.

```
> var lastFourDocs = db.numbers.find().sort({value: -1}).limit(4);
  lastFourDocs.forEach(printjson);
  var executionStats = db.numbers.find().sort({value: -1}).limit(4).explain("executionStats");
  print(executionStats.executionStats.executionTimeMillis);
< { _id: ObjectId("6585f892cab8f524589937d0"), value: 99999 }
< { _id: ObjectId("6585f892cab8f524589937cf"), value: 99998 }
< { _id: ObjectId("6585f892cab8f524589937ce"), value: 99997 }
< { _id: ObjectId("6585f892cab8f524589937cd"), value: 99996 }
< 227
```

Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis) 227.

Создайте индекс для ключа value.
Получите информацию о всех индексах коллекции numbres.
Выполните запрос 2.

```
> db.numbers.createIndex({value: 1});
< value_1
> db.numbers.getIndexes();
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { value: 1 }, name: 'value_1' }
  ]
> var lastFourDocs = db.numbers.find().sort({value: -1}).limit(4);
> var executionStats = db.numbers.find().sort({value: -1}).limit(4).explain("executionStats");
  print(executionStats.executionStats.executionTimeMillis);
< 41
```

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

41

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

Конечно, с индексом быстрее (41 < 227). Это связано с тем, что индекс позволяет базе данных эффективно находить и сортировать данные, не просматривая каждый документ.

Вывод:

В ходе выполнения данной лабораторной работы мы получили практические навыки работы с CRUD-операциями (создание, чтение, обновление, удаление данных) в MongoDB. Мы изучили основные команды для выполнения этих операций в коллекциях MongoDB. Также была освоена работа с вложенными объектами в коллекциях, выполнение агрегаций данных и изменение данных с помощью ссылок и индексов. В результате приобретён практический опыт в выполнении различных операций с данными в MongoDB.