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

Отчет

по Лабораторной Работе № 6

по дисциплине **«Базы Данных»**

Автор: Акулов Даниил Даниилович

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова Марина Михайловна

Санкт-Петербург, 2023

Содержание работы

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 работы:

Практическое задание 2.1.1:

1. *Создайте базу данных learn.*

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

2. *Заполните коллекцию единорогов unicorns:*

```
> 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("657f010ca944b86e5356b995")
    }
  }
> 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'});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("657f012fa944b86e5356b99f")
    }
  }
learn>
```

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

```
> document=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
< {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
> db.unicorns.insert(document)
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("657f02dea944b86e5356b9a0")
    }
  }
learn>
```

4.*Проверьте содержимое коллекции с помощью метода find.*

```
> db.unicorns.find()
< {
    _id: ObjectId("657f010ca944b86e5356b995"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId("657f012fa944b86e5356b996"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("657f012fa944b86e5356b997"),
    name: 'Unicrom',
    loves: [
```

```
  name: 'Unicrom',
  loves: [
    'energon',
    'redbull'
  ],
  weight: 984,
  gender: 'm',
  vampires: 182
}
{
  _id: ObjectId("657f012fa944b86e5356b998"),
  name: 'Roooooodles',
  loves: [
    'apple'
  ],
  weight: 575,
  gender: 'm',
  vampires: 99
}
{
  _id: ObjectId("657f012fa944b86e5356b999"),
  name: 'Solnara',
  loves: [
    'apple',
    'carrot',
    'chocolate'
  ],
```

```
  weight: 550,
  gender: 'f',
  vampires: 80
}
{
  _id: ObjectId("657f012fa944b86e5356b99a"),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 733,
  gender: 'f',
  vampires: 40
}
{
  _id: ObjectId("657f012fa944b86e5356b99b"),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 39
}
{
```

```
  {
    _id: ObjectId("657f012fa944b86e5356b99c"),
    name: 'Raleigh',
    loves: [
      'apple',
      'sugar'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99d"),
    name: 'Leia',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99e"),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99f"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("657f02dea944b86e5356b9a0"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
learn>
```

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.*

```
> db.unicorns.find({gender:'f'}).sort({name:1}).limit(3)
< {
    _id: ObjectId("657f012fa944b86e5356b996"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99a"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99d"),
    name: 'Leia',
    loves: [
```

```
> db.unicorns.find({gender:'m'}).sort({name:1})
< {
    _id: ObjectId("657f02dea944b86e5356b9a0"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("657f010ca944b86e5356b995"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99b"),
    name: 'Kenny',
    loves: [
```

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

```
> db.unicorns.find({gender:'f', loves:'carrot'}).limit(1)
< {
    _id: ObjectId("657f012fa944b86e5356b996"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
learn>
```

Практическое задание 2.2.2:

1. *Модифицируйте 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})
< {
    _id: ObjectId("657f02dea944b86e5356b9a0"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
  {
    _id: ObjectId("657f010ca944b86e5356b995"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99b"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99e"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99c"),
```

```
    _id: ObjectId("657f012fa944b86e5356b99c"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  }
  {
    _id: ObjectId("657f012fa944b86e5356b998"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  }
  {
    _id: ObjectId("657f012fa944b86e5356b997"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
learn>
```

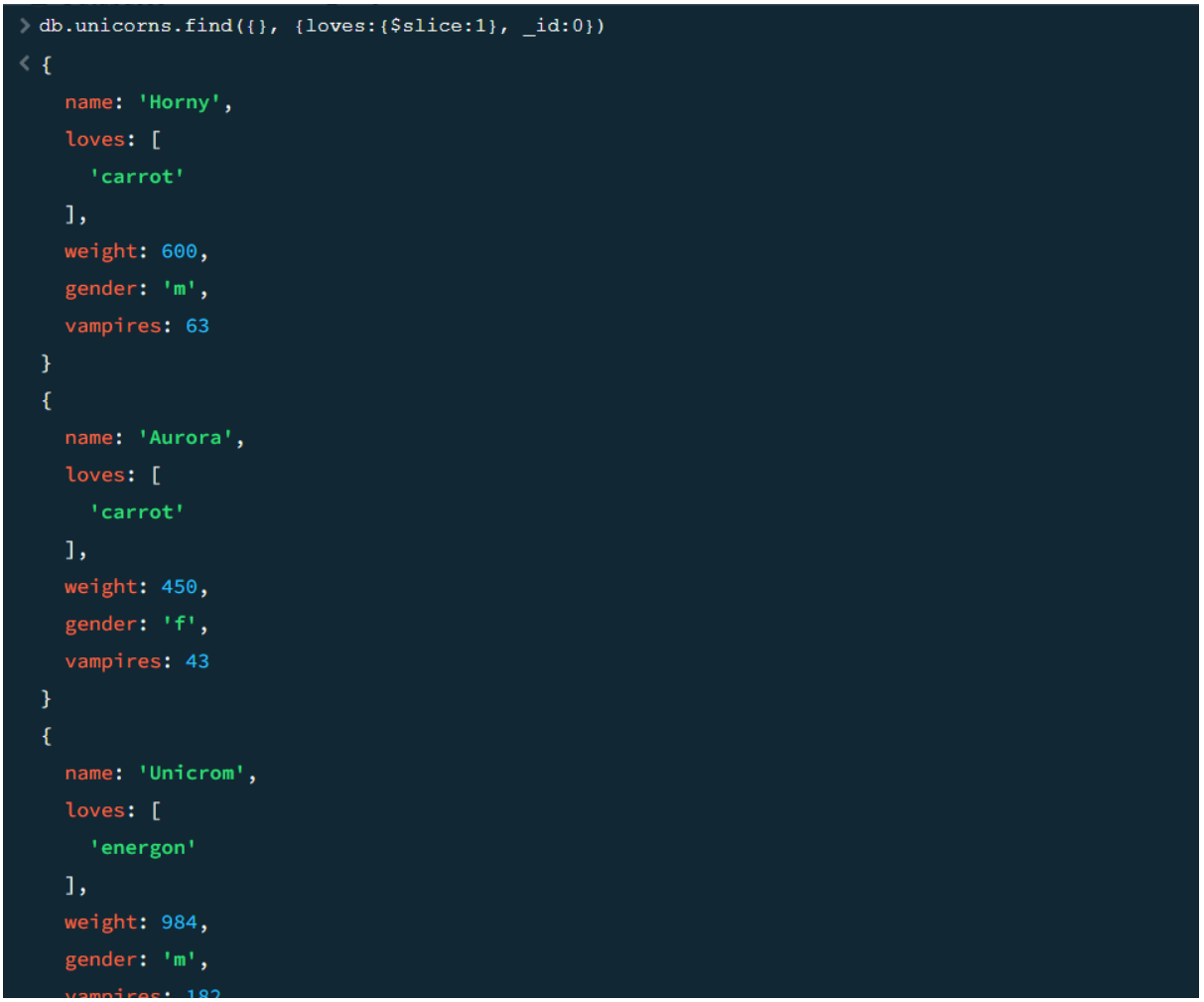
Практическое задание 2.2.3:

1. *Вывести список единорогов в обратном порядке добавления.*

```
> db.unicorns.find().sort({_id:-1})
< {
    _id: ObjectId("657f02dea944b86e5356b9a0"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99f"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("657f012fa944b86e5356b99e"),
    name: 'Pilot',
    loves: [
```

Практическое задание 2.2.4:

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



```
> 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
```

Практическое задание 2.3.1:

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})
< {
    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',
```

Практическое задание 2.3.2:

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

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

Практическое задание 2.3.3:

1. *Найти всех единорогов, не имеющих ключ* `vampires`

```
> db.unicorns.find({vampires:{$exists:false}})
< {
    _id: ObjectId("657f012fa944b86e5356b99f"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
learn>
```

Практическое задание 2.3.4:

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

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

```
  }
  {
    name: 'Pilot',
    loves: [
      'apple'
    ]
  }
  {
    name: 'Raleigh',
    loves: [
      'apple'
    ]
  }
  {
    name: 'Roooooodles',
    loves: [
      'apple'
    ]
  }
  {
    name: 'Unicrom',
    loves: [
      'energon'
    ]
  }
learn>
```

Практическое задание 3.1.1:

1. *Создайте коллекцию towns, включающую следующие документы:*

```
> db.towns.insertOne({name: "Punxsutawney ",
  populatiuon: 6200,
  last_sensus: ISODate("2008-01-31"),
  famous_for: [""],
  mayor: {
     name: "Jim Wehrle"
  }})
< {
    acknowledged: true,
    insertedId: ObjectId("657f5ac7e005fe5ec03475b8")
  }
> db.towns.insertOne({name: "New York",
  populatiuon: 22200000,
  last_sensus: ISODate("2009-07-31"),
  famous_for: ["status of liberty", "food"],
  mayor: {
     name: "Michael Bloomberg",
  party: "I"}}
  )
< {
    acknowledged: true,
    insertedId: ObjectId("657f5b06e005fe5ec03475b9")
  }
> db.towns.insertOne({name: "Portland",
  populatiuon: 528000,
  last_sensus: ISODate("2009-07-20"),
  famous_for: ["beer", "food"],
```

```
> db.towns.insertOne({name: "Portland",
  populatiuon: 528000,
  last_sensus: ISODate("2009-07-20"),
  famous_for: ["beer", "food"],
  mayor: {
     name: "Sam Adams",
  party: "D"}}
  )
< {
    acknowledged: true,
    insertedId: ObjectId("657f5b13e005fe5ec03475ba")
  }
learn>
```

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.party':1,'mayor.name':1,_id:0})
< {
    name: 'New York',
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
learn>
```

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.name':1,_id:0})
< {
    name: 'Punxsutawney ',
    mayor: {
      name: 'Jim Wehrle'
    }
  }
learn>
```

Практическое задание 3.1.2:

1. *Сформировать функцию для вывода списка самцов единорогов. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке. Вывести результат, используя forEach.*

```
> function maleUnicornsList(){
  let cursor = db.unicorns.find({gender:'m'});null;
  cursor.sort({name:1}).limit(2);null;
  cursor.forEach(el => print(el.name))
  }
< [Function: maleUnicornsList]
> maleUnicornsList()
< Dunx
< Horny
learn>
```

Практическое задание 3.2.1:

1. *Вывести количество самок единорогов весом от полутонны до 600 кг.*

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

Практическое задание 3.2.2:

1. *Вывести список предпочтений.*

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

Практическое задание 3.2.3:

1. *Посчитать количество особей единорогов обоих полов.*

```
> 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'})
```

2. *Проверить содержимое коллекции* `unicorns`.

```
    vampires: 165
  }
  {
    _id: ObjectId("657f6a38e005fe5ec03475bb"),
    name: 'Barny',
    loves: [
      'grape'
    ],
    weight: 340,
    gender: 'm'
  }
learn>
```

Практическое задание 3.3.2:

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

2. Проверить содержимое коллекции `unicorns`*.*

Практическое задание 3.3.3:

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

2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.updateOne({name:'Raleigh'}, {$set:{loves:['redbull']}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name:'Raleigh'})
< {
    _id: ObjectId("657f012fa944b86e5356b99c"),
    name: 'Raleigh',
    loves: [
      'redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
learn>
```

Практическое задание 3.3.4:

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

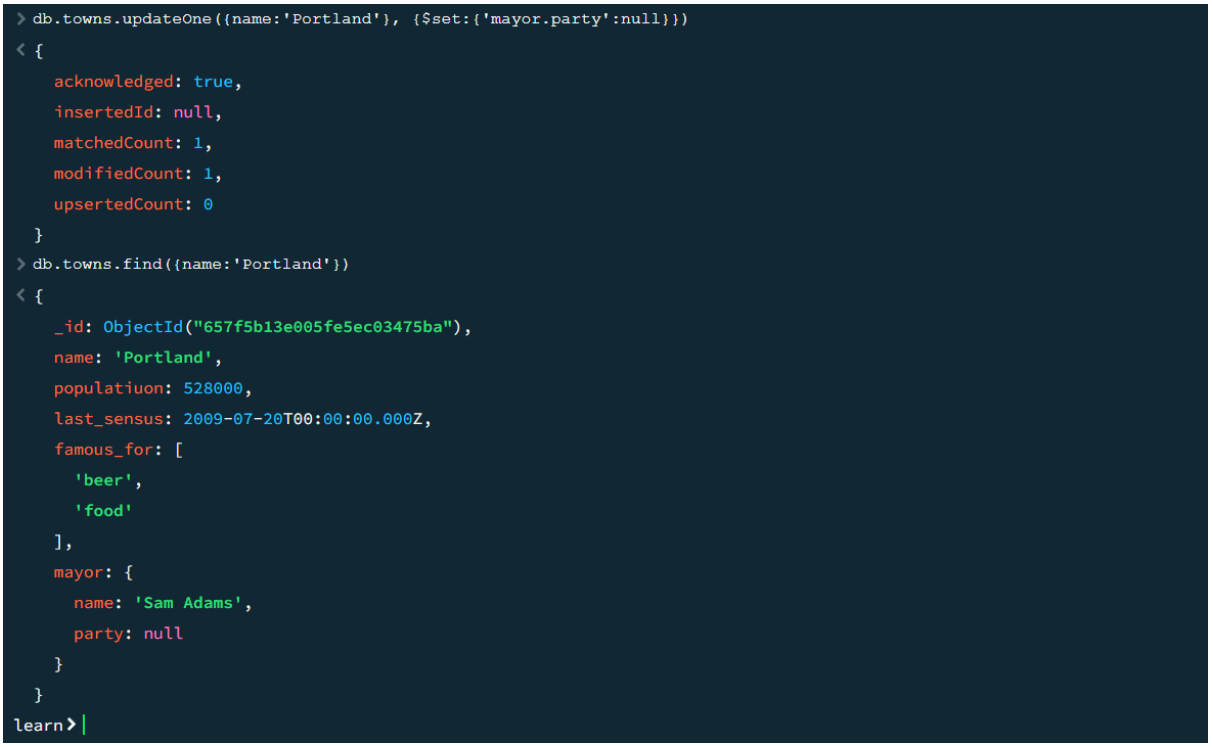
2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.updateMany({gender:'m'}, {$inc:{vampires:5}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 8,
    modifiedCount: 8,
    upsertedCount: 0
  }
> db.unicorns.find({gender:'m'})
< {
    _id: ObjectId("657f010ca944b86e5356b995"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
  {
    _id: ObjectId("657f012fa944b86e5356b997"),
    name: 'Unicrom',
    loves: [
      'energon',
      'redbull'
    ],
```

Практическое задание 3.3.5:

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

2. *Проверить содержимое коллекции* `towns`.



Практическое задание 3.3.6:

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

2. *Проверить содержимое коллекции* `unicorns`.

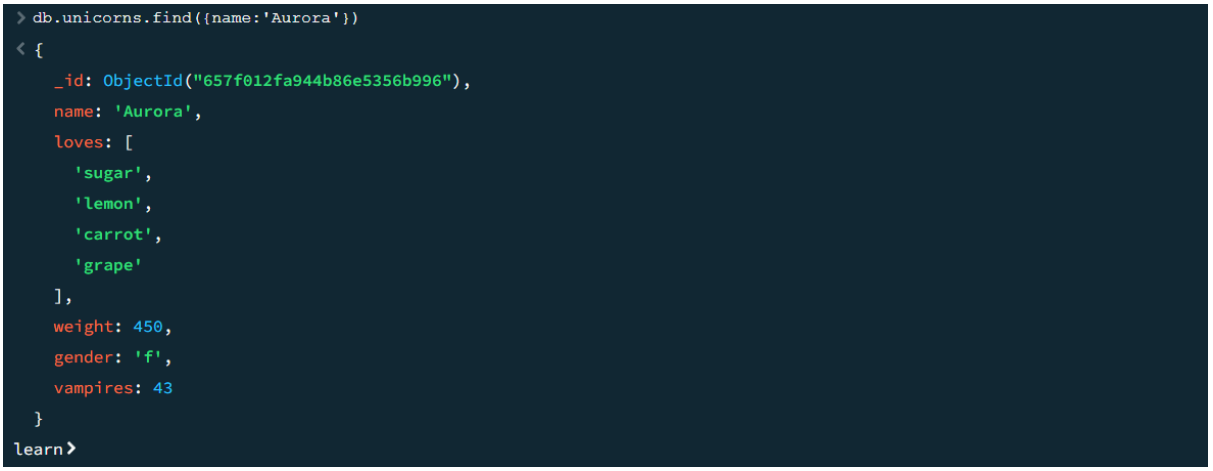
```
> db.unicorns.updateOne({name:'Pilot'}, {$addToSet:{loves:'chocolate'}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name:'Pilot'})
< {
    _id: ObjectId("657f012fa944b86e5356b99e"),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon',
      'chocolate'
    ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
learn >
```

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`.

```
> db.unicorns.updateOne({name:'Aurora', gender:'f'}, {$addToSet:{loves:{$each:['sugar', 'lemon']}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```



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"}}
```

```
  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("657f7266e005fe5ec03475bc"),
      '1': ObjectId("657f7266e005fe5ec03475bd"),
      '2': ObjectId("657f7266e005fe5ec03475be")
    }
  }
learn>
```

2. *Удалите документы с беспартийными мэрами.*

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

3. *Проверьте содержание коллекции.*

```
> db.towns.find()
< {
    _id: ObjectId("657f7378e005fe5ec03475c0"),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: 2009-07-31T00:00:00.000Z,
    famous_for: [
      'status of liberty',
      'food'
    ],
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
  {
    _id: ObjectId("657f7378e005fe5ec03475c1"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams',
```

4. *Очистите коллекцию.*

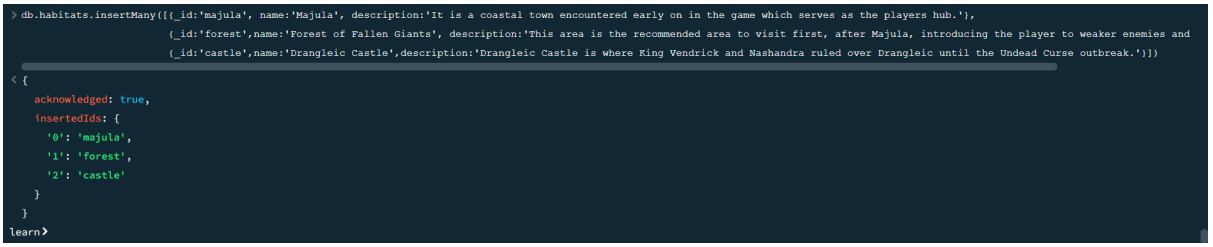
```
> db.towns.deleteMany({})
< {
    acknowledged: true,
    deletedCount: 2
  }
learn>
```

5. *Просмотрите список доступных коллекций.*

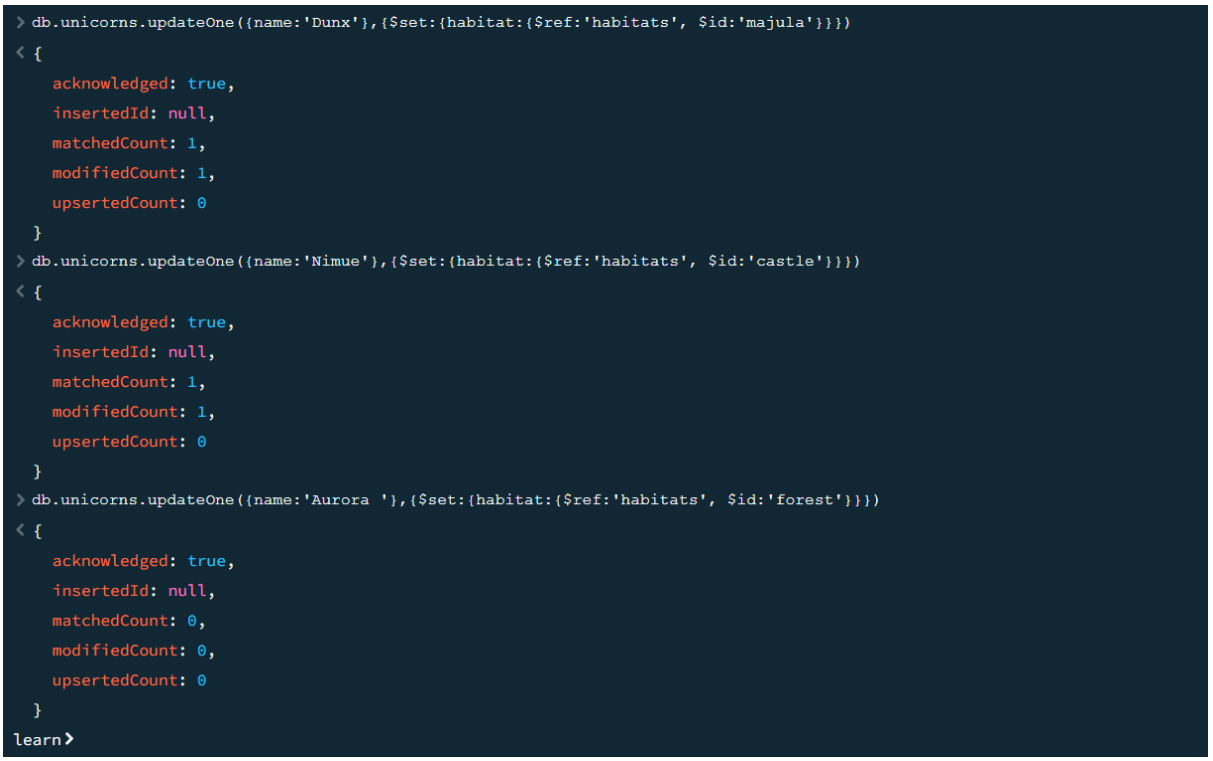
```
> show collections
< towns
  unicorns
learn>
```

Практическое задание 4.1.1:

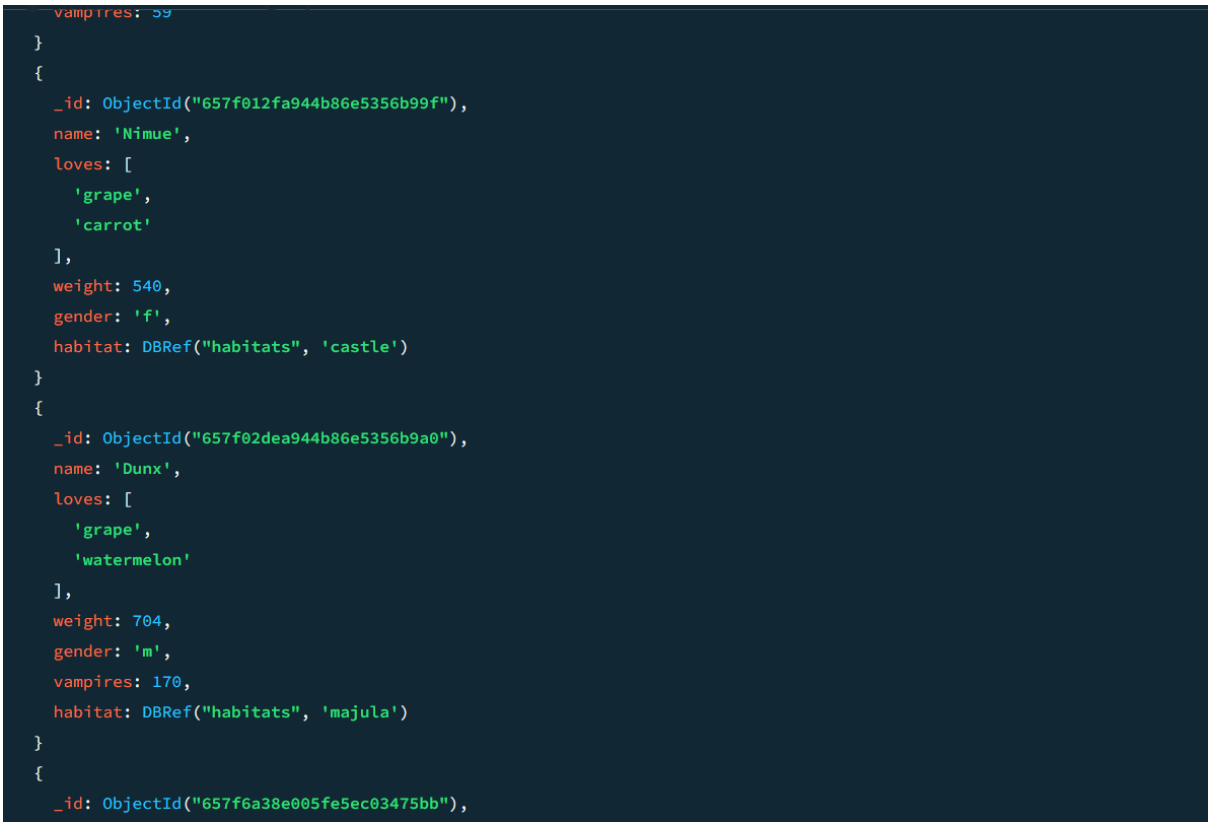
1. *Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.*



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



3. *Проверьте содержание коллекции единорогов.*



```
  vampires: 59
}
{
  _id: ObjectId("657f012fa944b86e5356b99f"),
  name: 'Nimue',
  loves: [
    'grape',
    'carrot'
  ],
  weight: 540,
  gender: 'f',
  habitat: DBRef("habitats", 'castle')
}
{
  _id: ObjectId("657f02dea944b86e5356b9a0"),
  name: 'Dunx',
  loves: [
    'grape',
    'watermelon'
  ],
  weight: 704,
  gender: 'm',
  vampires: 170,
  habitat: DBRef("habitats", 'majula')
}
{
  _id: ObjectId("657f6a38e005fe5ec03475bb"),
```

Практическое задание 4.2.1:

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

```
> db.unicorns.createIndex({name:1}, {unique:true})
< name_1
> db.unicorns.find()
< {
    _id: ObjectId("657f010ca944b86e5356b995"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
  {
    _id: ObjectId("657f012fa944b86e5356b996"),
    name: 'Aurora',
    loves: [
      'sugar',
      'lemon',
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
```

Практическое задание 4.3.1:

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>
```

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

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

3. *Попытайтесь удалить индекс для идентификатора.*

```
> db.unicorns.dropIndex('_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})}
```

```
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
< DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("657f844afa02196e61f7f9c2")
    }
  }
learn>
```

2. *Выберите последних четыре документа.*

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

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

```
    executionStats: {
      executionSuccess: true,
      nReturned: 4,
      executionTimeMillis: 102,
      totalKeysExamined: 0,
      totalDocsExamined: 100000,
```

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

```
> db.numbers.createIndex({value:1},{'unique':true})
< value_1
learn>
```

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

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

6. *Выполните запрос 2.*

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

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

```
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 16,
    totalKeysExamined: 4,
    totalDocsExamined: 4,
    executionStages: {
```

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

Без индекса запрос выполнился за 102 миллисекунды, а с индексом – за 16. Время выполнения уменьшилось в 6,375 раз.

Вывод

В ходе данной лабораторной работы я научился работать в СУБД MongoDB. Я овладел практическими навыками работы с CRUD-операциями, со вложенными объектами в коллекции, с агрегациями и изменениями данных, с ссылками и индексами в базе данных MongoDB.