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:Джаватов И.А.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

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('657f82842442d8bc060bc4b0') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
learn>
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b2') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b3') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b4') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b5') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b6') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b7') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
```

```
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b7') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b8') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4b9') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f82852442d8bc060bc4ba') }
}
```

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('657f837d2442d8bc060bc4bb') }
}
learn>
```

4) Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('657f82842442d8bc060bc4b0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657f82842442d8bc060bc4b1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
learn>
    _id: ObjectId('657f82852442d8bc060bc4b3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b4'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
```


{
_id: ObjectId('657f82852442d8bc060bc4b8'),
name: 'Leia',
loves: ['apple', 'watermelon'],
weight: 601,
gender: 'f',
vampires: 33
},
{
_id: ObjectId('657f82852442d8bc060bc4b9'),
name: 'Pilot',
loves: ['apple', 'watermelon'],
weight: 650,
gender: 'm',
vampires: 54
},
{
_id: ObjectId('657f82852442d8bc060bc4ba'),
name: 'Nimue',
loves: ['grape', 'carrot'],
weight: 540,
gender: 'f'
},
{
_id: ObjectId('657f837d2442d8bc060bc4bb'),
name: 'Dunx',
loves: ['grape', 'watermelon'],
weight: 704,
gender: 'm',
vampires: 165
}
earn>

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('657f837d2442d8bc060bc4bb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657f82842442d8bc060bc4b0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b7'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
```

Список самок:

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('6576428c2255ab0de2acaf67'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6576429f2255ab0de2acaf6b'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657642ac2255ab0de2acaf6e'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

findOne:

```
learn> db.unicorns.findOne({gender: "f", loves: 'carrot'})
{
  _id: ObjectId('657f82842442d8bc060bc4b1'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

Limit:

```
learn> db.unicorns.find({gender: "f", loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId('657f82842442d8bc060bc4b1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn> _
```

Практическое задание 2.2.2:

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

```
learn> db.unicorns.find({gender: "m"}, {loves:0, gender:0}).sort({name: 1})
[
  {
    _id: ObjectId('657f837d2442d8bc060bc4bb'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('657f82842442d8bc060bc4b0'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b6'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b9'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b7'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b3'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b2'),
    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('657f837d2442d8bc060bc4bb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4ba'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b8'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b7'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b4'),
    name: 'Solnara',
```


_id: ObjectId('657f82852442d8bc060bc4b4'),
name: 'Solnara',
loves: ['apple', 'carrot', 'chocolate'],
weight: 550,
gender: 'f',
vampires: 80
},
{
_id: ObjectId('657f82852442d8bc060bc4b3'),
name: 'Roooooodles',
loves: ['apple'],
weight: 575,
gender: 'm',
vampires: 99
},
{
_id: ObjectId('657f82852442d8bc060bc4b2'),
name: 'Unicrom',
loves: ['energon', 'redbull'],
weight: 984,
gender: 'm',
vampires: 182
},
{
_id: ObjectId('657f82842442d8bc060bc4b1'),
name: 'Aurora',
loves: ['carrot', 'grape'],
weight: 450,
gender: 'f',
vampires: 43
},
{
_id: ObjectId('657f82842442d8bc060bc4b0'),
name: 'Horny',
loves: ['carrot', 'papaya'],
weight: 600,
gender: 'm',
vampires: 63
}
earn>

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
}

earn>
```

Практическое задание 2.3.1

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

```
learn> db.unicorns.find({weight: {$gt:500, $lt: 700}}, {_id: 0})
[
  {
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    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

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId('657642b52255ab0de2acaf70'),
    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"}, {name: 1, loves: {$slice: 1}, _id: 0 }).sort({name:1})
[
  { name: 'Dunx', loves: [ 'grape' ] },
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Roooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
```

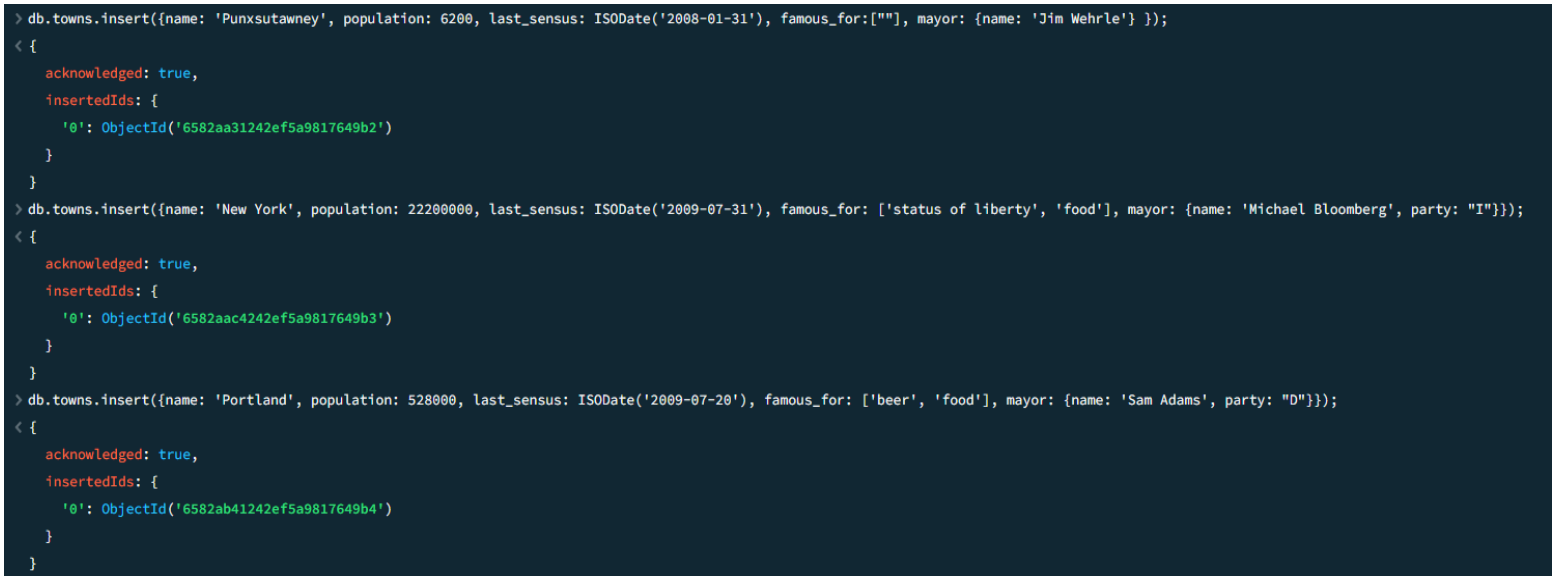
Практическое задание 3.1.1

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"}}
```



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

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

Практическое задание 3.1.2

4) Сформировать функцию для вывода списка самцов единорогов.

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

6) Вывести результат, используя forEach.

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

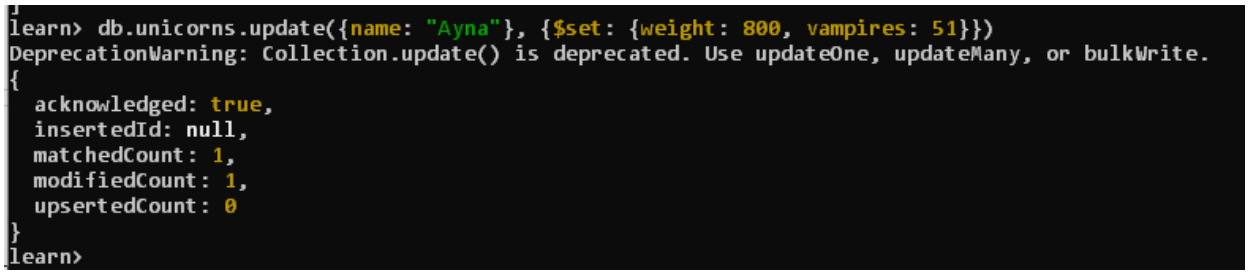
> db.unicorns.save({name: 'Barny', loves: ['grape'],

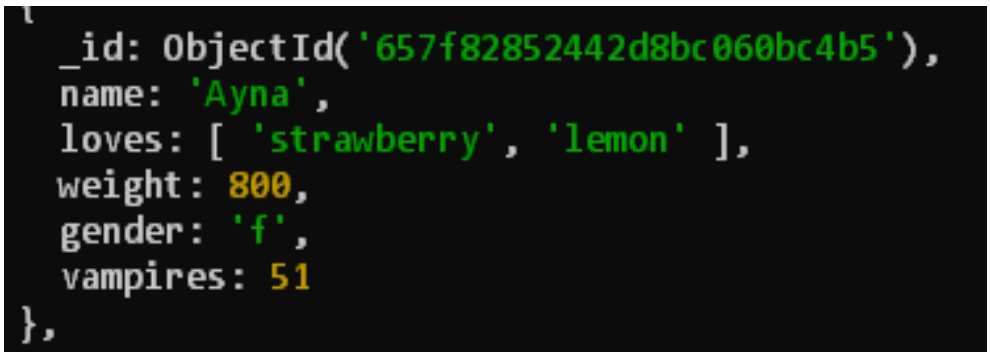
weight: 340, gender: 'm'})

2) Проверить содержимое коллекции unicorns.

Практическое задание 3.3.2

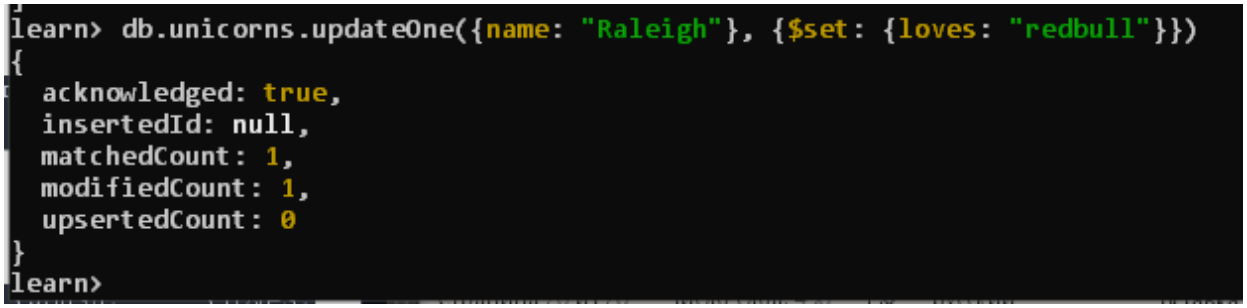
Для самки единорога Ayna внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.





Практическое задание 3.3.3

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



```
},
{
  _id: ObjectId('657f82852442d8bc060bc4b7'),
  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.updateMany({gender: "m"}, {$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn>
```

```
learn> db.unicorns.find({gender: "m"})
[
  {
    _id: ObjectId('657f82842442d8bc060bc4b0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('657f82852442d8bc060bc4b7'),
    name: 'Raleigh',
    loves: 'redbull',
    weight: 421,
    gender: 'm',
    vampires: 7
  },
```

Практическое задание 3.3.5

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

```
learn> db.towns.update({name: "Portland"}, {$set: {"mayor.party": "I"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: "Portland"})
[
  {
    _id: ObjectId('657f98272442d8bc060bc4c5'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'I' }
  }
]
learn>
```

Практическое задание 3.3.6

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

```
learn> db.unicorns.updateOne({name: "Pilot"}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('657f82852442d8bc060bc4b9'),
    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.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('657f82842442d8bc060bc4b1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 3.4.1

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

```
{name: "Punxsutawney ",
popujatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: ["phil the groundhog"],
mayor: {
   name: "Jim Wehrle"
   }}

{name: "New York",
popujatiuon: 22200000,
last_sensus: ISODate("2009-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
   name: "Michael Bloomberg",
   party: "I"}}

{name: "Portland",
popujatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
   party: "D"}}
```

2. Удалите документы с беспартийными мэрами.

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

```
learn> db.towns.deleteMany({ "mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()
[
  {
    _id: ObjectId('657f996e2442d8bc060bc4c7'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('657f996e2442d8bc060bc4c8'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

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

Практическое задание 4.1.1

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

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

```
> db.areas.insertMany([
    {_id: "fr", name: "forest", description: "Unicorn habitat in the forest. Abundant with trees and nature."},
    {_id: "md", name: "meadow", description: "Unicorn habitat in meadows. Green expanses and blooming fields."},
    {_id: "mn", name: "mountains", description: "Unicorn habitat in the mountains. High peaks and crystal-clear air."}
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': 'fr',
      '1': 'md',
      '2': 'mn'
    }
  }
learn>
```

```
> db.unicorns.insertMany([
    {
      "name": "Sparkle",
      "age": 5,
      "habitat": "fr" // Ссылка на зону обитания в лесу
    },
    {
      "name": "Rainbow",
      "age": 3,
      "habitat": "md" // Ссылка на зону обитания на лугах
    },
    {
      "name": "Thunder",
      "age": 7,
      "habitat": "mn" // Ссылка на зону обитания в горах
    }
  ]);
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('65848ae4242ef5a9817649bb'),
      '1': ObjectId('65848ae4242ef5a9817649bc'),
      '2': ObjectId('65848ae4242ef5a9817649bd')
    }
  }
learn>
```

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`.

```
learn> 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.*

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

3. *Попытайтесь удалить индекс для идентификатора.*

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

Практическое задание 4.4.1

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

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

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



{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657cdcc68743a0e00b3651b5') }
}
```

2. *Выберите последних четыре документа.*

```
learn> 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`*)*

```
  nReturned: 4,
  executionTimeMillis: 142,
  totalKeysExamined: 0,
```

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

```
learn> db.numbers.createIndex({value: 1})
value_1
learn>
```

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' }
]
```

6. *Выполните запрос 2.*

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

```
  nReturned: 4,
  executionTimeMillis: 20,
  totalKeysExamined: 4,
```

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

Время выполнения без индекса: 91 мс

Время выполнения с индексом: 4 мс

Время выполнения с индексом превосходит в 22.5 раза.

ДОП ЗАДАНИЕ:

1) В чем отличие DBref от FK?
DBRef: Использует специальный тип данных DBRef, который представляет собой объект с полями $ref (имя коллекции), $id (идентификатор документа), и $db (необязательное поле с именем базы данных).
FK: Использует обычное поле, содержащее идентификатор связанного документа.

2) Какими способами можно добавить данные (одну запись или несколько) в массив. Чем они отличаются? Приведите два способа.
2 способа: Оператор $push и Оператор $addToSet.
Отличие: Если вам нужно добавить значение в массив вне зависимости от того, есть ли оно уже в массиве или нет, используйте $push. Если вам нужно удостовериться, что значение уникально в массиве, используйте $addToSet.
Пример:

```
db.students.update(
  { "_id": ObjectId("идентификатор_документа") },
  { $push: { "grades": 95 } }
);
```

Пример:

```
db.students.update(
  { "_id": ObjectId("идентификатор_документа") },
  { $addToSet: { "courses": "Math" } }
);
```

3) Создайте новую базу данных MongoDB с именем "task9db". В этой базе данных создайте коллекцию "students". Добавьте информацию про нескольких студентов: ФИО и средний балл.

```
> use task9db;
< switched to db task9db
> db.createCollection("students");
< { ok: 1 }
> db.students.insertMany([
  {"name": "Иванов Иван Иванович", "average_score": 4.5},
  {"name": "Петров Петр Петрович", "average_score": 3.8},
  {"name": "Ильясов Ильяс Ильясович", "average_score": 4.2},])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('65847272242ef5a9817649b8'),
      '1': ObjectId('65847272242ef5a9817649b9'),
      '2': ObjectId('65847272242ef5a9817649ba')
    }
  }
```

4) Напишите JavaScript-функцию, которая выполнит выборку всех студентов, у которых средний балл выше определенного значения (например, 4.0). Значение должно передаваться как параметр. Сохраните эту функцию и продемонстрируйте её работоспособность.

```
> function findStudentsAboveAverage(averageScore) {
  var cursor = db.students.find({ "average_score": { $gt: averageScore} });
  cursor.forEach(function(student){
  print(student.name + " - " + student.average_score);
  });
  }
< [Function: findStudentsAboveAverage]
> findStudentsAboveAverage(4.0);
< Иванов Иван Иванович - 4.5
< Ильясов Ильяс Ильясович - 4.2
task9db >
```

Вывод

В процессе выполнения данной практической работы были освоены конкретные методы взаимодействия с базой данных в MongoDB, такие как создание, чтение, обновление и удаление данных1. Были изучены фундаментальные команды, позволяющие управлять информацией в коллекциях. В рамках лаборатории были также ознакомлены с использованием вложенных объектов в структуре коллекций, проведением агрегаций данных и методами изменения данных с применением ссылок и индексов. Суммируя опыт работы, можно заключить, что данная лабораторная работа дала уверенность в применении ключевых навыков для взаимодействия с базой данных в MongoDB, а также обогатила практическим опытом выполнения разнообразных операций с данными.