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: Гусев Н.С.

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:

```
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
'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.u{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0aed') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
2});
db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vam{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0aee') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
imue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0aef') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af0') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af1') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af2') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af3') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af4') }
}
```

```
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af5') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af6') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658fc26e4fdda97e04de0af7') }
}
```

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('658fc3214fdda97e04de0af8') }
}
```

4) Проверьте содержимое коллекции с помощью метода find.

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}).limit(3)
[
  {
    _id: ObjectId('658fc3214fdda97e04de0af8'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aed'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Список самок:

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('658fc26e4fdda97e04de0aee'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af2'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af5'),
    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.

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId('658fc26e4fdda97e04de0aee'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId('658fc26e4fdda97e04de0aee'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 2.2.2:

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

```
learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('658fc3214fdda97e04de0af8'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aed'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af3'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  }
]
```

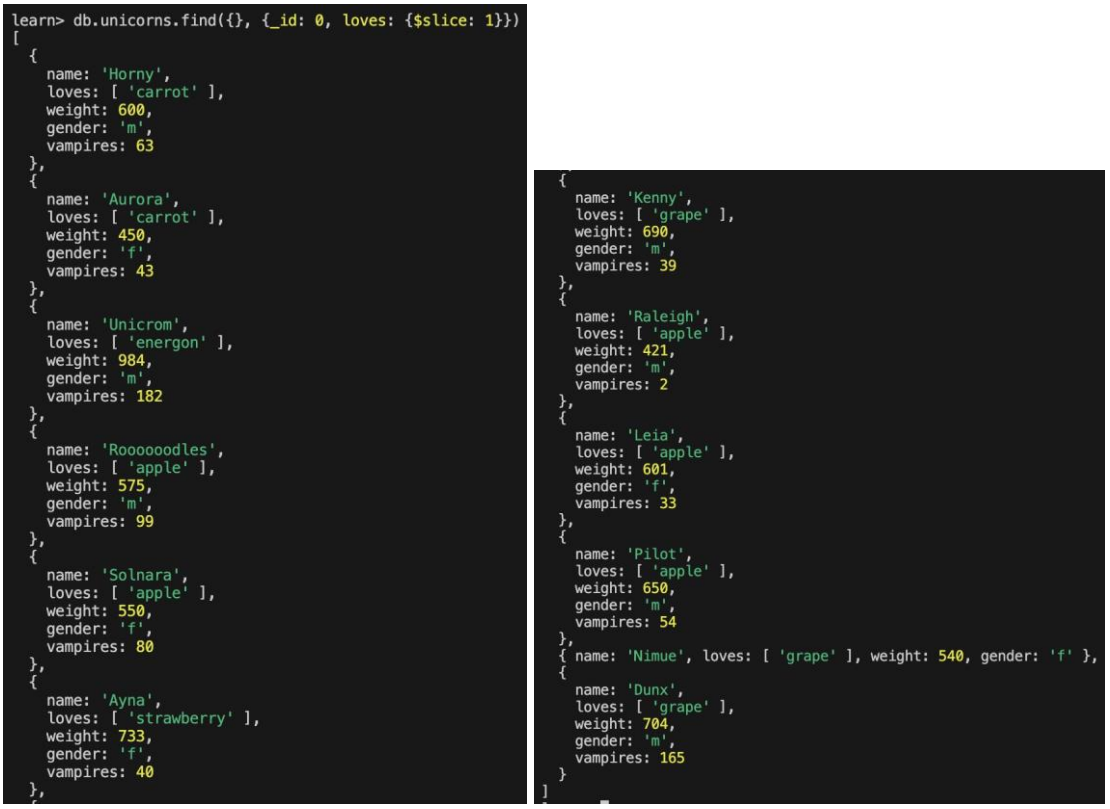
Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId('658fc3214fdda97e04de0af8'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af7'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af5'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af4'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af3'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af2'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af1'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aef'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aee'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aed'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
```

Практическое задание 2.1.4

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



```
learn> db.unicorns.find({}, {_id: 0, loves: {$slice: 1}})
[
  {
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Aurora',
    loves: [ 'carrot' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Ayna',
    loves: [ 'strawberry' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    name: 'Leia',
    loves: [ 'apple' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  { name: 'Nimue', loves: [ 'grape' ], weight: 540, gender: 'f' },
  {
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

Практическое задание 2.3.1

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

```
learn> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 700}}, {_id: 0})
[
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.2

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

```
learn> db.unicorns.find({gender: 'm', weight: {$gte: 500}, loves: {$all: ['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`.

```
learn> db.unicorns.find({vampires: {$exists: 0}})
[
  {
    _id: ObjectId('658fc26e4fdda97e04de0af7'),
    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.

```
learn> 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: '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"}}
```

```
learn> db.towns.insert([
... {name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...   name: "Jim Wehrle"
... }},
... {name: "New York",
-07-20"),
famous_for: ["beer", "food"],
... 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('658fd7b64fdda97e04de0af9'),
    '1': ObjectId('658fd7b64fdda97e04de0afa'),
    '2': ObjectId('658fd7b64fdda97e04de0afb')
  }
}
```

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'}, {_id: 0, name: 1, mayor: 1})
[
  {
    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.

```
learn> db.towns.find({'mayor.party': {$exists: 0}}, {_id: 0, name: 1, mayor: 1})
[ { 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> var f = () => db.unicorns.find({gender: 'm'})

learn> var cursor = f().sort({name: 1}).limit(2)

learn> cursor.forEach((i) => print(i.name))
Dunx
Horny
```

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2

Вывести список предпочтений.

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

Практическое задание 3.2.3

Посчитать количество особей единорогов обоих полов

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

Практическое задание 3.3.1

1) Выполнить команду:

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

weight: 340, gender: 'm'})

.save() has been deprecated

TypeError: db.unicorns.save is not a function

```
learn> db.unicorns.insert({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658ff3dc4fdda97e04de0afc') }
}
```

2) Проверить содержимое коллекции unicorns.

```
  },
  {
    _id: ObjectId('658ff3dc4fdda97e04de0afc'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
```

Практическое задание 3.3.2

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

DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.

```
learn> db.unicorns.updateOne({name: 'Ayna', gender: 'f'}, {$set: {weight: 800, vampires: 51}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
{
  _id: ObjectId('658fc26e4fdda97e04de0af2'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
```

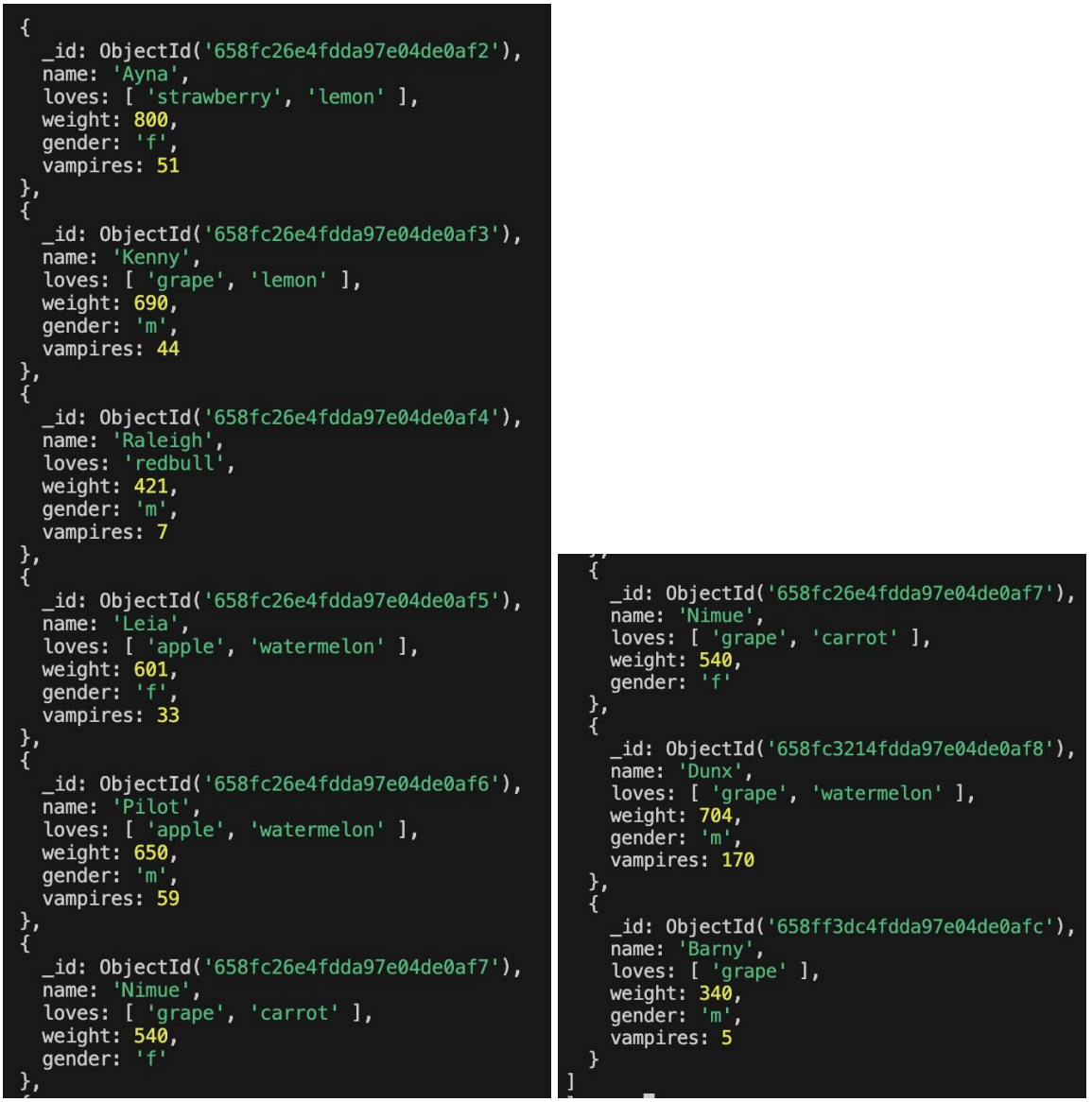
Практическое задание 3.3.3

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

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> db.unicorns.find()
[
  {
    _id: ObjectId('658fc26e4fdda97e04de0aed'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aee'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0aef'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af0'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('658fc26e4fdda97e04de0af1'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
```



Практическое задание 3.3.5

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

```
learn> db.towns.updateOne({name: 'Portland'}, {$unset: {'mayor.party': 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find()
[
  {
    _id: ObjectId('658fd7b64fdda97e04de0af9'),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('658fd7b64fdda97e04de0afa'),
    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('658fd7b64fdda97e04de0afb'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('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.

```
learn> db.unicorns.updateOne({name: 'Pilot', gender: 'm'}, {$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId('658fc26e4fdda97e04de0af6'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

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', 'lemon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId('658fc26e4fdda97e04de0aee'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    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.*

DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.

```
learn> db.towns.insert([{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"}},{n
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('6590161e4fdda97e04de0b00'),
    '1': ObjectId('6590161e4fdda97e04de0b01'),
    '2': ObjectId('6590161e4fdda97e04de0b02')
  }
}
learn> db.towns.deleteMany({'mayor.party': {$exists: 0}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()
[
  {
    _id: ObjectId('6590161e4fdda97e04de0b01'),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('6590161e4fdda97e04de0b02'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
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:

```
learn> db.areas.insert([
... {_id: 'ru', name: 'Russia', description: 'Russia is a country spanning Eastern Europe and Northern Asia. It is the largest country in the world by area, extends acros
s eleven time zones, and shares land boundaries with fourteen countries. It is the world\'s ninth-most populous country and Europe\'s most populous country. The country\'
s capital and largest city is Moscow. Saint Petersburg is Russia\'s second-largest city and cultural capital. Other major urban areas in the country include Novosibirsk,
Yekaterinburg, Nizhny Novgorod, Chelyabinsk, Krasnoyarsk, and Kazan.'},
... {_id: 'ua', name: 'Ukraine', description: 'Ukraine is a country in Eastern Europe. It is the second-largest European country after Russia, which borders it to the eas
t and northeast. It also borders Belarus to the north; Poland, Slovakia, and Hungary to the west; and Romania and Moldova to the southwest; with a coastline along the Bla
ck Sea and the Sea of Azov to the south and southeast. Kyiv is the nation\'s capital and largest city, followed by Kharkiv, Dnipro and Odesa. Ukraine\'s official language
 is Ukrainian; Russian is also widely spoken, especially in the east and south.'},
... {_id: 'by', name: 'Belarus', description: 'Belarus, officially the Republic of Belarus, is a landlocked country in Eastern Europe. It is bordered by Russia to the eas
t and northeast, Ukraine to the south, Poland to the west, and Lithuania and Latvia to the northwest. Covering an area of 207,600 square kilometres (80,200 sq mi) and wit
h a population of 9.2 million, Belarus is the 13th-largest and the 20th-most populous country in Europe. The country has a hemiboreal climate and is administratively divi
ded into six regions. Minsk is the capital and largest city; it is administered separately as a city with special status.'}
... ])
{
  acknowledged: true,
  insertedIds: { '0': 'ru', '1': 'ua', '2': 'by' }
}
```

```
learn> db.unicorns.update({_id: ObjectId('658fc26e4fdda97e04de0aed')}, {$set: {area: {$ref: 'areas', $id: 'ru'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({_id: ObjectId('658fc26e4fdda97e04de0aee')}, {$set: {area: {$ref: 'areas', $id: 'ua'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({_id: ObjectId('658fc26e4fdda97e04de0aef')}, {$set: {area: {$ref: 'areas', $id: 'by'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

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' ]
```

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.dropIndex('name_1')
{ nIndexesWas: 2, ok: 1 }
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})}
```

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?*

Я изначально использовал

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

Но видимо это слишком умно, остальное задание теряет смысл

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

```
},
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 0,
  totalKeysExamined: 0,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'limit',
```

```
}
learn> db.numbers.find().sort({$natural: -1}).limit(4)
[
  { _id: ObjectId('659034394fdda97e04df91ba'), value: 99999 },
  { _id: ObjectId('659034394fdda97e04df91b9'), value: 99998 },
  { _id: ObjectId('659034394fdda97e04df91b8'), value: 99997 },
  { _id: ObjectId('659034394fdda97e04df91b7'), value: 99996 }
]
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('659034394fdda97e04df91ba'), value: 99999 },
  { _id: ObjectId('659034394fdda97e04df91b9'), value: 99998 },
  { _id: ObjectId('659034394fdda97e04df91b8'), value: 99997 },
  { _id: ObjectId('659034394fdda97e04df91b7'), value: 99996 }
]
```

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

```
},
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 93,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
  executionStages: {
    stage: 'sort',
```

```
}
learn> db.numbers.ensureIndex({'value' : 1}, {'unique' : true})
[ 'value_1' ]
learn> db.numbers.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { value: 1 }, name: 'value_1', unique: true }
]
learn> db.numbers.explain('executionStats').find().sort({value: -1}).limit(4)
{
  explainVersion: '2',
  queryPlanner: {
```

```
},
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 9,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'limit',
```

Время выполнения без индекса: 93 мс

Время выполнения с индексом: 9 мс

С индексом в 10 раз быстрее

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.