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»

Обучающийся Малахов Алексей Витальевич
Факультет прикладной информатики
Группа K3239
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2025/2026

1. **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.
2. **Практическое задание:**

   Выполнить задания с 2.1.1 по 4.4.1

3. **Выполнение:**

Практическое задание 2.1.1:

Вставка документов в коллекцию

```
admin> use learn
switched to db learn
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('683337125e9af1f81e65d0fc') }
}
learn> 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('6833372c5e9af1f81e65d106') }
}
```

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

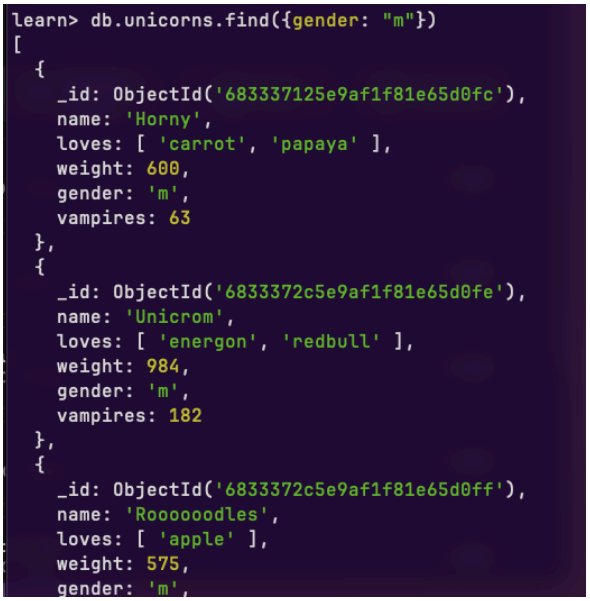
```
learn> db.unicorns.insert(document)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('683337cd5e9af1f81e65d108') }
}
```

Проверка:

```
learn> db.unicorns.find().forEach(doc => print(JSON.stringify(doc)))
{"_id":"683337125e9af1f81e65d0fc","name":"Horny","loves":["carrot","papaya"],"weight":600,"gender":"m","vampires":63}
{"_id":"6833372c5e9af1f81e65d0fd","name":"Aurora","loves":["carrot","grape"],"weight":450,"gender":"f","vampires":43}
{"_id":"6833372c5e9af1f81e65d0fe","name":"Unicrom","loves":["energon","redbull"],"weight":984,"gender":"m","vampires":182}
{"_id":"6833372c5e9af1f81e65d0ff","name":"Roooooodles","loves":["apple"],"weight":575,"gender":"m","vampires":99}
{"_id":"6833372c5e9af1f81e65d100","name":"Solnara","loves":["apple","carrot","chocolate"],"weight":550,"gender":"f","vampire
s":80}
{"_id":"6833372c5e9af1f81e65d101","name":"Ayna","loves":["strawberry","lemon"],"weight":733,"gender":"f","vampires":40}
{"_id":"6833372c5e9af1f81e65d102","name":"Kenny","loves":["grape","lemon"],"weight":690,"gender":"m","vampires":39}
{"_id":"6833372c5e9af1f81e65d103","name":"Raleigh","loves":["apple","sugar"],"weight":421,"gender":"m","vampires":2}
{"_id":"6833372c5e9af1f81e65d104","name":"Leia","loves":["apple","watermelon"],"weight":601,"gender":"f","vampires":33}
{"_id":"6833372c5e9af1f81e65d105","name":"Pilot","loves":["apple","watermelon"],"weight":650,"gender":"m","vampires":54}
{"_id":"6833372c5e9af1f81e65d106","name":"Nimue","loves":["grape","carrot"],"weight":540,"gender":"f"}
{"_id":"683337cd5e9af1f81e65d108","name":"Dunx","loves":["grape","watermelon"],"weight":704,"gender":"m","vampires":165}
```

Практическое задание 2.2.1:

Выводим только самцов:



```
learn> db.unicorns.find({gender: "m"})
[
  {
    _id: ObjectId('683337125e9af1f81e65d0fc'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d0fe'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d0ff'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
```

Выводим 3-ех самок:

```
learn> db.unicorns.find({gender: "f"}).limit(3)
[
  {
    _id: ObjectId('6833372c5e9af1f81e65d0fd'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d100'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d101'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
]
```

Сортировка по имени:

```
learn> db.unicorns.find({gender: "f"}, {name: 1}).limit(3).sort({name: 1})
[
  { _id: ObjectId('6833372c5e9af1f81e65d0fd'), name: 'Aurora' },
  { _id: ObjectId('6833372c5e9af1f81e65d101'), name: 'Ayna' },
  { _id: ObjectId('6833372c5e9af1f81e65d104'), name: 'Leia' }
]
```

Только первая самка, любящая carrot:

```
learn> db.unicorns.findOne({gender: "f", loves: "carrot"})
{
  _id: ObjectId('6833372c5e9af1f81e65d0fd'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn> db.unicorns.find({gender: "f", loves: "carrot"}).limit(1)
[
  {
    _id: ObjectId('6833372c5e9af1f81e65d0fd'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

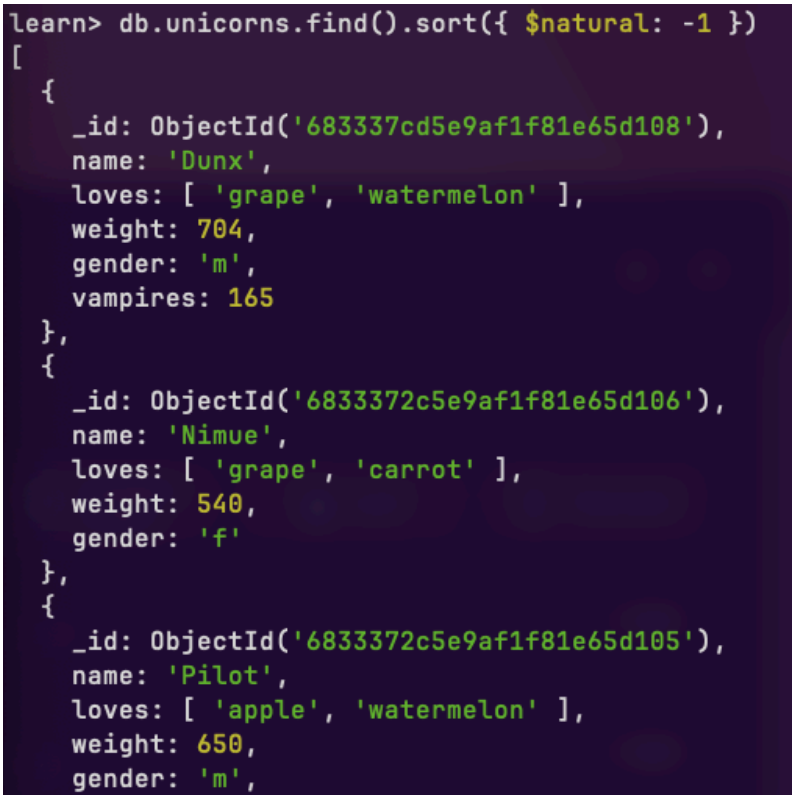
Практическое задание 2.2.2:

Вывод самцов исключив из результата информацию о предпочтениях и поле:

```
learn> db.unicorns.find({gender: "m"}, {loves: 0, gender:0})
[
  {
    _id: ObjectId('683337125e9af1f81e65d0fc'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d0fe'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d0ff'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
```

Практическое задание 2.2.3:

Список единорогов в обратном порядке добавления:



```
learn> db.unicorns.find().sort({ $natural: -1 })
[
  {
    _id: ObjectId('683337cd5e9af1f81e65d108'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d106'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d105'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
```

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
  },
```

Практическое задание 2.3.1:
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:
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:
Поиск всех единорогов, не имеющих ключ vampires:

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId('6833372c5e9af1f81e65d106'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.4:
Упорядоченный список имен самцов единорогов с информацией об их первом предпочтении:

```
learn> db.unicorns.find({gender: 'm'}, {name: -1, loves: {$slice: 1}}).sort({name: 1})
[
  {
    _id: ObjectId('683337cd5e9af1f81e65d108'),
    name: 'Dunx',
    loves: [ 'grape' ]
  },
  {
    _id: ObjectId('683337125e9af1f81e65d0fc'),
    name: 'Horny',
    loves: [ 'carrot' ]
  },
  {
    _id: ObjectId('6833372c5e9af1f81e65d102'),
    name: 'Kenny',
    loves: [ 'grape' ]
  },
```

Практическое задание 3.1.1:
Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре.

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

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

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

Практическое задание 3.1.2:

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

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

Практическое задание 3.2.1:

Количество самок единорогов весом от полутонны до 600 кг:

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

Практическое задание 3.2.2:

Список предпочтений:

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

Практическое задание 3.2.3:

Количество особей единорогов обоих полов:

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

Практическое задание 3.3.1:

Выполнить команду save и проверить результат:

Метод save не поддерживается в последней версии mongodb shell, используем updateOne:

```
learn> db.unicorns.updateOne({ name: "Barny" }, { $set: { name: "Barny", loves: ["grape"], weight: 340, gender: "m" } }, { u
psert: true })
{
  acknowledged: true,
  insertedId: ObjectId('68335229c144186454ceccd1'),
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 1
}
learn> db.unicorns.find().forEach(function(doc) { print(JSON.stringify(doc)) })
{"_id":"683337125e9af1f81e65d0fc","name":"Horny","loves":["carrot","papaya"],"weight":600,"gender":"m","vampires":63}
{"_id":"6833372c5e9af1f81e65d0fd","name":"Aurora","loves":["carrot","grape"],"weight":450,"gender":"f","vampires":43}
{"_id":"6833372c5e9af1f81e65d0fe","name":"Unicrom","loves":["energon","redbull"],"weight":984,"gender":"m","vampires":182}
{"_id":"6833372c5e9af1f81e65d0ff","name":"Roooooodles","loves":["apple"],"weight":575,"gender":"m","vampires":99}
{"_id":"6833372c5e9af1f81e65d100","name":"Solnara","loves":["apple","carrot","chocolate"],"weight":550,"gender":"f","vampire
s":80}
{"_id":"6833372c5e9af1f81e65d101","name":"Ayna","loves":["strawberry","lemon"],"weight":733,"gender":"f","vampires":40}
{"_id":"6833372c5e9af1f81e65d102","name":"Kenny","loves":["grape","lemon"],"weight":690,"gender":"m","vampires":39}
{"_id":"6833372c5e9af1f81e65d103","name":"Raleigh","loves":["apple","sugar"],"weight":421,"gender":"m","vampires":2}
{"_id":"6833372c5e9af1f81e65d104","name":"Leia","loves":["apple","watermelon"],"weight":601,"gender":"f","vampires":33}
{"_id":"6833372c5e9af1f81e65d105","name":"Pilot","loves":["apple","watermelon"],"weight":650,"gender":"m","vampires":54}
{"_id":"6833372c5e9af1f81e65d106","name":"Nimue","loves":["grape","carrot"],"weight":540,"gender":"f"}
{"_id":"683337cd5e9af1f81e65d108","name":"Dunx","loves":["grape","watermelon"],"weight":704,"gender":"m","vampires":165}
{"_id":"68335229c144186454ceccd1","name":"Barny","gender":"m","loves":["grape"],"weight":340}
```

upsert: true — если нет такого документа, вставить его

Практическое задание 3.3.2:

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

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

learn> db.unicorns.find({name: "Ayna"})
[
  {
    _id: ObjectId('6833372c5e9af1f81e65d101'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

Практическое задание 3.3.3:
Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.

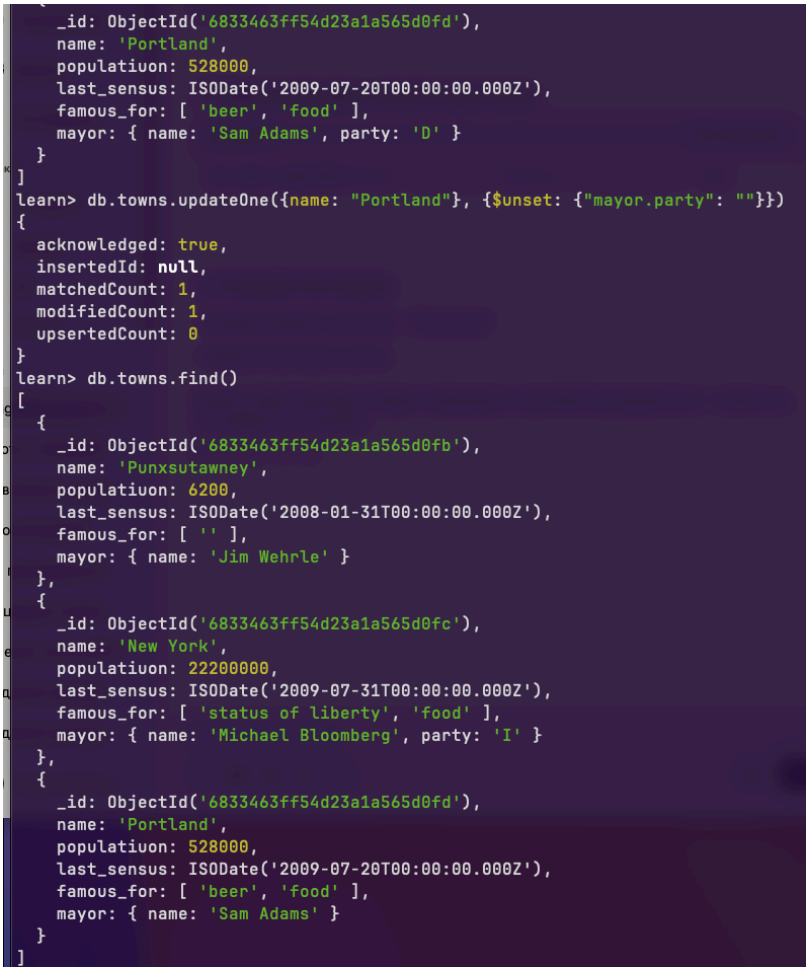
```
learn> db.unicorns.updateOne({name: "Raleigh"}, {$addToSet: {loves: "redbull"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Raleigh"})
[
  {
    _id: ObjectId('6833372c5e9af1f81e65d103'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

Практическое задание 3.3.4:
Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
learn> db.unicorns.find({gender: "m"}).forEach(function(doc) { print(JSON.stringify(doc)) })
{"_id":"683337125e9af1f81e65d0fc","name":"Horny","loves":["carrot","papaya"],"weight":600,"gender":"m","vampires":63}
{"_id":"6833372c5e9af1f81e65d0fe","name":"Unicrom","loves":["energon","redbull"],"weight":984,"gender":"m","vampires":182}
{"_id":"6833372c5e9af1f81e65d0ff","name":"Roooooodles","loves":["apple"],"weight":575,"gender":"m","vampires":99}
{"_id":"6833372c5e9af1f81e65d102","name":"Kenny","loves":["grape","lemon"],"weight":690,"gender":"m","vampires":39}
{"_id":"6833372c5e9af1f81e65d103","name":"Raleigh","loves":["apple","sugar","redbull"],"weight":421,"gender":"m","vampires":
2}
{"_id":"6833372c5e9af1f81e65d105","name":"Pilot","loves":["apple","watermelon"],"weight":650,"gender":"m","vampires":54}
{"_id":"683337cd5e9af1f81e65d108","name":"Dunx","loves":["grape","watermelon"],"weight":704,"gender":"m","vampires":165}
{"_id":"68335229c144186454ceccd1","name":"Barny","gender":"m","loves":["grape"],"weight":340}

learn> db.unicorns.updateMany({gender: "m"}, {$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: "m"}).forEach(function(doc) { print(JSON.stringify(doc)) })
{"_id":"683337125e9af1f81e65d0fc","name":"Horny","loves":["carrot","papaya"],"weight":600,"gender":"m","vampires":68}
{"_id":"6833372c5e9af1f81e65d0fe","name":"Unicrom","loves":["energon","redbull"],"weight":984,"gender":"m","vampires":187}
{"_id":"6833372c5e9af1f81e65d0ff","name":"Roooooodles","loves":["apple"],"weight":575,"gender":"m","vampires":104}
{"_id":"6833372c5e9af1f81e65d102","name":"Kenny","loves":["grape","lemon"],"weight":690,"gender":"m","vampires":44}
{"_id":"6833372c5e9af1f81e65d103","name":"Raleigh","loves":["apple","sugar","redbull"],"weight":421,"gender":"m","vampires":
7}
{"_id":"6833372c5e9af1f81e65d105","name":"Pilot","loves":["apple","watermelon"],"weight":650,"gender":"m","vampires":59}
{"_id":"683337cd5e9af1f81e65d108","name":"Dunx","loves":["grape","watermelon"],"weight":704,"gender":"m","vampires":170}
{"_id":"68335229c144186454ceccd1","name":"Barny","gender":"m","loves":["grape"],"weight":340,"vampires":5}
```

Практическое задание 3.3.5:
Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.



```
    _id: ObjectId('6833463ff54d23a1a565d0fd'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.updateOne({name: "Portland"}, {$unset: {"mayor.party": ""}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find()
[
  {
    _id: ObjectId('6833463ff54d23a1a565d0fb'),
    name: 'Punxsutawney',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('6833463ff54d23a1a565d0fc'),
    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('6833463ff54d23a1a565d0fd'),
    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"}, {$addToSet: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('6833372c5e9af1f81e65d105'),
    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('6833372c5e9af1f81e65d0fd'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 3.4.1:
Удалить документы с беспартийными мэрами
Метод remove больше не поддерживается, используем deleteMany:

```
learn> db.towns.deleteMany({"mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()
[
  {
    _id: ObjectId('683355d3bb4ab8b73f65d0fc'),
    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('683355d3bb4ab8b73f65d0fd'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

Очищаем коллекцию через deleteMany без аргументов:

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

learn> show collections
towns
unicorns
```

Практическое задание 4.1.1:

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

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

```
learn> db.habitats.insertMany([{_id: "NF", name: "Northern Forest", description: "Dense forest in the north, rich
 in berries and mushrooms."}, {_id: "DS", name: "Desert Sands", description: "Hot, dry desert area with occasiona
l oases."}, {_id: "CR", name: "Crystal Ridge", description: "Mountainous region glittering with crystal formation
s."}])
{
  acknowledged: true,
  insertedIds: { '0': 'NF', '1': 'DS', '2': 'CR' }
}
learn> db.unicorns.updateOne({name: "Aurora"}, {$set: {habitat: {$ref: "habitats", $id: "NF"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: "Unicrom"}, {$set: {habitat: {$ref: "habitats", $id: "DS"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: "Solnara"}, {$set: {habitat: {$ref: "habitats", $id: "CR"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
learn> db.unicorns.find({name: {$in: ["Aurora", "Unicrom", "Solnara"]}})
[
  {
    _id: ObjectId('68335bc6bb4ab8b73f65d0ff'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat: DBRef('habitats', 'NF')
  },
  {
    _id: ObjectId('68335bc6bb4ab8b73f65d100'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182,
    habitat: DBRef('habitats', 'DS')
  },
  {
    _id: ObjectId('68335bc6bb4ab8b73f65d102'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80,
    habitat: DBRef('habitats', 'CR')
  }
]
```

Практическое задание 4.2.1:
Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique.
Метод ensureIndex устарел, используем новый:

```
learn> db.unicorns.createIndex({name: 1}, {unique: true})
name_1
```

Практическое задание 4.3.1:
Получите информацию о всех индексах коллекции unicorns. Удалите все индексы, кроме индекса для идентификатора. Попытайтесь удалить индекс для идентификатора.

```
learn> db.unicorns.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
learn> db.unicorns.dropIndexes()
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
learn> db.unicorns.dropIndex("_id_")
MongoServerError[InvalidOptions]: cannot drop _id index
learn> db.unicorns.getIndexes()
[ { v: 2, key: { _id: 1 }, name: '_id_' } ]
```

Практическое задание 4.4.1:
Заполняем БД:

```
learn> for (i = 0; i < 100000; i++) { db.numbers.insertOne({value: i}) }
{
  acknowledged: true,
  insertedId: ObjectId('68338f13bb4ab8b73f691be3')
}
```

Скорость выполнения запроса на последние 4 документа:

```
learn> db.numbers.find().sort({value: -1}).limit(4).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BA27D965',
    planCacheShapeHash: 'BA27D965',
    planCacheKey: '7A892B81',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'SORT',
      sortPattern: { value: -1 },
      memLimit: 104857600,
      limitAmount: 4,
      type: 'simple',
      inputStage: { stage: 'COLLSCAN', direction: 'forward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 39,
```

Все индексы коллекции:

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

Тот же запрос но с индексами:

```
learn> db.numbers.find().sort({value: -1}).limit(4).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BA27D965',
    planCacheShapeHash: 'BA27D965',
    planCacheKey: '7A892B81',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: {
        stage: 'FETCH',
        inputStage: {
          stage: 'IXSCAN',
          keyPattern: { value: 1 },
          indexName: 'value_1',
          isMultiKey: false,
          multiKeyPaths: { value: [] },
          isUnique: false,
          isSparse: false,
          isPartial: false,
          indexVersion: 2,
          direction: 'backward',
          indexBounds: { value: [ '[MaxKey, MinKey]' ] }
        }
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 1,
```

Очевидно, что более эффективен запрос с включенными индексами.

Выводы: В ходе лабораторной работы я получил практические навыки работы с CRUD-операциями, вложенными объектами, агрегацией, изменением данных, ссылками между коллекциями и индексами в базе данных MongoDB.