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 Милютин Никита Александровчич
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

```
test> use learn
switched to db learn
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
... db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
... db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
... db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
... db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
... db.unicorns.insert({name:'Ayna', loves: [ 'strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
... db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
... db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
... db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
... db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
... db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
...
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6834b5b22368bfcc8dd861ea') }
}
learn>
```

```
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('6834b7062368bfcc8dd861eb') }
}
learn>
```

Проверка:

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  }
```

Практическое задание 2.2.1:

Выводим только самцов:

```
learn> db.unicorns.find({"gender":"m"})
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e6'),
```

Выводим 3-ех самок:

```
learn> db.unicorns.find({"gender":"f"}).limit(3)
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e4'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
]
```

Сортировка по имени:

```
switched to db learn
learn> db.unicorns.find({"gender":"f"}, {"name":1}).limit(3).sort({"name":1})
[
  { _id: ObjectId('6834b5b22368bfcc8dd861e1'), name: 'Aurora' },
  { _id: ObjectId('6834b5b22368bfcc8dd861e5'), name: 'Ayna' },
  { _id: ObjectId('6834b5b22368bfcc8dd861e8'), name: 'Leia' }
]
```

Только первая самка, любящая carrot:

```
learn> db.unicorns.findOne({gender:"f", "loves":"carrot"})
{
  _id: ObjectId('6834b5b22368bfcc8dd861e1'),
  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('6834b5b22368bfcc8dd861e0'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e2'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e3'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e6'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e7'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e9'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('6834b7062368bfcc8dd861eb'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
```

Практическое задание 2.2.3:

Список единорогов в обратном порядке добавления:

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId('6834b7062368bfcc8dd861eb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861ea'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e8'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
```

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',
```

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({vimpires:{$exists: false}})
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
```

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

```
learn> db.unicorns.find({gender: "m"}, {loves:{$slice:1}, name: 1}).sort({name:1})
[
  {
    _id: ObjectId('6834b7062368bfcc8dd861eb'),
    name: 'Dunx',
    loves: [ 'grape' ]
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e0'),
    name: 'Horny',
    loves: [ 'carrot' ]
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e6'),
    name: 'Kenny',
    loves: [ 'grape' ]
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e9'),
    name: 'Pilot',
    loves: [ 'apple' ]
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e7'),
    name: 'Raleigh',
    loves: [ 'apple' ]
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e3'),
    name: 'Roooooodles',
    loves: [ 'apple' ]
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e2'),
    name: 'Unicrom',
    loves: [ 'energon' ]
  }
]
```

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

learn>
```

Практическое задание 3.2.1:

Количество самок единорогов весом от полутонны до 600 кг:

```
learn> db.unicorns.find({gender: "m", weight: {$gte: 500, $lte: 600}}).count()
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: 'm', count: 7 }, { _id: 'f', count: 5 } ]
```

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: {
...       loves: ['grape'],
...       weight: 340,
...       gender: 'm'
...     }
...   },
...   { upsert: true } // опционально: создаст докуме
... )
{
  acknowledged: true,
  insertedId: ObjectId('68363aa53d91682948f1c880'),
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 1
}
```

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
}
```

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

```
learn> db.unicorns.updateOne({name: "Raleight", gender:"m"}, {$set:{loves:[ "redbule"]}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:"Raleight"})
[
  {
    _id: ObjectId('68363f12f1bbb85eefd861e0'),
    name: 'Raleight',
    gender: 'm',
    loves: [ 'redbule' ]
  }
]
```

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: 9,
  modifiedCount: 9,
  upsertedCount: 0
}
learn> db.unicorns.find({gender:"m"})
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
```

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({name: "Portland"})
[
  {
    _id: ObjectId('68362a6d976bd1873ad861e2'),
    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: "choco"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilott", gender: "m"})

learn> db.unicorns.find({name: "Pilot", gender: "m"})
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'choco' ],
    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", gender: "f"}, {$addToSet: {"loves": {$each:["sugar", "lemons"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Aurora", gender: "f"})
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    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: 0}})
{ acknowledged: true, deletedCount: 2 }
learn> db.towns.find()
[
  {
    _id: ObjectId('68362a6d976bd1873ad861e1'),
    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' }
  }
]
learn> db.towns.deleteMany({})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()

learn>
```

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: "FR",
...         name: "Enchanted Forest",
...         description: "A magical forest where unicorns roam freely among ancient trees and glowing flowers"
...     },
...     {
...         _id: "MT",
...         name: "Crystal Mountains",
...         description: "High mountain peaks with crystal caves where unicorns take refuge and graze on magical herbs"
...     },
...     {
...         _id: "VL",
...         name: "Rainbow Valley",
...         description: "A hidden valley filled with ever-changing rainbows where unicorns come to drink from magical springs"
...     }
... ])
{
  acknowledged: true,
  insertedIds: { '0': 'FR', '1': 'MT', '2': 'VL' }
}
learn> db.habitats.find()
[
  {
    _id: 'FR',
    name: 'Enchanted Forest',
    description: 'A magical forest where unicorns roam freely among ancient trees and glowing flowers'
  },
  {
    _id: 'MT',
    name: 'Crystal Mountains',
    description: 'High mountain peaks with crystal caves where unicorns take refuge and graze on magical herbs'
  },
  {
    _id: 'VL',
    name: 'Rainbow Valley',
    description: 'A hidden valley filled with ever-changing rainbows where unicorns come to drink from magical springs'
  }
]
```

```
learn> db.unicorns.updateOne({name: "Aurora"}, {$set: {habitat: {$ref: "habitats", $id: "FR"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: "Ayna"}, {$set: {habitat: {$ref: "habitats", $id: "MT"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: "Kenny"}, {$set: {habitat: {$ref: "habitats", $id: "VL"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:{$in:["Aurora","Ayna","Kenny"]}})
[
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat: DBRef('habitats', 'FR')
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51,
    habitat: DBRef('habitats', 'MT')
  },
  {
    _id: ObjectId('6834b5b22368bfcc8dd861e6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44,
    habitat: DBRef('habitats', 'VL')
  }
]
```

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.ensureIndex({name: 1}, {unique: 1})
MongoServerError[DuplicateKey]: Index build failed: 7e4d9eb2-5cd7-45f4-9c73-75a9658f0f19: Collection learn.unicorns ( c5ac915c-0995-48e0-af03-5f08b5e86bd4 ) :: caused by :: E11000 duplicate key error collection: learn.unicorns index: name
 1 dup key: { name: "Roooooodles" }
```

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_' } ]
learn>
```

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

Практическое задание 4.4.1:
Заполняем БД и Скорость выполнения запроса на последние 4 документа:

```
learn> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6836f0954b885139a7da217f') }
}
learn> db.numbers.find().count()
114591
learn> db.numbers.find().sort(-1).limit(4).explain("executionStats")
MongoInvalidArgumentError: Invalid sort format: -1 Sort must be a valid object
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: 101,
    totalKeysExamined: 0,
    totalDocsExamined: 114591,
    executionStages: {
      isCached: false,
      stage: 'SORT',
      nReturned: 4,
      executionTimeMillisEstimate: 90,
      works: 114597,
      advanced: 4,
      needTime: 114592,
      needYield: 0,
      saveState: 5,
      restoreState: 5,
      isEOF: 1,
      sortPattern: { value: -1 },
      memLimit: 104857600,
      limitAmount: 4,
      type: 'simple',
```

Все индексы коллекции:

```
learn> db.numbers.createIndex({value: 1})
value_1
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: 0,
```

Очевидно, что более эффективен запрос с включенными индексами.

Выводы:

В ходе лабораторной работы были приобретены практические навыки работы с MongoDB, включая:

- выполнение CRUD-операций (создание, чтение, обновление и удаление документов);
- работу с вложенными объектами и сложными структурами данных;
- применение агрегационных пайплайнов для обработки и анализа данных;
- изменение документов и управление ссылками между коллекциями;
- создание и использование индексов для оптимизации запросов.

В результате были освоены основные принципы работы с документо ориентированной СУБД, что позволяет эффективно хранить, изменять и извлекать данные в MongoDB.