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»**

Выполнил: Захарчук А. И.

Факультет: ИКТ

Группа: К3239

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).

Оглавление

Выполнение работы

2.1.1

1. Создайте базу данных learn.
2. Заполните коллекцию единорогов unicorns:
3. Используя второй способ, вставьте в коллекцию единорогов документ:
4. Проверьте содержимое коллекции с помощью метода find.

```
test> 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('657742a3725b845e335cd667') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd668') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd669') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd66a') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd66b') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd66c') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd66d') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd66e') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd66f') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd670') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657742a3725b845e335cd671') }
}
learn>
```

```
learn> doc = {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(doc)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6577433f725b845e335cd672') }
}
learn>
```

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('657742a3725b845e335cd667'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657742a3725b845e335cd668'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657742a3725b845e335cd669'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('657742a3725b845e335cd66a'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('657742a3725b845e335cd66b'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('657742a3725b845e335cd66c'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
```

2.2.1

1. Сформируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.
2. Найдите всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций findOne и limit.

```
learn> db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('657742a3725b845e335cd668'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657742a3725b845e335cd66c'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657742a3725b845e335cd66f'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

```
learn> db.unicorns.find({gender: "m"}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('6577433f725b845e335cd672'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657742a3725b845e335cd667'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657742a3725b845e335cd66d'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
learn>
```

```
learn> db.unicorns.findOne({gender: "f", loves: "carrot"})
{
  _id: ObjectId('657742a3725b845e335cd668'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn>
```

```
learn> db.unicorns.find({gender: "f", loves: "carrot"}).limit(1)
[
  {
    _id: ObjectId('657742a3725b845e335cd668'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

2.2.2

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

```
learn> db.unicorns.find({gender: "m"}, {loves: 0}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('6577433f725b845e335cd672'),
    name: 'Dunx',
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657742a3725b845e335cd667'),
    name: 'Horny',
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657742a3725b845e335cd66d'),
    name: 'Kenny',
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
learn>
```

2.2.3

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId('6577433f725b845e335cd672'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657742a3725b845e335cd671'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('657742a3725b845e335cd670'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657742a3725b845e335cd66f'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('657742a3725b845e335cd66e'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('657742a3725b845e335cd66d'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('657742a3725b845e335cd66c'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
```

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
    }
  ]
  learn>
```

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'
  }
]
learn>
```

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
  }
]
learn>
```

2.3.3

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId('657742a3725b845e335cd671'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

2.3.4

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

```
learn> db.unicorns.find({gender: "m"}, {name: 1, _id: 0, 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' ] }
]
learn>
```

3.1.1

1. Создайте коллекцию towns, включающую следующие документы.
2. Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре.
3. Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре.

```
learn> db.towns.insertMany([{name: "Punxsutawney ", populatiuon: 6200, last_s
ensus: 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 Bloombe
rg", party: "I"}}, {name: "Portland", populatiuon: 528000, last_sensus: ISODa
te("2009-07-20"), famous_for: ["beer", "food"], mayor: { name: "Sam Adams", p
arty: "D"}}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('657749fd725b845e335cd673'),
    '1': ObjectId('657749fd725b845e335cd674'),
    '2': ObjectId('657749fd725b845e335cd675')
  }
}
learn>
```

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

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

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> fn = () => gender === "m"
[Function: fn]
learn> db.unicorns.find({$where: fn})
[
  {
    _id: ObjectId('657742a3725b845e335cd667'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657742a3725b845e335cd668'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657742a3725b845e335cd669'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
```

```
learn> var cursor = db.unicorns.find({$where: fn}).sort({name: 1}).limit(2); null;
null
learn>
```

```
learn> cursor.forEach(unicorn => print(unicorn))
{
  _id: ObjectId('657742a3725b845e335cd668'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
{
  _id: ObjectId('657742a3725b845e335cd66c'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
}

learn>
```

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
learn>
```

3.2.2

Вывести список предпочтений.

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

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>
```

3.3.1

1. Выполнить команду:
2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.save(
... {
... name: 'Barny',
... loves: ['grape'],
... weight: 340,
... gender: 'm'
... }
... )
TypeError: db.unicorns.save is not a function
```

Method save is now deprecated and does not work

3.3.2

1. Для самки единорога Ayna внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.
2. Проверить содержимое коллекции unicorns.

```
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"})
[
  {
    _id: ObjectId('657742a3725b845e335cd66c'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
learn>
```

3.3.3

1. Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.
2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.updateOne({name: "Raleigh", gender: "m"}, {$set: {loves: ["redbull"]}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Raleigh"})
[
  {
    _id: ObjectId('657742a3725b845e335cd66e'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

3.3.4

1. Всем самцам единорогов увеличить количество убитых вапмиров на 5.
2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({gender: "m"}, {name: 1, vampires: 1, _id: 0})
[
  { name: 'Horny', vampires: 68 },
  { name: 'Unicrom', vampires: 187 },
  { name: 'Roooooodles', vampires: 104 },
  { name: 'Kenny', vampires: 44 },
  { name: 'Raleigh', vampires: 7 },
  { name: 'Pilot', vampires: 59 },
  { name: 'Dunx', vampires: 170 }
]
learn> db.unicorns.updateMany({gender: "m"}, {$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 7,
  modifiedCount: 7,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: "m"}, {name: 1, vampires: 1, _id: 0})
[
  { name: 'Horny', vampires: 73 },
  { name: 'Unicrom', vampires: 192 },
  { name: 'Roooooodles', vampires: 109 },
  { name: 'Kenny', vampires: 49 },
  { name: 'Raleigh', vampires: 12 },
  { name: 'Pilot', vampires: 64 },
  { name: 'Dunx', vampires: 175 }
]
learn>
```

3.3.5

1. Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.
2. Проверить содержимое коллекции towns.

```
learn> db.towns.find({name: "Portland"})
[
  {
    _id: ObjectId('657749fd725b845e335cd675'),
    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": 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: "Portland"})
[
  {
    _id: ObjectId('657749fd725b845e335cd675'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
learn>
```

3.3.6

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

Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('657742a3725b845e335cd670'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 64
  }
]
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('657742a3725b845e335cd670'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 64
  }
]
learn>
```

3.3.7

Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.

Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId('657742a3725b845e335cd668'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
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('657742a3725b845e335cd668'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

3.4.1

1. Создайте коллекцию towns, включающую следующие документы.
2. Удалите документы с беспартийными мэрами.
3. Проверьте содержание коллекции.
4. Очистите коллекцию.
5. Просмотрите список доступных коллекций.

```
learn> db.towns.insertMany([{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", popujatiu
on: 520000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: { name: "Sam Adams", party: "D"}}]
... )
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('657757f0725b845e335cd676'),
    '1': ObjectId('657757f0725b845e335cd677'),
    '2': ObjectId('657757f0725b845e335cd678')
  }
}
learn>
```

```
learn> db.towns.find({"mayor.party": {$exists: false}})
[
  {
    _id: ObjectId('657757f0725b845e335cd676'),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  }
]
learn> db.towns.deleteMany({"mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find({"mayor.party": {$exists: false}})

learn>
```

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

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.

```
learn> db.zones.insertMany([ { _id: "forest", name: "Enchanted Forest", description: "A magical forest where unicorns roam freely." }, { _id: "meadows", nam
e: "Crystal Meadows", description: "Vast meadows filled with crystal-clear streams, perfect for unicorns." }, { _id: "valley", name: "Starlit Valley", descr
iption: "A serene valley bathed in starlight, the preferred habitat of mystical unicorns." } ]);
{
  acknowledged: true,
  insertedIds: { '0': 'forest', '1': 'meadows', '2': 'valley' }
}
learn>
```

```
learn> db.unicorns.updateOne({_id: ObjectId('6577433f725b845e335cd672')}, {$set: {habitat: {$ref: "zones", $id: "forest"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({_id: ObjectId('657742a3725b845e335cd671')}, {$set: {habitat: {$ref: "zones", $id: "meadow"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({_id: ObjectId('657742a3725b845e335cd670')}, {$set: {habitat: {$ref: "zones", $id: "valley"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn>
```

```
learn> db.unicorns.find({habitat: {$exists: true}})
[
  {
    _id: ObjectId('657742a3725b845e335cd670'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 64,
    habitat: DBRef('zones', 'valley')
  },
  {
    _id: ObjectId('657742a3725b845e335cd671'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f',
    habitat: DBRef('zones', 'meadow')
  },
  {
    _id: ObjectId('6577433f725b845e335cd672'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 175,
    habitat: DBRef('zones', 'forest')
  }
]
learn>
```

4.2.1

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

```
learn> db.unicorns.ensureIndex({name: 1}, {unique: true})
[ 'name_1' ]
learn>
```

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

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?

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

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

При таком запросе без индекса время выполнения 55, с индексом 2.

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

При таком запросе время выполнения 28, с индексом 35.

```
learn> db.numbers.explain("executionStats").find().skip(db.numbers.countDocuments() - 4)
```

Для запроса с сортировкой вариант с индексом работает 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.