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 Федоров Даниил Михайлович
Факультет прикладной информатики
Группа K3240
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2024/2025

1 Цель работы

Овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, ссылками и индексами в базе данных MongoDB.

Практическое задание 2.1.1

1. Создайте базу данных learn.

```
test> use learn
switched to db learn
```

2. Заполните коллекцию единорогов unicorns

```
        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'});
```

```
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('68382e3562a93709e56c4bda') }
}
```

3. Используя второй способ, вставьте в коллекцию единорогов документ:

```
        {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires:
165}
```

```
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('68382efb62a93709e56c4bdb') }
}
```

4. Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('68382e3562a93709e56c4bd0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd4'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
```

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

learn> db.unicorns.find({gender: 'm'}).sort({name: 1})

learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)

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

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId('68382e3562a93709e56c4bd1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId('68382e3562a93709e56c4bd1'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn>
```

learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})

Практическое задание 2.2.2

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

```
learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0}).sort({name:1})
[
  {
    _id: ObjectId('68382efb62a93709e56c4bdb'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd0'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd6'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd9'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd7'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd3'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd2'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
]
learn>
```

learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0}).sort({name:1})

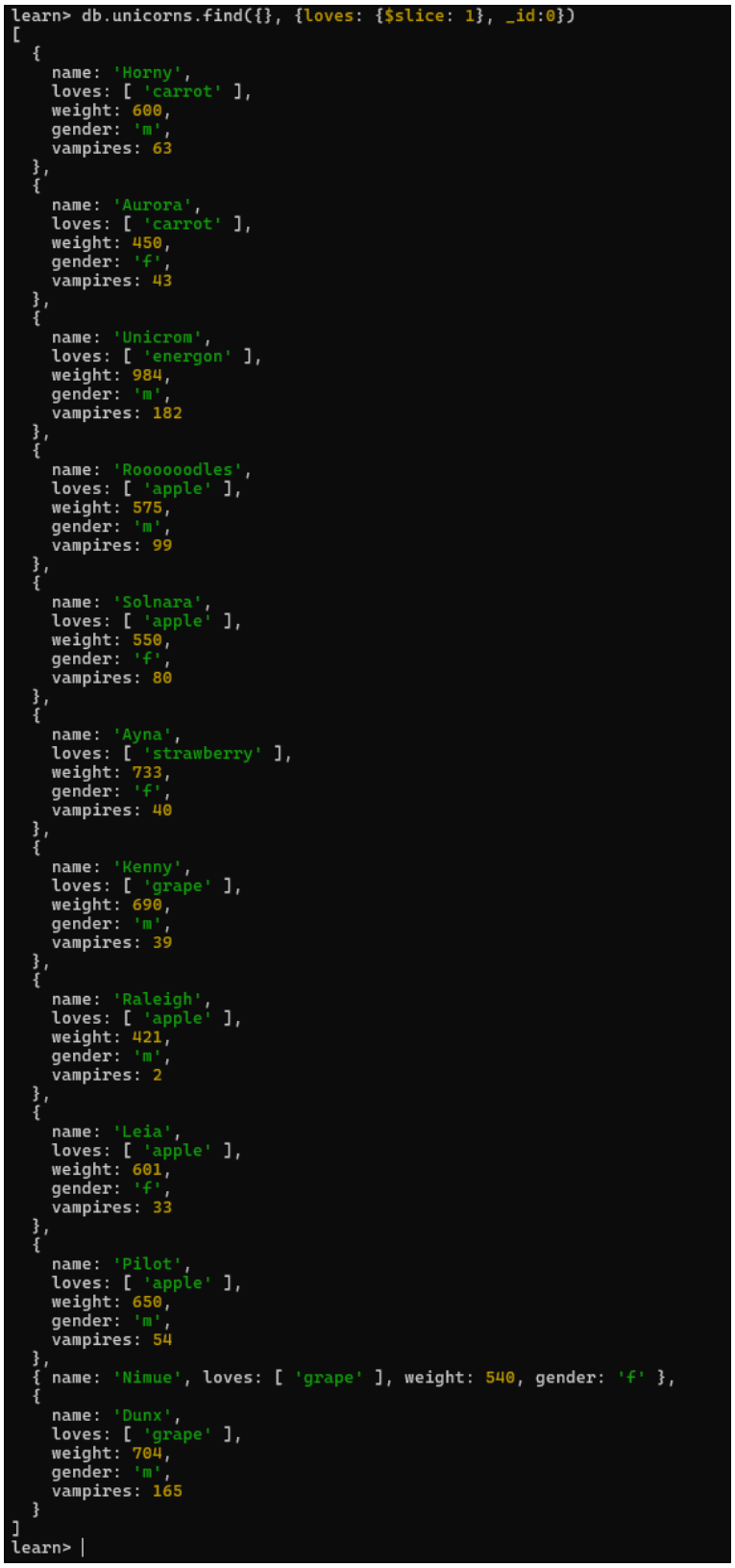
Практическое задание 2.2.3

1) Вывести список единорогов в обратном порядке добавления.

learn> db.unicorns.find().sort({ $natural: -1})

Практическое задание 2.2.4

1) 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>
```

learn> db.unicorns.find({}, {loves: {$slice: 1}, _id:0})

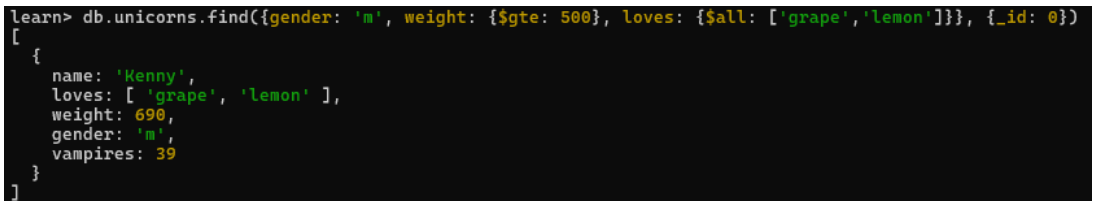
Практическое задание 2.3.1

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

learn> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lt: 700}}, {_id: 0})

Практическое задание 2.3.2

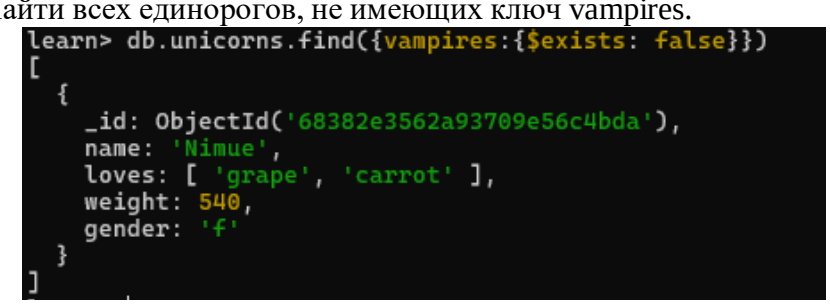
1) Вывести список самцов единорогов весом от полутонны и предпочитающих grape и lemon, исключив вывод идентификатора.



learn> db.unicorns.find({gender: 'm', weight: {$gte: 500}, loves: {$all: ['grape','lemon']}}, {_id: 0})

Практическое задание 2.3.3

1) Найти всех единорогов, не имеющих ключ vampires.



learn> db.unicorns.find({vampires:{$exists: false}})

Практическое задание 2.3.4

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

```
learn> db.unicorns.find({gender: 'm'}, {name: 1, loves: {$slice: 1}, _id: 0}).sort({name: 1})
[
  { name: 'Dunx', loves: [ 'grape' ] },
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Roooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
learn>
```

learn> db.unicorns.find({gender: 'm'}, {name: 1, loves: {$slice: 1}, _id: 0}).sort({name: 1})

Практическое задание 3.1.1

{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.insertMany([{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"}}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68383cb462a93709e56c4bdc'),
    '1': ObjectId('68383cb462a93709e56c4bdd'),
    '2': ObjectId('68383cb462a93709e56c4bde')
  }
}
learn>
```

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' }
  }
]
```

learn> db.towns.find({'mayor.party': 'I'}, {name: 1, mayor: 1, _id: 0})

2) 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' } } ]
learn>
```

learn> db.towns.find({'mayor.party': {$exists: false}}, {name: 1, mayor: 1, _id: 0})

Практическое задание 3.1.2

1) Сформировать функцию для вывода списка самцов единорогов.

```
learn> fn = function() {return this.gender == 'm';}
[Function: fn]
```

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

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

3) Вывести результат, используя forEach.

```
learn> cursor.forEach(function(obj){print(obj.name); })
Dunx
Horny

learn>
```

Практическое задание 3.2.1

1) Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2

1) Вывести список предпочтений.

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

Практическое задание 3.2.3

1) Подсчитать количество особей единорогов обоих полов.

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

Практическое задание 3.3.1

1) Выполнить команду: db.unicorns.insertOne({ name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'}) – выполнил эту команду, так как при использовании save выдавало ошибку db.unicorns.save is not a function

```
learn> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
TypeError: db.unicorns.save is not a function
learn> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedId: ObjectId('683d8e9c62a93709e56c4bdf')
}
```

2) Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('68382e3562a93709e56c4bd0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd4'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd7'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd8'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bda'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('68382efb62a93709e56c4bdb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('683d8e9c62a93709e56c4bdf'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
learn>
```

Практическое задание 3.3.2

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

```
{
  _id: ObjectId('68382e3562a93709e56c4bd5'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
learn> db.unicorns.update({name: 'Ayna', gender: 'f' },{$set: {name: 'Ayna', loves : ['strawberry', 'lemon'], weight: 800, vampires: 51}},{upsert: true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn>
```

2) Проверить содержимое коллекции unicorns

```
{
  _id: ObjectId('68382e3562a93709e56c4bd5'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
```

Практическое задание 3.3.3

1) Для самца единорога Raleigh внести изменения в БД: теперь он любит ред булл.

```
learn> db.unicorns.update({name: 'Raleigh', gender: 'm'},{$set: {loves: ['redbull']}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2) Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId('68382e3562a93709e56c4bd7'),
  name: 'Raleigh',
  loves: [ 'redbull' ],
  weight: 421,
  gender: 'm',
  vampires: 2
},
```

Практическое задание 3.3.4

1) Всем самцам единорогов увеличить количество убитых вампиров на 5.

```
learn> db.unicorns.update({gender: 'm'},{$inc: {vampires: 5}}, {multi: true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
```

2) Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({gender: 'm'})
[
  {
    _id: ObjectId('68382e3562a93709e56c4bd0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 73
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd3'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd6'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd7'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  },
  {
    _id: ObjectId('68382efb62a93709e56c4bdb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170
  },
  {
    _id: ObjectId('683d8e9c62a93709e56c4bdf'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 5
  }
]
```

Практическое задание 3.3.5

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

```
learn> db.towns.update({name: 'Portland'}, {$unset: {'mayor.party': 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2) Проверить содержимое коллекции towns.

```
{
  _id: ObjectId('68383cb462a93709e56c4bde'),
  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

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

```
learn> db.unicorns.update({name: 'Pilot', gender: 'm' },{$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2) Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId('68382e3562a93709e56c4bd9'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  weight: 650,
  gender: 'm',
  vampires: 59
},
```

Практическое задание 3.3.7

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

```
learn> db.unicorns.update({name: 'Aurora', gender: 'f'},{$addToSet: {loves: {$each: ['sugar','lemon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2) Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId('68382e3562a93709e56c4bd1'),
  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"}}])

```
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", popujatiuon: 528000,
... last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"],
... mayor: {
... name: "Sam Adams", party: "D"}}])
...
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('683d9c9b62a93709e56c4be0'),
    '1': ObjectId('683d9c9b62a93709e56c4be1'),
    '2': ObjectId('683d9c9b62a93709e56c4be2')
  }
}
```

2) Удалите документы с беспартийными мэрами.

```
learn> db.towns.remove({'mayor.party': {$exists: false}})
{ acknowledged: true, deletedCount: 3 }
```

3) Проверьте содержание коллекции.

```
learn> db.towns.find()
[
  {
    _id: ObjectId('68383cb462a93709e56c4bdd'),
    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('683d9c9b62a93709e56c4be1'),
    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('683d9c9b62a93709e56c4be2'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

4) Очистите коллекцию.

```
learn> db.towns.remove({})
{ acknowledged: true, deletedCount: 3 }
learn>
```

5) Просмотрите список доступных коллекций

```
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.

```
learn> db.zone.insertMany([ { _id: 'rus', name: 'Russia', description: 'Russia mazafaka' }, { _id: 'usa', name: 'United States of America', description: 'bad lands' }])
{ acknowledged: true, insertedIds: { '0': 'rus', '1': 'usa' } }
learn>
```

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

```
learn> db.unicorns.update({name: 'Horny'}, {$set: {lifeZone: {$ref: 'zone', $id: 'rus'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Aurora'}, {$set: {lifeZone: {$ref: 'zone', $id: 'usa'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

3) Проверьте содержание коллекции единорогов

```
learn> db.unicorns.find({name: {$in: ['Horny', 'Aurora']}})
[
  {
    _id: ObjectId('68382e3562a93709e56c4bd0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 73,
    lifeZone: DBRef('zone', 'rus')
  },
  {
    _id: ObjectId('68382e3562a93709e56c4bd1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    lifeZone: DBRef('zone', 'usa')
  }
]
learn>
```

Практическое задание 4.2.1

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' ]
```

Практическое задание 4.3.1

1) Получите информацию о всех индексах коллекции unicorns.

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

2) Удалите все индексы, кроме индекса для идентификатора.

```
learn> db.unicorns.dropIndex('name_1')
{ nIndexesWas: 2, ok: 1 }
```

3) Попытайтесь удалить индекс для идентификатора.

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

Практическое задание 4.4.1

1) Создайте объемную коллекцию numbers, задействовав курсор: for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}

```
learn> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('683db77a62a93709e56ecbb1') }
}
```

2) Выберите последних четыре документа.

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('683db77a62a93709e56ecbb1'), value: 99999 },
  { _id: ObjectId('683db77a62a93709e56ecbb0'), value: 99998 },
  { _id: ObjectId('683db77a62a93709e56ecbaf'), value: 99997 },
  { _id: ObjectId('683db77a62a93709e56ecbae'), value: 99996 }
]
learn>
```

3) Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis).

```
learn> db.numbers.explain('executionStats').find().sort({value: -1}).limit(4)
{
  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: 103,
```

Запрос выполнялся 103 миллисекунды

4) Создайте индекс для ключа value.

```
learn> db.numbers.ensureIndex({value: 1},{'unique': true})
[ 'value_1' ]
```

5) Получите информацию о всех индексах коллекции numbers.

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

6) Выполните запрос 2.

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('683db77a62a93709e56ecbb1'), value: 99999 },
  { _id: ObjectId('683db77a62a93709e56ecbb0'), value: 99998 },
  { _id: ObjectId('683db77a62a93709e56ecbaf'), value: 99997 },
  { _id: ObjectId('683db77a62a93709e56ecbae'), value: 99996 }
]
```

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

```
learn> db.numbers.explain('executionStats').find().sort({value: -1}).limit(4)
{
  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: true,
          isSparse: false,
          isPartial: false,
          indexVersion: 2,
          direction: 'backward',
          indexBounds: { value: [ '[MaxKey, MinKey]' ] }
        }
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 0,
```

Время выполнения запроса с установленным индексом составило меньше миллисекунды(почти мгновенно)

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

Очевидно, что запрос выполняется быстрее тогда, когда установлен индекс, что показывает время выполнения тестов, в случае, когда отсутствует индекс, время выполнения составляет 103 миллисекунды, тогда как при наличии индекса запрос выполняется почти мгновенно.

2 Вывод

В ходе выполнения лабораторной работы были изучены и отработаны основные принципы работы в СУБД MongoDB. Были получены такие навыки как работа с коллекциями, манипулирование данными внутри коллекций, работа с индексами и ссылками. Цели лабораторной работы были достигнуты.