

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего
образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №6 «Реализация БД с использованием СУБД MongoDB. Запросы к
базе данных.»

по дисциплине «**Проектирование и реализация баз данных**»

Автор: Цой С.А.

Факультет: ИКТ

Группа: K3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Вставка документов в коллекцию	3
Практическое задание 2.1.1	3
Практическое задание 2.2.1	6
Практическое задание 2.2.2	8
Практическое задание 2.2.3	9
Практическое задание 2.2.4	11
Логические операторы	12
Практическое задание 2.3.1	12
Практическое задание 2.3.2	12
Практическое задание 2.3.3	13
Практическое задание 2.3.4	13
Запрос к вложенным объектам	13
Практическое задание 3.1.1	13
Курсоры	14
Практическое задание 3.1.2	14
Практическое задание 3.2.1	15
Практическое задание 3.2.2	15
Редактирование данных	16
Практическое задание 3.3.2	17
Практическое задание 3.3.3	17
Практическое задание 3.3.4	18
Практическое задание 3.3.5	19
Практическое задание 3.3.6	19
Практическое задание 3.3.7	20
Удаление данных из коллекции	21
Практическое задание 3.4.1	21
Ссылки в БД	23
Практическое задание 4.1.1	23
Настройка индексов	24
Практическое задание 4.2.1	24
Управление индексами	24
Практическое задание 4.3.1	24
План Запроса	24
Практическое задание 4.4.1	24
Вывод	26

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).

Вставка документов в коллекцию

Практическое задание 2.1.1

1. Создайте базу данных learn.
2. Заполните коллекцию единорогов unicorns:

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

learn> db.createCollection("unicorns")
{ ok: 1 }
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("658a0d6176fe46494ee1d295") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6276fe46494ee1d296") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6276fe46494ee1d297") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6276fe46494ee1d298") }
}
learn> db.unicorns.insert({name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f', vampires: 80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6376fe46494ee1d299") }
}
learn> db.unicorns.insert({name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6376fe46494ee1d29a") }
}
learn> db.unicorns.insert({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6376fe46494ee1d29b") }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6476fe46494ee1d29c") }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6476fe46494ee1d29d") }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6476fe46494ee1d29e") }
}
learn>
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a0d6476fe46494ee1d29f") }
}
learn> _
```

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

```
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658a108976fe46494ee1d2a0") }
}
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("658a108976fe46494ee1d2a0") }
}
learn> _
```

4. Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId("658a0d6176fe46494ee1d295"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d296"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d297"),
    name: 'Unicrom',
    loves: [ 'enargon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d298"),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("658a0d6376fe46494ee1d299"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
]
```

```
learn>
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
{
  _id: ObjectId("658a0d6376fe46494ee1d29b"),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 39
},
{
  _id: ObjectId("658a0d6476fe46494ee1d29c"),
  name: 'Raleigh',
  loves: [ 'apple', 'sugar' ],
  weight: 421,
  gender: 'm',
  vampires: 2
},
{
  _id: ObjectId("658a0d6476fe46494ee1d29d"),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33
},
{
  _id: ObjectId("658a0d6476fe46494ee1d29e"),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon' ],
  weight: 650,
  gender: 'm',
  vampires: 54
},
{
  _id: ObjectId("658a0d6476fe46494ee1d29f"),
  name: 'Nimue',
  loves: [ 'grape', 'carrot' ],
  weight: 540,
  gender: 'f'
},
{
  _id: ObjectId("658a108976fe46494ee1d2a0"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
]
learn> _
```

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})
[
  {
    _id: ObjectId("658a108976fe46494ee1d2a0"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("658a0d6176fe46494ee1d295"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("658a0d6376fe46494ee1d29b"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39,
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    _id: ObjectId("658a0d6476fe46494ee1d29e"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29c"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d298"),
    name: 'Rooooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d297"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
```

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId("658a0d6276fe46494ee1d296"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("658a0d6376fe46494ee1d29a"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29d"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

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

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId("658a0d6276fe46494ee1d296"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId("658a0d6276fe46494ee1d296"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```


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'}, {gender: 0, loves: 0}).sort({name:1})
[
  {
    _id: ObjectId("658a108976fe46494ee1d2a0"),
    name: 'Dunx',
    weight: 704,
    vamps: 165
  },
  {
    _id: ObjectId("658a0d6176fe46494ee1d295"),
    name: 'Horny',
    weight: 600,
    vamps: 63
  },
  {
    _id: ObjectId("658a0d6376fe46494ee1d29b"),
    name: 'Kenny',
    weight: 690,
    vamps: 39
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29e"),
    name: 'Pilot',
    weight: 650,
    vamps: 54
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29c"),
    name: 'Raleigh',
    weight: 421,
    vamps: 2
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d298"),
    name: 'Rooooooodles',
    weight: 575,
    vamps: 99
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d297"),
    name: 'Unicrom',
    weight: 984,
    vamps: 182
  }
]
```


Практическое задание 2.2.3

1. Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId("658a108976fe46494ee1d2a0"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29f"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29e"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29d"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("658a0d6476fe46494ee1d29c"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("658a0d6376fe46494ee1d29b"),
    name: 'Kenny',
    weight: 733,
    gender: 'f',
    vampires: 40
  },
]
```

```
{
  _id: ObjectId("658a0d6376fe46494ee1d299"),
  name: 'Solnara',
  loves: [ 'apple', 'carrot', 'chocolate' ],
  weight: 550,
  gender: 'f',
  vampires: 80
},
{
  _id: ObjectId("658a0d6276fe46494ee1d298"),
  name: 'Rooooooodles',
  loves: [ 'apple' ],
  weight: 575,
  gender: 'm',
  vampires: 99
},
{
  _id: ObjectId("658a0d6276fe46494ee1d297"),
  name: 'Unicrom',
  loves: [ 'energon', 'redbull' ],
  weight: 984,
  gender: 'm',
  vampires: 182
},
{
  _id: ObjectId("658a0d6276fe46494ee1d296"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
},
{
  _id: ObjectId("658a0d6176fe46494ee1d295"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
]
```

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: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
]
```

Логические операторы

Практическое задание 2.3.1

1. Вывести список самок единорогов весом от 500кг до 700 кг, исключив вывод идентификатора.

```
learn> db.unicorns.find({weight: {$lt: 700, $gt: 500}, gender: 'f'}, {_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

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

```
learn> db.unicorns.find({weight: {$gt: 500}, loves: {$all: ['grape', 'lemon']}}, {_id: 0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Практическое задание 2.3.3

1. Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId("658a0d6476fe46494ee1d29f"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

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: 'Rooooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
```

Запрос к вложенным объектам

Практическое задание 3.1.1

1. Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.find()
[
  {
    _id: ObjectId("658a2f8976fe46494ee1d2a1"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("658a2fb776fe46494ee1d2a2"),
    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("658a2fcf76fe46494ee1d2a3"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

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

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

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

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

Курсоры

Практическое задание 3.1.2

1. Сформировать функцию для вывода списка самцов единорогов.

```
learn> function printMaleUnicorns(){
... return db.unicorns.find({gender:'m'})
... }
[Function: printMaleUnicorns]
learn> printMaleUnicorns()
learn>
[
  {
    _id: ObjectId("658a0d6176fe46494ee1d295"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d297"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d298"),
```


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

```
learn> var cursor = printMaleUnicorns().sort({name: 1}).limit(2);
learn> cursor.forEach(function(obj){print(obj)});
{
  _id: ObjectId("658a108976fe46494ee1d2a0"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId("658a0d6176fe46494ee1d295"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

Практическое задание 3.2.1

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

```
learn> db.unicorns.find({gender: 'f', weight:{$gt: 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'
]
```

Практическое задание 3.2.3

1. Посчитать количество особей единорогов обоих полов.

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


Редактирование данных

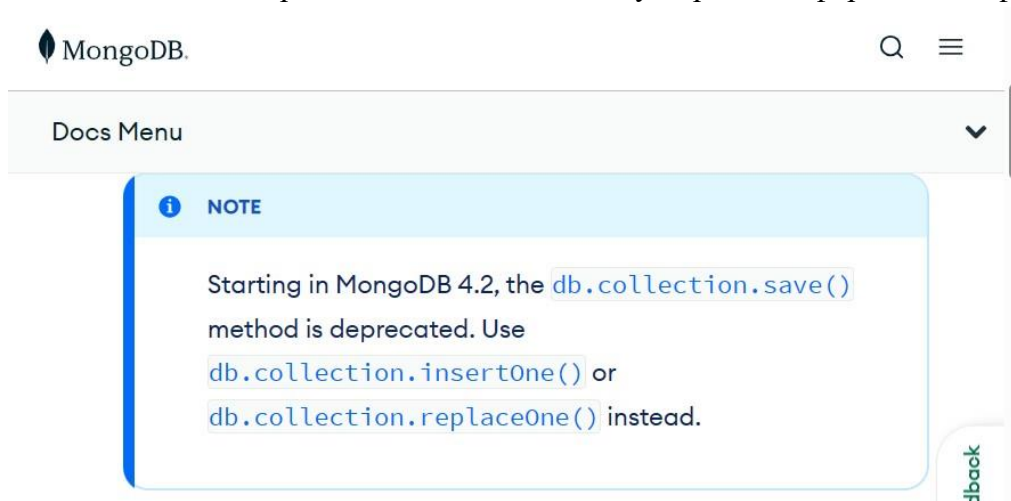
Практическое задание 3.3.1

1. Выполнить команду:

```
> db.unicorns.save({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
```

```
learn> db.unicorns.save({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
TypeError: db.unicorns.save is not a function .exit)
learn> db.unicorns.insert({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
{typeError: db.unicorns.save is not a function
  acknowledged: true,
  insertedIds: { '0': ObjectId("658b2ec376fe46494ee1d2a4") }
}
```

команда save не распознаётся, так как она устарела. Информация с официального сайта



2. Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("658b2ec376fe46494ee1d2a4"),
  name: 'Barney',
  loves: [ 'grape' ],
  weight: 340,
  gender: 'm'
}
```

Практическое задание 3.3.2

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

```
learn> db.unicorns.replaceOne({name: "Ayna"}, {name: "Ayna", loves: ["strawberry", "lemon"],  
weight: 800, gender: "f", vampires: 51})  
{  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 1,  
  modifiedCount: 1,  
  upsertedCount: 0
```

2. Проверить содержимое коллекции unicorns.

```
},  
{  
  _id: ObjectId("658a0d6376fe46494ee1d29a"),  
  name: 'Ayna',  
  loves: [ 'strawberry', 'lemon' ],  
  weight: 800,  
  gender: 'f',  
  vampires: 51  
},  
{  
  id: ObjectId("658a0d6376fe46494ee1d29b"),
```

Практическое задание 3.3.3

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

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

2. Проверить содержимое коллекции unicorns.

```
  _id: ObjectId("658b4a1f76fe46494ee1d2a5"),  
  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.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId("658a0d6176fe46494ee1d295"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d296"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d297"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("658a0d6276fe46494ee1d298"),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId("658a0d6376fe46494ee1d299"),
```

Практическое задание 3.3.5

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

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

2. Проверить содержимое коллекции towns.

```
{
  _id: ObjectId("658a2fcf76fe46494ee1d2a3"),
  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.
2. Проверить содержимое коллекции unicorns.

```
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("658a0d6476fe46494ee1d29e"),
    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.
2. Проверить содержимое коллекции unicorns.

```
]learn> db.unicorns.updateOne({name: 'Aurora'}, {$push: {loves: {$each: ['sugar', 'lemon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId("658a0d6276fe46494ee1d296"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Удаление данных из коллекции

Практическое задание 3.4.1

1. Создайте коллекцию towns, включающую следующие документы

```
learn> db.towns.find()
[
  {
    _id: ObjectId("658a2f8976fe46494ee1d2a1"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("658a2fb776fe46494ee1d2a2"),
    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("658a2fcf76fe46494ee1d2a3"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  },
  {
    _id: ObjectId("658b5b0a76fe46494ee1d2a6"),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("658b5b4376fe46494ee1d2a7"),
    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' }
  }
]

learn>

name: 'Portland',
popujatiuon: 528000,
last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
famous_for: [ 'beer', 'food' ],
mayor: { name: 'Sam Adams', party: 'D' }
]
```

2. Удалите документы с беспартийными мэрами.

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


3. Проверьте содержание коллекции.

```
learn> db.towns.find()
[
  {
    _id: ObjectId("658a2fb776fe46494ee1d2a2"),
    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("658b5b4376fe46494ee1d2a7"),
    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("658b5b6776fe46494ee1d2a8"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

4. Очистите коллекцию.

```
learn> db.towns.drop();
true
learn>
```

5. Просмотрите список доступных коллекций.

```
learn> show collections
unicorns
```


Ссылки в БД

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.places.insertMany([{"_id": "forest", "name": "Forest", "description": "coming soon"},
{"_id": "dreams", "name": "Dreams", "description": "coming later"}])
{ acknowledged: true, insertedIds: { '0': 'forest', '1': 'dreams' } }
learn> _
```

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

```
learn> var forestId = db.places.findOne({_id: "forest"})._id
learn> var dreamsId = db.places.findOne({_id: "dreams"})._id
learn> db.unicorns.updateMany({name: "Ayna"},{$set: {places: {$ref: "places", $id: forestId}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateMany({name: "Aurora"},{$set: {places: {$ref: "places", $id: dreamsId}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

3. Проверьте содержание коллекции единорогов.

```
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId("658a0d6376fe46494ee1d29a"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51,
    places: DBRef("places", 'forest')
  }
]
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId("658a0d6276fe46494ee1d296"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    places: DBRef("places", 'dreams')
  }
]
```

Настройка индексов

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' ]
```

Управление индексами

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

3. Попробуйте удалить индекс для идентификатора.

```
learn> db.unicorns.dropIndexes('_id_')
MongoServerError: cannot drop _id index
```

План Запроса

Практическое задание 4.4.1

1. Создайте объемную коллекцию numbers, задействовав курсор:

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

```
learn> db.createCollection("numbers")
{ ok: 1 }
learn> for(i=0; i < 100000; i++){db.numbers.insert({value:i})}

{
  acknowledged: true,
  insertedIds: { '0': ObjectId("658b815476fe46494ee4dfe8") }
}
```

```
learn> db.numbers.find().count()
100000
```

2. Выберите последних четыре документа.

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

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 122,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
```

4. Создайте индекс для ключа value.

```
learn> db.numbers.createIndex({value: 1})
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' }
]
```

6. Выполните запрос 2.

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 6,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
```

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

Запрос, сделанный после добавления индекса выполнялся гораздо быстрее, чем без индекса и оказался более эффективным

Вывод

В ходе выполнения данной лабораторной работы мы получили практические навыки работы с CRUD-операциями (создание, чтение, обновление, удаление данных) в MongoDB. Мы изучили основные команды для выполнения этих операций в коллекциях MongoDB.

Также была освоена работа с вложенными объектами в коллекциях, выполнение агрегаций данных и изменение данных с помощью ссылок и индексов. В результате приобретён практический опыт в выполнении различных операций с данными в MongoDB.

Лабораторная работа позволила овладеть основными навыками работы с базами данных в MongoDB и получить практический опыт в работе с данными.