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

Отчет

По лабораторной работе №5

по лабораторной работе «Работа с БД в СУБД MongoDB» по дисциплине «Проектирование и реализация баз данных»

Автор: Чан Дык Минь

Факультет: ИКТ

Группа: К32392

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)

Практическое задание 8.1.1:

1) Создайте базу данных learn.

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

2) Заполните коллекцию единорогов unicorns.

```
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("647751db270f0aadb8d24bf5") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647751e3270f0aadb8d24bf6") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647751ee270f0aadb8d24bf7") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647751f7270f0aadb8d24bf8") }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampvampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647751fc270f0aadb8d24bf9") }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64775200270f0aadb8d24bfa") }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64775205270f0aadb8d24bfb") }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6477520b270f0aadb8d24bfc") }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6477520f270f0aadb8d24bfd") }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64775213270f0aadb8d24bfe") }
}
```

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

```
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("6477540f270f0aadb8d24c00") }
}
```

4) Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId("647751db270f0aadb8d24bf5"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("647751e3270f0aadb8d24bf6"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("647751ee270f0aadb8d24bf7"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("647751f7270f0aadb8d24bf8"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("647751fc270f0aadb8d24bf9"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("64775200270f0aadb8d24bfa"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
```

Практическое задание 8.1.2:

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: 'f'}).limit(3).sort({name:1})
[
  {
    _id: ObjectId("647751e3270f0aadb8d24bf6"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("64775200270f0aadb8d24bfa"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("6477520f270f0aadb8d24bfd"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

```
learn> db.unicorns.find({gender: 'm'}).limit(3).sort({name:1})
[
  {
    _id: ObjectId("6477540f270f0aadb8d24c00"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("647751db270f0aadb8d24bf5"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("64775205270f0aadb8d24bfb"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

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

Практическое задание 8.1.3:

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})
[
  {
    _id: ObjectId("647751db270f0aadb8d24bf5"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("647751ee270f0aadb8d24bf7"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("647751f7270f0aadb8d24bf8"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("64775205270f0aadb8d24bfb"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("6477520b270f0aadb8d24bfc"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("64775213270f0aadb8d24bfe"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("6477540f270f0aadb8d24c00"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
```

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId("6477540f270f0aadb8d24c00"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("64775218270f0aadb8d24bff"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("64775213270f0aadb8d24bfe"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("6477520f270f0aadb8d24bfd"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("6477520b270f0aadb8d24bfc"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("64775205270f0aadb8d24bfb"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
```

```
{
  _id: ObjectId("64775200270f0aadb8d24bfa"),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
{
  _id: ObjectId("647751fc270f0aadb8d24bf9"),
  name: 'Solnara',
  loves: [ 'apple', 'carrot', 'chocolate' ],
  weight: 550,
  gender: 'f',
  vampires: 80
},
{
  _id: ObjectId("647751f7270f0aadb8d24bf8"),
  name: 'Roooooodles',
  loves: [ 'apple' ],
  weight: 575,
  gender: 'm',
  vampires: 99
},
{
  _id: ObjectId("647751ee270f0aadb8d24bf7"),
  name: 'Unicrom',
  loves: [ 'energon', 'redbull' ],
  weight: 984,
  gender: 'm',
  vampires: 182
},
{
  _id: ObjectId("647751e3270f0aadb8d24bf6"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
},
{
  _id: ObjectId("647751db270f0aadb8d24bf5"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}

earn>
```

Практическое задание 8.1.5:

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор.

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

Практическое задание 8.1.6:

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

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

Практическое задание 8.1.7:

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

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

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId("64775218270f0aadb8d24bff"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
```

Практическое задание 8.1.9:

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

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

Практическое задание 8.2.1:

1) Создайте коллекцию towns

```
learn> db.createCollection('towns')
```

```
learn> arr = [{
... name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...  name: "Jim Wehrle"
... }},
... {name: "New York",
or: ["beer", "food"],
... 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_fmayor: {
...  name: "Sam Adams",
... party: "D"}}
... ]
```

```
learn> db.towns.insert(arr)
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64775d85270f0aadb8d24c01"),
    '1': ObjectId("64775d85270f0aadb8d24c02"),
    '2': ObjectId("64775d85270f0aadb8d24c03")
  }
}
```

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"}, {_id: 0, name: 1, mayor: 1})
[
  {
    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}}, {_id: 0, name: 1, mayor: 1})
[
  { name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } },
```

Практическое задание 8.2.2:

1) Сформировать функцию для вывода списка самцов единорогов.

```
learn> getMaleUnicorns = function() {return this.gender == 'm';}
```

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

```
learn> let cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2); null;
null
```

3) Вывести результат, используя forEach.

```
learn> cursor.forEach(o => print(o))
{
  _id: ObjectId("6477540f270f0aadb8d24c00"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId("647751db270f0aadb8d24bf5"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

Практическое задание 8.2.3:

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 8.2.4:

Вывести список предпочтений.

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

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

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

Практическое задание 8.2.6:

1) Выполнить команду

```
learn> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedId: ObjectId("64776531270f0aadb8d24c04")
}
```

2) Проверить содержимое коллекции unicorns.

Практическое задание 8.2.7:

1) 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
}
```

2) Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId("64775200270f0aadb8d24bfa"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

Практическое задание 8.2.8:

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

```
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.

```
learn> db.unicorns.find({name: 'Raleigh'})
[
  {
    _id: ObjectId("6477520b270f0aadb8d24bfc"),
    name: 'Raleigh',
    loves: 'redbull',
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

Практическое задание 8.2.9:

1) Всем самцам единорогов увеличить количество убитых вапмиров 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({gender: 'm'})
[
  {
    _id: ObjectId("647751db270f0aadb8d24bf5"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("647751ee270f0aadb8d24bf7"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("647751f7270f0aadb8d24bf8"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
```

Практическое задание 8.2.10:

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: 0,
  upsertedCount: 0
}
```

2) Проверить содержимое коллекции towns.

```
learn> db.towns.find()
[
  {
    _id: ObjectId("64775d85270f0aadb8d24c01"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("64775d85270f0aadb8d24c02"),
    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("64775d85270f0aadb8d24c03"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_fmayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

Практическое задание 8.2.11:

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

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

Практическое задание 8.2.12:

1) 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
}
```

Практическое задание 8.2.13:

1) Создайте коллекцию towns, включающую следующие документы:

```
learn> arrTowns = [
... {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.insert(arrTowns)
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64776a59270f0aadb8d24c05"),
    '1': ObjectId("64776a59270f0aadb8d24c06"),
    '2': ObjectId("64776a59270f0aadb8d24c07")
  }
}
```

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("64775d85270f0aadb8d24c02"),
    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("64776a59270f0aadb8d24c06"),
    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("64776a59270f0aadb8d24c07"),
    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.deleteMany({})
{ acknowledged: true, deletedCount: 3 }
learn>
```

5) Просмотрите список доступных коллекций.

```
learn> show collections
towns
unicorns
```

Практическое задание 8.3.1:

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

```
learn> arrZones = [
... {
... _id: '1',
... name: 'Zone 1',
... description: 'desc 1'
... },
... {
... _id: '2',
... name: 'Zone 2',
... description: 'desc 2'
... },
... {
... _id: '3',
... name: 'Zone 3',
... description: 'desc 3'
... }
... ]
[
  { _id: '1', name: 'Zone 1', description: 'desc 1' },
  { _id: '2', name: 'Zone 2', description: 'desc 2' },
  { _id: '3', name: 'Zone 3', description: 'desc 3' }
]
learn> db.zones.insert(arrZones)
{ acknowledged: true, insertedIds: { '0': '1', '1': '2', '2': '3' } }
learn>
```

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

```
learn> db.unicorns.update({name: 'Pilot'}, {$set: {area: {$ref: 'zones', $id: '1'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Dunx'}, {$set: {area: {$ref: 'zones', $id: '2'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Aurora'}, {$set: {area: {$ref: 'zones', $id: '3'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

3) Проверьте содержание коллекции единорогов.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId("647751db270f0aadb8d24bf5"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("647751e3270f0aadb8d24bf6"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    area: DBRef("zones", '3')
  },
  {
    _id: ObjectId("647751ee270f0aadb8d24bf7"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
```

Практическое задание 8.3.2:

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' ]
```

Задать такой индекс можно

Практическое задание 8.3.3:

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: cannot drop _id index
```

невозможно

Практическое задание 8.3.4:

1) Создайте объемную коллекцию numbers, задействовав курсор.

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

```
learn> for(let i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6477708a270f0aadb8d43cfd") }
}
learn>
```

2) Выберите последних четыре документа.

```
learn> db.numbers.find({}).sort({_id:-1}).limit(4);
[
  { _id: ObjectId("647771ef270f0aadb8d5c39d"), value: 99999 },
  { _id: ObjectId("647771ef270f0aadb8d5c39c"), value: 99998 },
  { _id: ObjectId("647771ef270f0aadb8d5c39b"), value: 99997 },
  { _id: ObjectId("647771ef270f0aadb8d5c39a"), value: 99996 }
]
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 0,
```

4) Создайте индекс для ключа value.

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

5) Получите информацию о всех индексах коллекции numbres.

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

6) Выполните запрос 2.

```
learn> db.numbers.find().sort({_id:-1}).limit(4)
[
  { _id: ObjectId("647771ef270f0aadb8d5c39d"), value: 99999 },
  { _id: ObjectId("647771ef270f0aadb8d5c39c"), value: 99998 },
  { _id: ObjectId("647771ef270f0aadb8d5c39b"), value: 99997 },
  { _id: ObjectId("647771ef270f0aadb8d5c39a"), value: 99996 }
]
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 0,
```

Невозможно сравнить в этом случае (поскольку оба запроса имеют одинаковое время 0 с)

Однако при больших запросах индекс может снизить производительность во время выполнения.

Вывод: В ходе лабораторной работы были изучены основы работы с СУБД MongoDB, создание баз данных и коллекций, а также базовые операции CRUD. Также получен опыт работы с индексами в базах данных NoSQL. Выполнение операций вставки, удаления, обновления, выбора данных. Создаются индексы, сравниваются запросы до и после применения индексов.