ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

Дисциплина:
«Проектирование и реализация баз данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5
«РАБОТА С БД В СУБД MONGODB»

Выполнил:
студент группы K33391
Черкес Артур Викторович

Проверил(а):
Говорова Марина Михайловна

Санкт-Петербург
2023 г.

Цель работы 1.1: овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в колекции 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)

Ход работы:

Практическое задание 8.1.1:

1). Создайте базу данных learn

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

2). Заполните коллекцию единорогов unicorn:

```
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
elon'], 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("647755b67d6f42cad950af25") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af26") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af27") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af28") }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af29") }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af2a") }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af2b") }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af2c") }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af2d") }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af2e") }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647755b67d6f42cad950af2f") }
}
```

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("647756117d6f42cad950af30") }
}
```

4). Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId("647755b67d6f42cad950af25"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("647755b67d6f42cad950af26"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("647755b67d6f42cad950af27"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("647755b67d6f42cad950af28"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("647755b67d6f42cad950af29"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2a"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2a"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2b"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2c"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2d"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2e"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2f"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("647756117d6f42cad950af30"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

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("647755b67d6f42cad950af26"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2a"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2d"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

Список самок

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    _id: ObjectId("647756117d6f42cad950af30"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("647755b67d6f42cad950af25"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2b"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2e"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2c"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("647755b67d6f42cad950af28"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("647755b67d6f42cad950af27"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
```

Список самцов

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("647755b67d6f42cad950af26"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId("647755b67d6f42cad950af26"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

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}).sort({name: 1})
[
  {
    _id: ObjectId("647756117d6f42cad950af30"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId("647755b67d6f42cad950af25"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2b"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2e"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2c"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("647755b67d6f42cad950af28"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("647755b67d6f42cad950af27"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
]
```

Практическое задание 8.1.4

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId("647756117d6f42cad950af30"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2f"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2e"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2d"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2c"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2b"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("647755b67d6f42cad950af2a"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("647755b67d6f42cad950af29"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("647755b67d6f42cad950af28"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("647755b67d6f42cad950af27"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("647755b67d6f42cad950af26"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("647755b67d6f42cad950af25"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
```

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
  },
```

Практическое задание 8.1.6

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

```
learn> db.unicorns.find({weight: {$gte: 500, $lte: 700}, 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 8.1.7

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

```
learn> db.unicorns.find({weight: {$gte: 500}, gender: 'm', loves: {$all: ['lemon', 'grape']}}, {_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("647755b67d6f42cad950af2f"),
    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' ] }
]
```

Практическое задание 8.2.1

1). Создайте коллекцию towns, включающую следующие документы:

```
learn> array = [
... {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"}}]
[
  {
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    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' }
  },
  {
    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.array.find()

learn> db.towns.insert(array)
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("647781707d6f42cad950af31"),
    '1': ObjectId("647781707d6f42cad950af32"),
    '2': ObjectId("647781707d6f42cad950af33")
  }
}
```

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: 0}}, {_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.

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("647756117d6f42cad950af30"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId("647755b67d6f42cad950af25"),
  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: {$gte: 500, $lte: 600}}).count()
(node:25383) [MONGODB DRIVER] Warning: cursor.count is deprecated and will be
(Use `node --trace-warnings ...` to show where the warning was created)
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: 'm', count: 7 }, { _id: 'f', count: 5 } ]
```

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.update({gender: 'f', name: 'Ayna'}, {$set: {weight: 800, vampires: 51}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2). Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId("647755b67d6f42cad950af2a"),
    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.

2). Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.update({gender: 'm', name: 'Raleigh'}, {$addToSet: {loves: "redbull"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Raleigh'})
[
  {
    _id: ObjectId("647755b67d6f42cad950af2c"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

Практическое задание 8.2.9

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

2). Проверить содержимое коллекции unicorns.

```
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'})
[
  {
    _id: ObjectId("647755b67d6f42cad950af25"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("647755b67d6f42cad950af27"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("647755b67d6f42cad950af28"),
    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.

2). Проверить содержимое коллекции towns.

```
learn> db.towns.update({name: 'Portland'}, {$unset: {"mayor.party": 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find()
[
  {
    _id: ObjectId("647781707d6f42cad950af31"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("647781707d6f42cad950af32"),
    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("647781707d6f42cad950af33"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

Практическое задание 8.2.11

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

2). Проверить содержимое коллекции unicorns.

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

```
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId("647755b67d6f42cad950af2e"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

Практическое задание 8.2.12

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

2). Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.update({gender: 'f', 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("647755b67d6f42cad950af26"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 8.2.13

1). Создайте коллекцию towns, включающую следующие документы:

```
learn> towns = [
... {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"}}]
[
  {
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    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' }
  },
  {
    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.city.insertMany(towns)
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("6477a8f17d6f42cad950af34"),
    '1': ObjectId("6477a8f17d6f42cad950af35"),
    '2': ObjectId("6477a8f17d6f42cad950af36")
  }
}
```

2). Удалите документы с беспартийными мэрами.

3). Проверьте содержание коллекции

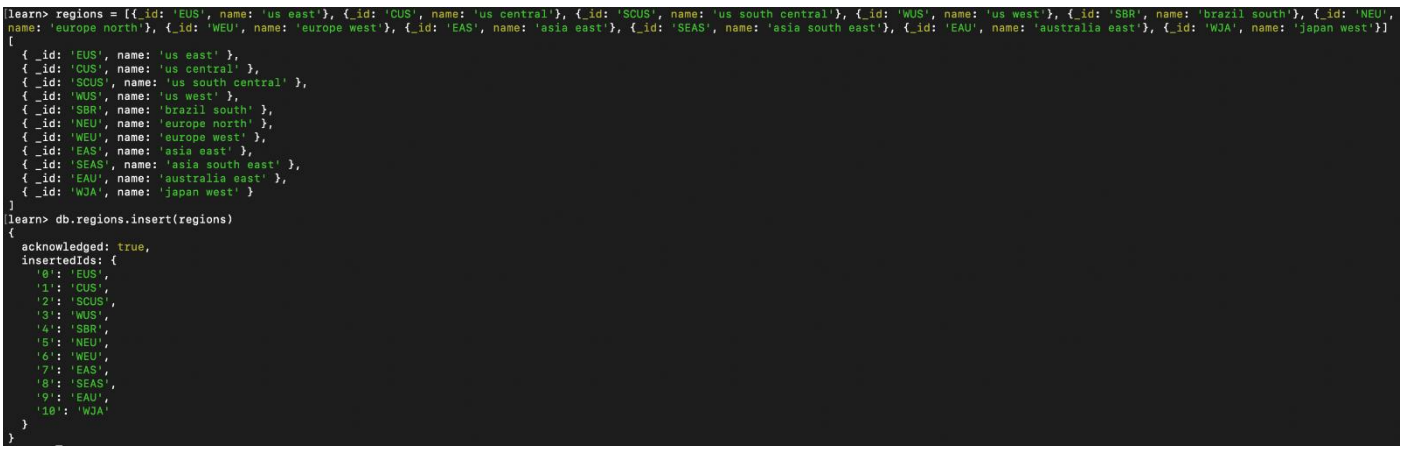
```
learn> db.city.deleteMany({"mayor.party": {$exists: 0}})
{ acknowledged: true, deletedCount: 1 }
learn> db.cityfind()
TypeError: db.cityfind is not a function
learn> db.city.find()
[
  {
    _id: ObjectId("6477a8f17d6f42cad950af35"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId("6477a8f17d6f42cad950af36"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

4). Очистите коллекцию.
5). Просмотрите список доступных коллекций.

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



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

```
learn> db.unicorns.update({name: 'Aurora'}, {$set: {regions: {$ref: "regions", $id: "WUS"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Dunx'}, {$set: {regions: {$ref: "regions", $id: "SBR"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Pilot'}, {$set: {regions: {$ref: "regions", $id: "WJA"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

3). Проверьте содержание коллекции едиорогов.

```
  _id: ObjectId("647755b67d6f42cad950af2d"),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33
},
{
  _id: ObjectId("647755b67d6f42cad950af2e"),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  weight: 650,
  gender: 'm',
  vampires: 59,
  regions: DBRef("regions", 'WJA')
},
{
  _id: ObjectId("647755b67d6f42cad950af2f"),
  name: 'Nimue',
  loves: [ 'grape', 'carrot' ],
  weight: 540,
  gender: 'f'
},
{
  _id: ObjectId("647756117d6f42cad950af30"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 170,
  regions: DBRef("regions", 'SBR')
}
```

Практическое задание 8.3.2

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
```

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(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647895dd7d6f42cad953bc76") }
}
```

2). Выберите последних четыре документа.

```
learn> db.number.find().skip(99996)
[
  { _id: ObjectId("6478973f7d6f42cad9554313"), value: 99996 },
  { _id: ObjectId("6478973f7d6f42cad9554314"), value: 99997 },
  { _id: ObjectId("6478973f7d6f42cad9554315"), value: 99998 },
  { _id: ObjectId("6478973f7d6f42cad9554316"), value: 99999 }
]
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 100000,
  executionTimeMillis: 27,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
```

4). Создайте индекс для ключа value.

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

5). Получите информацию о всех индексах коллекции numbres.

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

6). Выполните запрос 2.

```
learn> db.number.explain("executionStats").find({}).skip(99996)
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 26,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
  executionStages: {
```

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

Сделав несколько запросов я пришёл к выводу, что в данном случае индекс не оказывает существенного влияния на производительность, время запросов с индексом всегда находилось в диапазоне от 1мс до 3мс.

Вывод:

В ходе лабораторной работы №5 были изучены основы установки и работы с СУБД MongoDB, в частности – создание баз данных и коллекций, базовые CRUD операции. Также был получен опыт работы с индексами NoSQL базе данных.

Список команд:

8.1.1
use 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'});
document=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
db.unicorns.insert(document)
db.unicorns.find()
```

8.1.2

```
db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
db.unicorns.find({gender: 'm'}).sort({name: 1})
db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
db.unicorns.findOne({gender: 'f', loves: 'carrot'})
```

8.1.3

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

8.1.4

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

8.1.5

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

8.1.6

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

8.1.7

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

8.1.8

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

8.1.9

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

8.2.1

```
array = [
... {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"}}]
db.towns.insert(array)
db.towns.find({"mayor.party": 'l'}, {_id: 0, name: 1, mayor: 1})
db.towns.find({"mayor.party": {$exists: 0}}, {_id: 0, name: 1, mayor: 1})
```

8.2.2

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

8.2.3

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

8.2.4

```
db.unicorns.distinct("loves")
```

8.2.5

```
db.unicorns.aggregate({"$group": {_id: "$gender", count: {$sum: 1}}})
```

8.2.7

```
db.unicorns.update({gender: 'f', name: 'Ayna'}, {$set: {weight: 800, vampires: 51}})
db.unicorns.find({name: 'Ayna'})
```

8.2.8

```
db.unicorns.update({gender: 'm', name: 'Raleigh'}, {$addToSet: {loves: "redbull"}})
db.unicorns.find({name: 'Raleigh'})
```

8.2.9

```
db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
db.unicorns.find({gender: 'm'})
```

8.2.10

```
db.towns.update({name: 'Portland'}, {$unset: {"mayor.party": 1}})
db.unicorns.find()
```

8.2.11

```
db.unicorns.update({gender: 'm', name: 'Pilot'}, {$addToSet: {loves: "chocolate"}})
db.unicorns.find({name: 'Pilot'})
```

8.2.12

```
db.unicorns.update({gender: 'f', name: 'Aurora'}, {$addToSet: {loves: {$each: ["sugar", "lemons"]}}})
db.unicorns.find({name: 'Aurora'})
```

8.2.13

```
towns = [
... {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"}}]
db.city.insertMany(towns)
db.city.deleteMany({"mayor.party": {$exists: 0}})
db.city.deleteMany({})
show collections
```

8.3.1

```
regions = [{_id: 'EUS', name: 'us east'}, {_id: 'CUS', name: 'us central'}, {_id: 'SCUS', name: 'us south central'}, {_id: 'WUS', name: 'us west'}, {_id: 'SBR', name: 'brazil south'}, {_id: 'NEU', name: 'europe north'}, {_id: 'WEU', name: 'europe west'}, {_id: 'EAS', name: 'asia east'}, {_id: 'SEAS', name: 'asia south east'}, {_id: 'EAU', name: 'australia east'}, {_id: 'WJA', name: 'japan west'}]
[
db.regions.insert(regions)
db.unicorns.update({name: 'Aurora'}, {$set: {regions: {$ref: "regions", $id: "WUS"}}})
db.unicorns.update({name: 'Dunx'}, {$set: {regions: {$ref: "regions", $id: "SBR"}}})
db.unicorns.update({name: 'Pilot'}, {$set: {regions: {$ref: "regions", $id: "WJA"}}})
db.unicorns.find()
```

8.3.2

```
db.unicorns.ensureIndex({"name": 1}, {"unique": true})
```

8.3.3

```
db.unicorns.getIndexes()
db.unicorns.dropIndex("name_1")
db.unicorns.dropIndex("_id_")
```

8.3.4

```
for(i = 0; i < 100000; i++){db.number.insert({value: i})}
db.number.find().skip(99996)
db.number.explain("executionStats").find()
db.number.ensureIndex({"value": 1}, {unique: true})
db.number.explain("executionStats").find({}).skip(99996)
```