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

Отчет

по лабораторной работе №6 «Процедуры, функции, триггеры в PostgreSQL»

по дисциплине **«Работа с БД в СУБД MongoDB»**

Автор: Яковенко К. А.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2024

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).

2.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

2.1.1

Создайте базу данных learn.

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

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('660f9943fe10e8162416c9b5') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995efe10e8162416c9b6') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995efe10e8162416c9b7') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995efe10e8162416c9b8') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995efe10e8162416c9ba') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995ffe10e8162416c9bb') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995ffe10e8162416c9bc') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995ffe10e8162416c9bd') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f995ffe10e8162416c9be') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
```

Используя второй способ, вставьте в коллекцию единорогов документ:

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

```
learn> db.unicorns.insert({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('660f9a64fe10e8162416c9c0') }
}
```

Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('660f9b72fe10e8162416c9c1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('660f9b7afe10e8162416c9c2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('660f9b83fe10e8162416c9c3'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('660f9b8bfe10e8162416c9c4'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('660f9b93fe10e8162416c9c5'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('660f9b9cfe10e8162416c9c6'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
```

2.2 ВЫБОРКА ДАННЫХ ИЗ БД

2.2.1:

1. Сформируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.

   Список самцов, отсортированный по имени:

```
learn> db.unicorns.find({ gender: 'm' }).sort({ name: 1 })
[
  {
    _id: ObjectId('660f9beafe10e8162416c9cc'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('660f9b72fe10e8162416c9c1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('660f9baafe10e8162416c9c7'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('660f9bccfe10e8162416c9ca'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('660f9bb8fe10e8162416c9c8'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('660f9b8bfe10e8162416c9c4'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('660f9b83fe10e8162416c9c3'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
```

   Список самок, сортированный по имени и ограниченный 3 особями:

```
learn> db.unicorns.find({ gender: 'f' }).sort({ name: 1 }).limit(3)
[
  {
    _id: ObjectId('660f9b7afe10e8162416c9c2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('660f9b9cfe10e8162416c9c6'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('660f9bc3fe10e8162416c9c9'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
```

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('660f9b7afe10e8162416c9c2'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

```
learn> db.unicorns.find({ gender: 'f', loves: 'carrot' }).limit(1)
[
  {
    _id: ObjectId('660f9b7afe10e8162416c9c2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

2.2.2

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

```
learn> db.unicorns.find({ gender: 'm' }, { loves: 0, gender: 0 })
[
  {
    _id: ObjectId('660f9b72fe10e8162416c9c1'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('660f9b83fe10e8162416c9c3'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId('660f9b8bfe10e8162416c9c4'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('660f9baafe10e8162416c9c7'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('660f9bb8fe10e8162416c9c8'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('660f9bccfe10e8162416c9ca'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('660f9beafe10e8162416c9cc'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
```

2.2.3

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({ $natural: -1 })
[
  {
    _id: ObjectId('660f9beafe10e8162416c9cc'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('660f9bd6fe10e8162416c9cb'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('660f9bccfe10e8162416c9ca'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('660f9bc3fe10e8162416c9c9'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('660f9bb8fe10e8162416c9c8'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('660f9baafe10e8162416c9c7'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
```

Практическое задание 2.1.4:

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

```
learn> db.unicorns.find({}, { _id: 0, name: 1, first_love: { $arrayElemAt: ["$loves", 0] } })
[
  { name: 'Horny', first_love: 'carrot' },
  { name: 'Aurora', first_love: 'carrot' },
  { name: 'Unicrom', first_love: 'energon' },
  { name: 'Roooooodles', first_love: 'apple' },
  { name: 'Solnara', first_love: 'apple' },
  { name: 'Ayna', first_love: 'strawberry' },
  { name: 'Kenny', first_love: 'grape' },
  { name: 'Raleigh', first_love: 'apple' },
  { name: 'Leia', first_love: 'apple' },
  { name: 'Pilot', first_love: 'apple' },
  { name: 'Nimue', first_love: 'grape' },
  { name: 'Dunx', first_love: 'grape' }
]
```

2.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 2.3.1:

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

```
learn> db.unicorns.find({ gender: 'f', weight: { $gte: 500, $lte: 700 } }, { _id: 0 })
[
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.2:

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

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

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.

```
learn> db.unicorns.find({ vampires: { $exists: false } }, { _id: 0 })
[
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.4:

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

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

3 ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB. ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ

3.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 3.1.1:

1. *Создайте коллекцию towns, включающую следующие документы:*

```
{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",
...         population: 6200,
...         last_sensus: ISODate("2008-01-31"),
...         famous_for: [],
...         mayor: {
...             name: "Jim Wehrle"
...         }
...     },
...     {
...         name: "New York",
...         population: 22200000,
...         last_sensus: ISODate("2009-07-31"),
...         famous_for: ["Statue of Liberty", "food"],
...         mayor: {
...             name: "Michael Bloomberg",
...             party: "I"
...         }
...     },
...     {
...         name: "Portland",
...         population: 528000,
...         last_sensus: ISODate("2009-07-20"),
...         famous_for: ["beer", "food"],
...         mayor: {
...             name: "Sam Adams",
...             party: "D"
...         }
...     }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('660fa156fe10e8162416c9cd'),
    '1': ObjectId('660fa156fe10e8162416c9ce'),
    '2': ObjectId('660fa156fe10e8162416c9cf')
  }
}
```

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.name": 1 })
[ { name: 'New York', mayor: { name: 'Michael Bloomberg' } } ]
```

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.name": 1 })
[ { name: 'Punxsutawney', mayor: { name: 'Jim Wehrle' } } ]
```

Практическое задание 3.1.2:

3. *Сформировать функцию для вывода списка самцов единорогов.*

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

5. *Вывести результат, используя forEach.*

```
learn> var fun = () => db.unicorns.find({gender: 'm'});

learn> var cursor = fun().sort({name: 1}).limit(2);

learn> cursor.forEach((unicorn) => {
... printjson({
... name: unicorn.name,
... weight: unicorn.weight,
... vampires: unicorn.vampires,});});
{
  name: 'Dunx',
  weight: 704,
  vampires: 165
}
{
  name: 'Horny',
  weight: 600,
  vampires: 63
}
```

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2:

Вывести список предпочтений.

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

Практическое задание 3.2.3:

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 3.3.2:

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

```
learn> db.unicorns.updateOne({ name: 'Ayna' }, { $set: { weight: 800, vampires: 51 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
learn> db.unicorns.findOne({ name: 'Anya'});
null
learn> db.unicorns.findOne({ name: 'Ayna'});
{
  _id: ObjectId('660f9b9cfe10e8162416c9c6'),
  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 рэдбул.*
2. *Проверить содержимое коллекции* `unicorns`.

```
learn> db.unicorns.updateOne({ name: 'Raleigh' }, { $addToSet: { loves: 'redbull' } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({ name: 'Raleigh'});
{
  _id: ObjectId('660f9bb8fe10e8162416c9c8'),
  name: 'Raleigh',
  loves: [ 'apple', 'sugar', 'redbull' ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
```

Практическое задание 3.3.4:

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('660f9b72fe10e8162416c9c1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('660f9b83fe10e8162416c9c3'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('660f9b8bfe10e8162416c9c4'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId('660f9baafe10e8162416c9c7'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId('660f9bb8fe10e8162416c9c8'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId('660f9bccfe10e8162416c9ca'),
    name: 'Pilot',
```

Практическое задание 3.3.5:

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

```
learn> db.towns.updateOne({ name: 'Portland' }, { $unset: { "mayor.party": "1" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
learn> db.towns.findOne({ name: 'Portland'});
{
  _id: ObjectId('660fa156fe10e8162416c9cf'),
  name: 'Portland',
  population: 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' }, { $addToSet: { loves: 'chocolate' } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({ name: 'Pilot'});
{
  _id: ObjectId('660f9bccfe10e8162416c9ca'),
  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' }, { $addToSet: { loves: { $each: ['sugar', 'lemo
n'] } } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({ name: 'Aurora'});
{
  _id: ObjectId('660f9b7afe10e8162416c9c2'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

Практическое задание 3.4.1:

4. *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",
...         population: 6200,
...         last_sensus: ISODate("2008-01-31"),
...         famous_for: ["phil the groundhog"],
...         mayor: {
...             name: "Jim Wehrle"
...         }
...     },
...     {
...         name: "New York",
...         population: 22200000,
...         last_sensus: ISODate("2009-07-31"),
...         famous_for: ["status of liberty", "food"],
...         mayor: {
...             name: "Michael Bloomberg",
...             party: "I"
...         }
...     },
...     {
...         name: "Portland",
...         population: 528000,
...         last_sensus: ISODate("2009-07-20"),
...         famous_for: ["beer", "food"],
...         mayor: {
...             name: "Sam Adams",
...             party: "D"
...         }
...     }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('660fbab6fe10e8162416c9d0'),
    '1': ObjectId('660fbab6fe10e8162416c9d1'),
    '2': ObjectId('660fbab6fe10e8162416c9d2')
  }
}
```

5. *Удалите документы с беспартийными мэрами.*

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

6. *Проверьте содержание коллекции.*

```
learn> db.towns.find().toArray()
[
  {
    _id: ObjectId('660fa156fe10e8162416c9ce'),
    name: 'New York',
    population: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'Statue of Liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('660fbab6fe10e8162416c9d1'),
    name: 'New York',
    population: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('660fbab6fe10e8162416c9d2'),
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

7. *Очистите коллекцию.*

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

8. *Просмотрите список доступных коллекций.*

```
learn> db.getCollectionNames()
[ 'unicorns', 'towns' ]
```

4 ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

4.1 ССЫЛКИ В БД

Практическое задание 4.1.1:

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

```
learn> db.habitats.insertMany([
...     {
...         _id: "forest",
...         full_name: "Forest Habitat",
...         description: "A habitat characterized by dense trees and vegetation, providing ampl
e coverage and shelter for unicorns. It is rich in diverse plant and animal life, offering vari
ous food sources for unicorns."
...     },
...     {
...         _id: "grassland",
...         full_name: "Grassland Habitat",
...         description: "A habitat dominated by vast open plains covered with grasses and othe
r herbaceous plants. It provides ideal conditions for unicorns to roam freely and graze on abun
dant vegetation."
...     },
...     {
...         _id: "mountain",
...         full_name: "Mountain Habitat",
...         description: "A habitat located at high elevations with rugged terrain, rocky slope
s, and peaks. Unicorns living in mountain habitats are adapted to harsh conditions and often in
habit alpine meadows and rocky outcrops."
...     },
...     {
...         _id: "coast",
...         full_name: "Coastal Habitat",
...         description: "A habitat situated along coastlines, featuring sandy beaches, rocky s
hores, and tidal pools. Coastal habitats offer a diverse range of ecosystems, including mangrov
e forests, salt marshes, and coral reefs, providing unique habitats for marine life."
...     },
...     {
...         _id: "tundra",
...         full_name: "Tundra Habitat",
...         description: "A cold and treeless biome characterized by low temperatures, permafro
st, and short growing seasons. Tundra habitats are found in the Arctic and Antarctic regions, a
nd unicorns living here are adapted to survive in extreme cold and harsh conditions."
...     }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': 'forest',
    '1': 'grassland',
    '2': 'mountain',
    '3': 'coast',
learn>
  }
}
```

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

```
learn> db.unicorns.updateMany({ name: { $in: ["Aurora", "Pilot", "Raleigh", "Leia"] } },
 { $set: { habitat_id: "forest" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 4,
  modifiedCount: 4,
  upsertedCount: 0
}
```

9. Проверьте содержание коллекции едиорогов.

```
  {
    _id: ObjectId('660f9bb8fe10e8162416c9c8'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7,
    habitat_id: 'forest'
  },
  {
    _id: ObjectId('660f9bc3fe10e8162416c9c9'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33,
    habitat_id: 'forest'
  },
  {
    _id: ObjectId('660f9bccfe10e8162416c9ca'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59,
    habitat_id: 'forest'
  },
```

4.3 УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 4.3.1:

11. *Получите информацию о всех индексах коллекции* `unicorns`.

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

12. *Удалите все индексы, кроме индекса для идентификатора.*

```
db.unicorns.getIndexes().forEach(function(index) {
  if (index.name !== '_id_') {
      db.unicorns.dropIndex(index.name);
  }
});
```

13. *Попытайтесь удалить индекс для идентификатора.*

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

Этот индекс нельзя удалить, так как он создаётся автоматически для идентификации коллекции.

4.4 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})}
```

2. *Выберите последних четыре документа*

```
learn> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
;
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658348f9db12a495f317eabc') }
}
learn> ;

learn> db.numbers.find().count();
100000
learn> db.numbers.find({value: {$in: [9996, 9997, 9998, 9999]}})
[
  { _id: ObjectId('658348d2db12a495f3168b29'), value: 9996 },
  { _id: ObjectId('658348d2db12a495f3168b2a'), value: 9997 },
  { _id: ObjectId('658348d2db12a495f3168b2b'), value: 9998 },
  { _id: ObjectId('658348d2db12a495f3168b2c'), value: 9999 }
]
```

.

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

4. *Создайте индекс для ключа* `value`*.*

```
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 63,
    totalKeysExamined: 0,
    totalDocsExamined: 100000,
```

5. *Получите информацию о всех индексах коллекции* `numbres`.

6. *Выполните запрос 2.*

```
learn> db.numbers.ensureIndex({value: 1})
[ 'value_1' ]
learn> db.numbers.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { value: 1 }, name: 'value_1' }
]
learn> db.numbers.find({value: {$in: [9996, 9997, 9998, 9999]}})
[
  { _id: ObjectId('658348d2db12a495f3168b29'), value: 9996 },
  { _id: ObjectId('658348d2db12a495f3168b2a'), value: 9997 },
  { _id: ObjectId('658348d2db12a495f3168b2b'), value: 9998 },
  { _id: ObjectId('658348d2db12a495f3168b2c'), value: 9999 }
]
```

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

```
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 1,
    totalKeysExamined: 5,
    totalDocsExamined: 4,
```

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

Запросы с индексом выполняются быстрее, следовательно более эффективны.

Вывод

В ходе лабораторной работы была изучена работа с NoSQL БД MongoDB.