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

Отчет

по лабораторной работе «Реализация БД с использованием СУБД MongoDB. Запросы к базе данных»

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

Автор: Нестеренко Ю. А.

Факультет: ИКТ

Группа: К32422

Преподаватель: Говорова М. М.

Дата: 21.05.2023

Санкт-Петербург 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель работы: 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.

1 Практическое задание 8.1.1

1) Создайте базу данных learn.
2) Заполните коллекцию единорогов unicorns.
3) Используя второй способ, вставьте в коллекцию единорогов документ.
4) Проверьте содержимое коллекции с помощью метода find.

1–2

```
test> use learn
switched to db learn
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'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d30906") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d30907") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d30908") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d30909") }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d3090a") }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d3090b") }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d3090c") }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d3090d") }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d3090e") }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d3090f") }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646761f02928c7d078d30910") }
}
```

3

```
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("6467621b2928c7d078d30911") }
}
learn> db.unicorns.find()
[
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("646761f02928c7d078d30907"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646761f02928c7d078d30908"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("646761f02928c7d078d30909"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("646761f02928c7d078d3090a"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
```

4

```
  {
    _id: ObjectId("646761f02928c7d078d3090b"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("646761f02928c7d078d3090c"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("646761f02928c7d078d3090e"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646761f02928c7d078d30910"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
learn>
```

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

1.1

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("646761f02928c7d078d3090c"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("646761f02928c7d078d30909"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("646761f02928c7d078d30908"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
```

1.2

```
learn> db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
[
  {
    _id: ObjectId("646761f02928c7d078d30907"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646761f02928c7d078d3090b"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("646761f02928c7d078d3090e"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

2

```
learn>  db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId("646761f02928c7d078d30907"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
{
  _id: ObjectId("646761f02928c7d078d30907"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn>
```

3 Практическое задание 8.1.3

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

```
learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0})
[
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("646761f02928c7d078d30908"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("646761f02928c7d078d30909"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("646761f02928c7d078d3090c"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
]
```

4 Практическое задание 8.1.4

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("646761f02928c7d078d30910"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646761f02928c7d078d3090e"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("646761f02928c7d078d3090c"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("646761f02928c7d078d3090b"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("646761f02928c7d078d3090a"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("646761f02928c7d078d30909"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("646761f02928c7d078d30908"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("646761f02928c7d078d30907"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
```

5 Практическое задание 8.1.5

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

```
learn> db.unicorns.find({}, {loves: {$slice : 1}, _id: 0});
[
  {
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Aurora',
    loves: [ 'carrot' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Ayna',
    loves: [ 'strawberry' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    name: 'Leia',
    loves: [ 'apple' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  { name: 'Nimue', loves: [ 'grape' ], weight: 540, gender: 'f' },
  {
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

6 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: {$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'
  }
]
```

7 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: {$gte: 500}, loves: {$all: ['grape', 'lemon']}}, {_id: 0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

8 Практическое задание 8.1.8

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId("646761f02928c7d078d30910"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

9 Практическое задание 8.1.9

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

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

10 Практическое задание 8.2.1

1) Создайте коллекцию towns.
2) Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре.
3) Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре.

```
learn> db.towns.insert({name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...     name: "Jim Wehrle"
... }}
... );
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6467d06b2928c7d078d30912") }
}
learn> db.towns.insert({name: "New York",
... populatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...     name: "Michael Bloomberg",
... party: "I"}}
... );
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6467d0892928c7d078d30913") }
}
learn> db.towns.insert({name: "Portland",
... populatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...     name: "Sam Adams",
... party: "D"}}
... );
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6467d0a52928c7d078d30914") }
}



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

11 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.
3) Вывести результат, используя forEach.

```
learn> fn = function() { return this.gender=='m'; }
[Function: fn]
learn> var cursor = db.unicorns.find(fn);null;
null
learn> cursor.limit(2).sort({name: 1});null;
null
learn> cursor.forEach(function(obj){ print(obj); })
MongoInvalidArgumentError: Query filter must be a plain object or ObjectId
```

Поиск с фильтрацией методом find() с использованием функции в качестве аргумента выдает ошибку. Скорее всего, проблема в версии mongodb.

```
learn> var cursor = db.unicorns.find({gender: 'm'});null;
null
learn> cursor.limit(2).sort({name: 1});null;
null
learn> cursor.forEach(function(obj){ print(obj); })
{
  _id: ObjectId("6467621b2928c7d078d30911"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId("646761f02928c7d078d30906"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

12 Практическое задание 8.2.3 / 8.2.4 / 8.2.5

8.2.3. Вывести кол-во самок единорогов весом от полутонны до 600 кг.

8.2.4. Вывести список предпочтений.

8.2.5. Посчитать количество особей единорогов обоих полов.

8.2.3

```
learn> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 600}}).count()
(node:88159) [MONGODB DRIVER] Warning: cursor.count is deprecated and will be removed in the next major
version, please use `collection.estimatedDocumentCount` or `collection.countDocuments` instead
(Use `node --trace-warnings ...` to show where the warning was created)
2
```

8.2.4

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

8.2.5

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

13 Практическое задание 8.2.6

1) Выполнить команду:

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

2) Проверить содержимое коллекции unicorns.

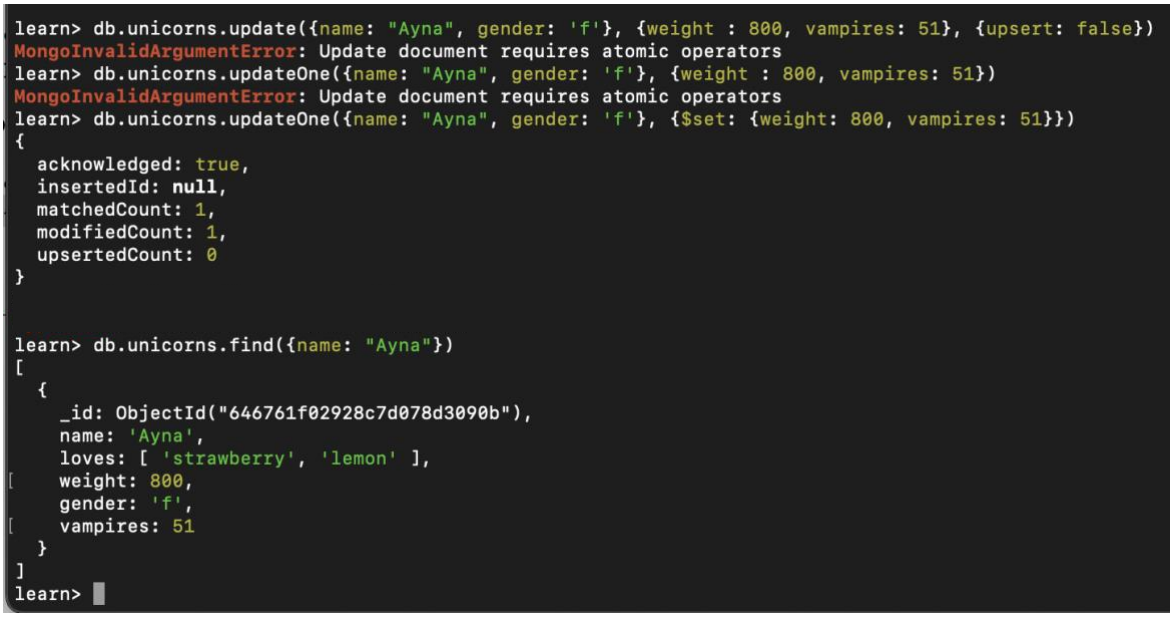
Метод save() не работает. Опять же, похоже, ошибка из-за версии mongodb. Вместо него использован метод insertOne().

```
learn> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'});
TypeError: db.unicorns.save is not a function
learn> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'});
{
  acknowledged: true,
  insertedId: ObjectId("646884a82928c7d078d30915")
}
learn> db.unicorns.find()
[
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("646761f02928c7d078d30907"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646761f02928c7d078d30908"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("646761f02928c7d078d30909"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("646761f02928c7d078d3090a"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("646761f02928c7d078d3090b"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("646761f02928c7d078d3090c"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("646761f02928c7d078d3090e"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646761f02928c7d078d30910"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("646884a82928c7d078d30915"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
learn>
```

14 Практическое задание 8.2.7

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

Метод update() не работает, если явно не использовать оператор $set.



15 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.find({name: "Raleigh"})
[
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn> db.unicorns.updateOne({name: "Raleigh", gender: 'm'}, {$push: {loves: "redbull"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Raleigh"})
[
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

16 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: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: 'm'})
[
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("646761f02928c7d078d30908"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("646761f02928c7d078d30909"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId("646761f02928c7d078d3090c"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44
  },
  {
    _id: ObjectId("646761f02928c7d078d3090d"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  },
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  },
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170
  },
  {
    _id: ObjectId("646884a82928c7d078d30915"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 5
  }
]
```

17 Практическое задание 8.2.10

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: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: "Portland"})
[
  {
    _id: ObjectId("6467d0a52928c7d078d30914"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

18 Практическое задание 8.2.11 / 8.2.12

8.2.11.

1) Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.
2) Проверить содержимое коллекции unicorns.

8.2.12.

1) Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.
2) Проверить содержимое коллекции unicorns.

8.2.11

```
learn> db.unicorns.updateOne({name: "Pilot", gender: 'm'}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId("646761f02928c7d078d3090f"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

8.2.12

```
learn> db.unicorns.updateOne({name: "Aurora", gender: 'f'}, {$addToSet: {loves: {$each: ['sugar', 'lemon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId("646761f02928c7d078d30907"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

19 Практическое задание 8.2.13

1) Создайте коллекцию towns.
2) Удалите документы с беспартийными мэрами.
3) Проверьте содержание коллекции.
4) Очистите коллекцию.
5) Просмотрите список доступных коллекций.

```
learn> db.towns.find()
[
  {
    _id: ObjectId("646893052928c7d078d30916"),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("646893052928c7d078d30917"),
    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("646893052928c7d078d30918"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.remove({"mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
learn>  db.towns.find()
[
  {
    _id: ObjectId("646893052928c7d078d30917"),
    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("646893052928c7d078d30918"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.remove({})
{ acknowledged: true, deletedCount: 2 }
learn> show collections
towns
unicorns
learn>
```

20 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.
3) Проверьте содержание коллекции едиорогов.

```
learn> db.locations.find()
[
  {
    _id: 'fst',
    name: 'Forest',
    description: 'Green and easy to hide in'
  },
  {
    _id: 'mtn',
    name: 'Mountains',
    description: 'Very high and hard to reach'
  },
  { _id: 'bch', name: 'Beach', description: 'Sunny and near the sea' }
]
learn> db.unicorns.update({name: 'Dunx'}, {$set: {location: {$ref:"locations", $id: "fst"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Ayna'}, {$set: {location: {$ref:"locations", $id: "mtn"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Horny'}, {$set: {location: {$ref:"locations", $id: "bch"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({location: {$exists: true}})
[
  {
    _id: ObjectId("646761f02928c7d078d30906"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    location: DBRef("locations", 'bch')
  },
  {
    _id: ObjectId("646761f02928c7d078d3090b"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51,
    location: DBRef("locations", 'mtn')
  },
  {
    _id: ObjectId("6467621b2928c7d078d30911"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170,
    location: DBRef("locations", 'fst')
  }
```

21 Практическое задание 8.3.2

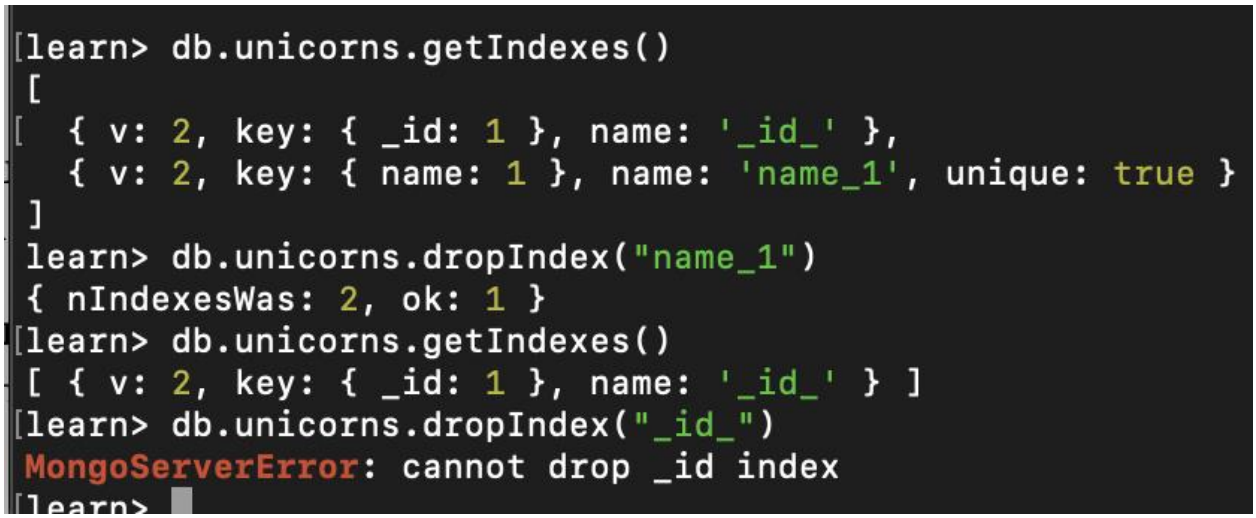
Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique.

```
learn> db.unicorns.ensureIndex({"name" : 1}, {"unique" : true})
[ 'name_1' ]
learn> db.unicorns.insert({name: 'Kenny', loves: ['redbull', 'sugar'], weight: 500, gender: 'm', vampires: 22});
Uncaught:
MongoBulkWriteError: E11000 duplicate key error collection: learn.unicorns index: name_1 dup key: { name: "Kenny" }
Result: BulkWriteResult {
  insertedCount: 0,
  matchedCount: 0,
  modifiedCount: 0,
  deletedCount: 0,
  upsertedCount: 0,
  upsertedIds: {},
  insertedIds: { '0': ObjectId("6468f1e42928c7d078d30919") }
}
[Write Errors: [
  WriteError {
[   err: {
      index: 0,
[     code: 11000,
      errmsg: 'E11000 duplicate key error collection: learn.unicorns index: name_1 dup key: { name: "Kenny" }',
[     errInfo: undefined,
      op: {
        name: 'Kenny',
        loves: [ 'redbull', 'sugar' ],
        weight: 500,
        gender: 'm',
        vampires: 22,
        _id: ObjectId("6468f1e42928c7d078d30919")
      }
    }
  }
]
learn>
```

Для коллекции unicorns индекс для ключа name с флагом unique задать можно. После создания такого индекса вставка документа с повторяющимся значением ключа name становится невозможной, и при попытке сделать это возникает ошибка.

22 Практическое задание 8.3.3

1) Получите информацию о всех индексах коллекции unicorns .
2) Удалите все индексы, кроме индекса для идентификатора.
3) Попытайтесь удалить индекс для идентификатора.



Индекс для идентификатора (ключ _id) удалить нельзя (в отличие от других индексов).

23 Практическое задание 8.3.4

1) Создайте объемную коллекцию numbers, задействовав курсор:
   for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
2) Выберите последних четыре документа.
3) Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis)
4) Создайте индекс для ключа value.
5) Получите информацию о всех индексах коллекции numbres.
6) Выполните запрос 2.
7) Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?
8) Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

При задействовании индекса время выполнения запроса сократилось: с 1 миллисекунды без индекса до 0 миллисекунд с индексом.

```
learn> db.numbers.find().sort({$natural: -1}).limit(4)
[
  { _id: ObjectId("6468f4432928c7d078d48fb9"), value: 99999 },
  { _id: ObjectId("6468f4432928c7d078d48fb8"), value: 99998 },
  { _id: ObjectId("6468f4432928c7d078d48fb7"), value: 99997 },
  { _id: ObjectId("6468f4432928c7d078d48fb6"), value: 99996 }
]
learn> db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: '17830885',
    planCacheKey: '17830885',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: { stage: 'COLLSCAN', direction: 'backward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 1,
    totalKeysExamined: 0,
    totalDocsExamined: 4,
```

```
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().sort({$natural: -1}).limit(4)
[
  { _id: ObjectId("6468f4432928c7d078d48fb9"), value: 99999 },
  { _id: ObjectId("6468f4432928c7d078d48fb8"), value: 99998 },
  { _id: ObjectId("6468f4432928c7d078d48fb7"), value: 99997 },
  { _id: ObjectId("6468f4432928c7d078d48fb6"), value: 99996 }
]
learn> db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: '17830885',
    planCacheKey: '17830885',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: { stage: 'COLLSCAN', direction: 'backward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 4,
```

ВЫВОДЫ

В рамках данной лабораторной работы получены практические навыки работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB. Выполнены 26 практических заданий с использованием консольного клиента mongo. Так как при установке СУБД MongoDB с официального сайта возникли проблемы, СУБД была установлена на компьютер через консоль. В процессе выполнения заданий возникли небольшие трудности с недоступностью некоторых команд, что потребовало поиска дополнительной информации.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Лекция 3. Введение в NoSQL. СУБД MongoDB. Создание БД. Манипулирование данными. 2023. (дата обращения 18.05.2023)
2. How to Install MongoDB on a Mac M1 [Электронный ресурс] // linuxhint. URL: https://linuxhint.com/install-mongodb-mac-m1/?__cf_chl_tk=CiHK93ud3PDSQ1EYOH9CMI_TeGCyuTvxb7_XRAi474g-1684493856-0-gaNycGzNC3s (дата обращения 19.05.2023)
3. MongoDB Database Manual [Электронный ресурс] // MongoDB. 2023. URL: https://www.mongodb.com/docs/manual/ (дата обращения: 20.05.2023).
4. MongoDB Developer Community [Электронный ресурс] // MongoDB. 2023. URL: https://www.mongodb.com/community/forums/c/community/getting-started/ (дата обращения: 20.05.2023).
5. Онлайн-руководство по MongoDB [Электронный ресурс] // METANIT.COM. Сайт о программировании. URL: https://metanit.com/nosql/mongodb/ (дата обращения: 20.05.2023).