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

Отчет
по лабораторной работе №6 «Работа с БД в СУБД MongoDB»

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

Автор: Юркин А.С.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

Элементы оглавления не найдены.

Цель: овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4+, 6.0.6 (текущая).

Практическое задание 2.1.1:

1. *Создайте базу данных learn.*
2.
```
test> use learn
switched to db learn
```
3. *Заполните коллекцию единорогов unicorns:*

```
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'});
```

3. *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('65832e1ddb12a495f3166413') }
}
```

4. *Проверьте содержимое коллекции с помощью метода find.*

2.2 ВЫБОРКА ДАННЫХ ИЗ БД

Пусть в БД добавлены документы:

```
> db.users.insert({"name": "Tom", "age": 28, languages: ["english", "spanish"]})
> db.users.insert({"name": "Bill", "age": 32, languages: ["english", "french"]})
```

```
> db.users.insert({"name": "Tom", "age": 32, languages: ["english", "german"]})
```

Выведем все документы, имеющие name=Tom:

```
> db.users.find({name: "Tom"})
```

```
users> db.users.find({name: "Tom"})
[
  {
    _id: ObjectId('65832ee3db12a495f3166414'),
    name: 'Tom',
    age: 28,
    languages: [ 'english', 'spanish' ]
  },
  {
    _id: ObjectId('65832ee3db12a495f3166416'),
    name: 'Tom',
    age: 32,
    languages: [ 'english', 'german' ]
  }
]
```

```
> db.users.find({languages: "german"})
```

```
users> db.users.find({languages: "german"})
[
  {
    _id: ObjectId('65832ee3db12a495f3166416'),
    name: 'Tom',
    age: 32,
    languages: [ 'english', 'german' ]
  }
]
```

```
> db.users.find({name: "Tom", age: 32})
> db.users.findOne({name: "Tom"})
> db.users.find().limit(3)

> db.users.find().skip(3)

> db.users.find().sort({name: 1})

> db.users.find().sort({name: 1}).skip(3).limit(3)
```

```
users> db.users.find({name: "Tom", age: 32})
[
  {
    _id: ObjectId('65832ee3db12a495f3166416'),
    name: 'Tom',
    age: 32,
    languages: [ 'english', 'german' ]
  }
]
users> db.users.findOne({name: "Tom"})
{
  _id: ObjectId('65832ee3db12a495f3166414'),
  name: 'Tom',
  age: 28,
  languages: [ 'english', 'spanish' ]
}
users> db.users.find().limit(3)
[
  {
    _id: ObjectId('65832ee3db12a495f3166414'),
    name: 'Tom',
    age: 28,
    languages: [ 'english', 'spanish' ]
  },
  {
    _id: ObjectId('65832ee3db12a495f3166415'),
    name: 'Bill',
    age: 32,
    languages: [ 'english', 'french' ]
  },
  {
    _id: ObjectId('65832ee3db12a495f3166416'),
    name: 'Tom',
    age: 32,
    languages: [ 'english', 'german' ]
  }
]
users> db.users.find().skip(3)

users> db.users.find().sort({name: 1})
[
  {
    _id: ObjectId('65832ee3db12a495f3166415'),
    name: 'Bill',
    age: 32,
    languages: [ 'english', 'french' ]
  },
  {
    _id: ObjectId('65832ee3db12a495f3166414'),
    name: 'Tom',
    age: 28,
    languages: [ 'english', 'spanish' ]
  },
  {
    _id: ObjectId('65832ee3db12a495f3166416'),
    name: 'Tom',
    age: 32,
    languages: [ 'english', 'german' ]
  }
]
users> db.users.find().sort({name: 1}).skip(3).limit(3)

users>
```

Практическое задание 2.2.1:

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

   Самцы:

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(3);
[
  {
    _id: ObjectId('65832e1ddb12a495f3166413'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166408'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640e'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Самки:

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3);
[
  {
    _id: ObjectId('65832e0fdb12a495f3166409'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640d'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166410'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'});
{
  _id: ObjectId('65832e0fdb12a495f3166409'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1);
[
  {
    _id: ObjectId('65832e0fdb12a495f3166409'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 2.2.2:

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('65832e0fdb12a495f3166408'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640a'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640b'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640e'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640f'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166411'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('65832e1ddb12a495f3166413'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
]
```

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId('65832e1ddb12a495f3166413'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166412'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166411'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166410'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640f'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640e'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640d'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640c'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
```

Практическое задание 2.2.4:

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

```
learn> db.unicorns.find({}, {_id: 0, loves: {$slice: [0, 1]}});
[
  {
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Aurora',
    loves: [ 'carrot' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Ayna',
    loves: [ 'strawberry' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
```

Практическое задание 2.3.1:

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

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

Практическое задание 2.3.2:

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

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

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.

```
learn> db.unicorns.find({vampires: {$exists: false}});
[
  {
    _id: ObjectId('65832e0fdb12a495f3166412'),
    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.find({gender: 'm'}, {name: 1, _id: 0, loves: {$slice: [0, 1]}});
[
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Unicrom', loves: [ 'energon' ] },
  { name: 'Roooooodles', loves: [ 'apple' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Dunx', loves: [ 'grape' ] }
]
```

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"}}
```

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'}, {'name': 1, 'mayor.name': 1})
[
  {
    _id: ObjectId('65833911db12a495f3166418'),
    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}}, {'name': 1, 'mayor.name': 1})
[
  {
    _id: ObjectId('658338dddb12a495f3166417'),
    name: 'Punxsutawney ',
    mayor: { name: 'Jim Wehrle' }
  }
]
```

Практическое задание 3.1.2:

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

```
learn> function getMaleUnicorns() {return db.unicorns.find({gender: 'm'})
[Function: getMaleUnicorns]
learn> getMaleUnicorns();
[
  {
    _id: ObjectId('65832e0fdb12a495f3166408'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640a'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
```

2. *Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.*
3. *Вывести результат, используя forEach.*

```
learn> var cursor = getMaleUnicorns();

learn> cursor.sort({name: 1}).limit(2);null;
null
learn> cursor.forEach(function(obj){print(obj)});
{
  _id: ObjectId('65832e1ddb12a495f3166413'),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId('65832e0fdb12a495f3166408'),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  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
learn>
```

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 3.3.1:

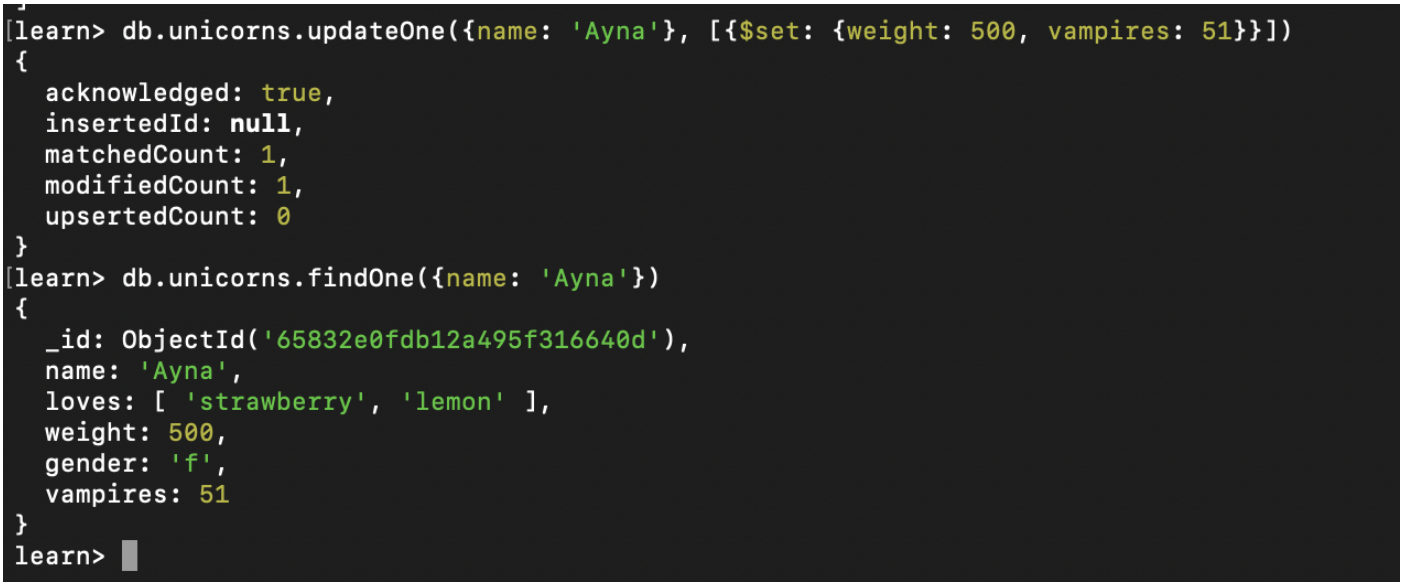
1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции* `unicorns`.

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`.



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'}, [{$set: {'loves': ['redbull']}}])
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({name: 'Raleigh'})
{
  _id: ObjectId('65832e0fdb12a495f316640f'),
  name: 'Raleigh',
  loves: [ '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.find({gender: 'm'})
[
  {
    _id: ObjectId('65832e0fdb12a495f3166408'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640a'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640b'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640e'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
```

```
[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('65832e0fdb12a495f3166408'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640a'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640b'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
```

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'}]);
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: 'Portland'})
[
  {
    _id: ObjectId('65833933db12a495f3166419'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

Практическое задание 3.3.6:

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

```
learn> db.unicorns.updateOne({name: "Pilot"}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('65832e0fdb12a495f3166411'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 64
  }
]
```

Практическое задание 3.3.7:

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

```
learn> db.unicorns.updateOne({name: "Aurora"}, {$push: {loves: {$each: ['sugar', 'lemon']}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId('65832e0fdb12a495f3166409'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 3.4.1:

1. *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"}}
```

2. *Удалите документы с беспартийными мэрами.*

3. *Проверьте содержание коллекции.*

```
[learn> db.towns.deleteMany({'mayor.party': {$exists: false}})
 { acknowledged: true, deletedCount: 3 }
[learn> db.towns.find()
 [
   {
     _id: ObjectId('65833911db12a495f3166418'),
     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('65834427db12a495f316641c'),
     name: 'Portland',
     popujatiuon: 528000,
     last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
     famous_for: [ 'beer', 'food' ],
     mayor: { name: 'Sam Adams', party: 'D' }
   }
 ]
```

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

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

[learn> show collections
 unicorns
```

Практическое задание 4.1.1:

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

```
learn> db.locations.insertMany([
...     {"_id": "ECH", "name": "Enchanted Forest", "description": "A mystical forest brimming with enchantment and hidden wonders."},
...     {"_id": "DIA", "name": "Diamond Valley", "description": "A dazzling valley where diamond-like crystals form natural sculptures."},
...     {"_id": "LUN", "name": "Lunar Plains", "description": "Expansive plains illuminated by a permanent luminescence, resembling moonligh
... ]);
{
  acknowledged: true,
  insertedIds: { '0': 'ECH', '1': 'DIA', '2': 'LUN' }
}
```

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

```
[learn> db.unicorns.updateMany({loves: {$in: ['grape']}}, {$set: {location: {$ref: "locations", $id: 'ECH'
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 3,
  upsertedCount: 0
}
```

3. *Проверьте содержание коллекции едиорогов.*

```
learn> db.unicorns.find({loves: {$in: ['grape']}})
[
  {
    _id: ObjectId('65832e0fdb12a495f3166409'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    location: DBRef('locations', 'ECH')
  },
  {
    _id: ObjectId('65832e0fdb12a495f316640e'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 49,
    location: DBRef('locations', 'ECH')
  },
  {
    _id: ObjectId('65832e0fdb12a495f3166412'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f',
    location: DBRef('locations', 'ECH')
  },
  {
    _id: ObjectId('65832e1ddb12a495f3166413'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 175,
    location: DBRef('locations', 'ECH')
  },
  {
    _id: ObjectId('6583406cdb12a495f316641a'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 10,
    location: DBRef('locations', 'ECH')
  }
]
learn>
```

Практическое задание 4.2.1:

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

```
learn> db.unicorns.ensureIndex({'name': 1}, {'unique': true})
[ 'name_1' ]
learn>
```

Практическое задание 4.3.1:

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

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

learn> db.unicorns.getIndexes()
[ { v: 2, key: { _id: 1 }, name: '_id_' } ]
learn> db.unicorns.dropIndex("_id_")
MongoServerError: cannot drop _id index
```

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`*)*

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

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

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?*

Ускорение при запросе с индексами сильно заметно, следовательно запрос на выборку конкретных значений с индексами намного эффективнее такого же запроса, но без индексов.

Вывод

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