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

Отчет

по лабораторной работе «Работа с БД в СУБД MongoDB»
по дисциплине «**Базы данных**»

Автор: Пырков Владислав Вячеславович

Факультет: Инфокоммуникационных технологий (ИКТ)

Группа: К32402

Преподаватель: Говорова М. М.

Дата: 28.05.2023

Санкт-Петербург 2023

ВВЕДЕНИЕ

Цель работы: овладеть практическими навыками установки СУБД MongoDB, 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.

Практическое задание:

1 Выполнение

Практическое задание 8.1.1:

1. Создайте базу данных learn.

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

2. Заполните коллекцию единорогов unicorns:

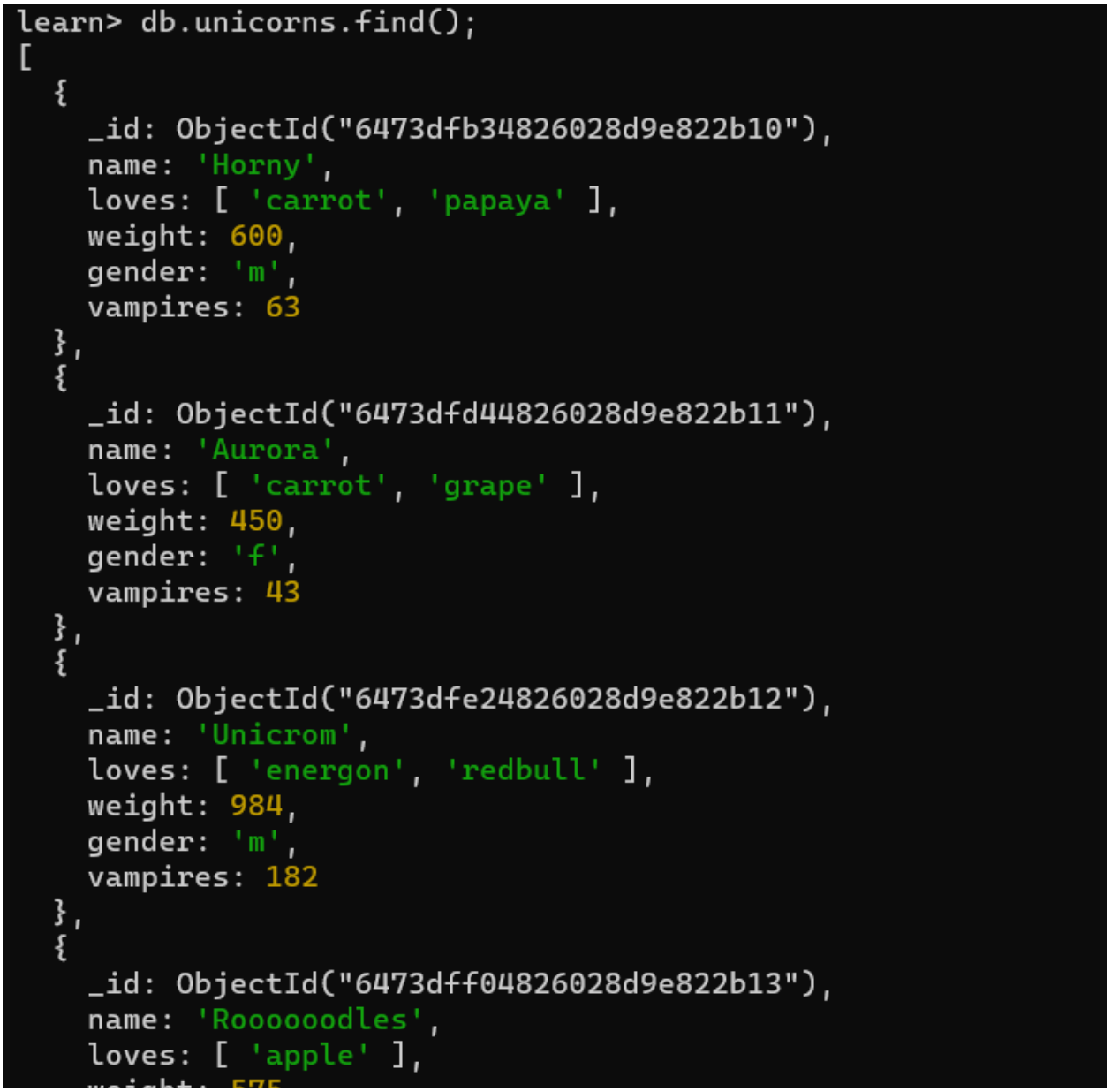
```
learn> db.unicorns.insertOne({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
{
  acknowledged: true,
  insertedId: ObjectId("6473dfb34826028d9e822b10")
}
learn> db.unicorns.insertOne({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedId: ObjectId("6473dfd44826028d9e822b11")
}
learn> db.unicorns.insertOne({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedId: ObjectId("6473dfe24826028d9e822b12")
}
learn> db.unicorns.insertOne({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedId: ObjectId("6473dff04826028d9e822b13")
}
learn> db.unicorns.insertOne({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedId: ObjectId("6473dfff4826028d9e822b14")
}
learn> db.unicorns.insertOne({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedId: ObjectId("6473e00f4826028d9e822b15")
}
learn> db.unicorns.insertOne({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedId: ObjectId("6473e01d4826028d9e822b16")
}
```

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

```
learn> db.unicorns.insertOne({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedId: ObjectId("6473e02c4826028d9e822b17")
}
learn> db.unicorns.insertOne({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedId: ObjectId("6473e0364826028d9e822b18")
}
learn> db.unicorns.insertOne({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedId: ObjectId("6473e0454826028d9e822b19")
}
learn> db.unicorns.insertOne({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedId: ObjectId("6473e0524826028d9e822b1a")
}
learn>
```

```
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.insertOne(document);
{
  acknowledged: true,
  insertedId: ObjectId("6473e0f34826028d9e822b1b")
}
learn>
```

4. Проверьте содержимое коллекции с помощью метода find.



```
learn> db.unicorns.find();
[
  {
    _id: ObjectId("6473dfb34826028d9e822b10"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("6473dfd44826028d9e822b11"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("6473dfe24826028d9e822b12"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("6473dff04826028d9e822b13"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
```

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: "m"}).sort({name:1});
[
  {
    _id: ObjectId("6473e0f34826028d9e822b1b"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6473dfb34826028d9e822b10"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("6473e01d4826028d9e822b16"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("6473e0454826028d9e822b19"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
```

```
  {
    _id: ObjectId("6473e0454826028d9e822b19"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("6473e02c4826028d9e822b17"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("6473dff04826028d9e822b13"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("6473dfe24826028d9e822b12"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
```

```
learn> db.unicorns.find({gender: "m"}).sort({name:1}).limit(3);
[
  {
    _id: ObjectId("6473e0f34826028d9e822b1b"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6473dfb34826028d9e822b10"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("6473e01d4826028d9e822b16"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
learn>
```

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"});
[
  {
    _id: ObjectId("6473dfd44826028d9e822b11"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("6473dfff4826028d9e822b14"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("6473e0524826028d9e822b1a"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

```
learn> db.unicorns.findOne({gender:"f",loves:"carrot"});
{
  _id: ObjectId("6473dfd44826028d9e822b11"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn>
```

```
learn> db.unicorns.find({gender:"f",loves:"carrot"}).limit(1);
[
  {
    _id: ObjectId("6473dfd44826028d9e822b11"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

Практическое задание 8.1.3: 1. 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("6473e0f34826028d9e822b1b"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId("6473dfb34826028d9e822b10"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("6473e01d4826028d9e822b16"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("6473e0454826028d9e822b19"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("6473e02c4826028d9e822b17"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("6473dff04826028d9e822b13"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("6473dfe24826028d9e822b12"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
]
```

Практическое задание 8.1.4: 1. Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({_id:-1});
[
  {
    _id: ObjectId("6473e0f34826028d9e822b1b"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6473e0524826028d9e822b1a"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("6473e0454826028d9e822b19"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("6473e0364826028d9e822b18"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("6473e02c4826028d9e822b17"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
```

```
{
  _id: ObjectId("6473e01d4826028d9e822b16"),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 39
},
{
  _id: ObjectId("6473e00f4826028d9e822b15"),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 733,
  gender: 'f',
  vampires: 40
},
{
  _id: ObjectId("6473dfff4826028d9e822b14"),
  name: 'Solnara',
  loves: [ 'apple', 'carrot', 'chocolate' ],
  weight: 550,
  gender: 'f',
  vampires: 80
},
{
  _id: ObjectId("6473dff04826028d9e822b13"),
  name: 'Roooooodles',
  loves: [ 'apple' ],
  weight: 575,
  gender: 'm',
  vampires: 99
},
{
  _id: ObjectId("6473dfe24826028d9e822b12"),
  name: 'Unicrom',
  loves: [ 'energon', 'redbull' ],
  weight: 984,
  gender: 'm',
  vampires: 182
},
{
  _id: ObjectId("6473dfd44826028d9e822b11"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
},
{
  _id: ObjectId("6473dfb34826028d9e822b10"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

Практическое задание 8.1.5:

1. 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
  },
```

Практическое задание 8.1.6:

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

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

Практическое задание 8.1.7: 6

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

```
learn> db.unicorns.find({gender:"m", weight:{$gt:500},loves:{$in:["grape","lemon"]}},{_id:0});
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
learn>
```

Практическое задание 8.1.8:

1. Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires:{$exists:false}});
[
  {
    _id: ObjectId("6473e0524826028d9e822b1a"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

Практическое задание 8.1.9:

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

```
learn> db.unicorns.find({gender:"m"}, {loves: {$slice:1}, _id:0,weight:0,gender:0,vampires: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' ] }
]
learn>
```

Практическое задание 8.2.1:

1. Создайте коллекцию towns, включающую следующие документы:

```
learn> db.createCollection("towns");
{ ok: 1 }
learn> db.towns.insertMany([{name: "Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: [learn> db.towns.insertMany([{name: "
Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: [""], mayor: { name: "Jim Wehrle" }}, {name: "New York", populatiuon: 222
00000, last_sensus: ISODate("2009-07-31"), famous_for: ["status of liberty", "food"],mayor: { name: "Michael Bloomberg", party: "I"}}, {name: "Portland", po
pulatiuon: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: { name: "Sam Adams", party: "D"}}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("6473f24343503b2143a84191"),
    '1': ObjectId("6473f24343503b2143a84192"),
    '2': ObjectId("6473f24343503b2143a84193")
  }
}
learn>
```

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:1, _id:0});
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn>
```

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:1, _id:0});
[ { name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } } ]
learn>
```

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> fn = function () {return db.unicorns.find({ gender: "m" }); };
[Function: fn]
learn> const cursor = fn();null;
null
learn> cursor.sort({name:1}).limit(2);
[
  {
    _id: ObjectId("6473e0f34826028d9e822b1b"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6473dfb34826028d9e822b10"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
learn>
```

3. Вывести результат, используя forEach.

```
learn> fn = function () {return db.unicorns.find({ gender: "m" }); };
[Function: fn]
learn> const cursor = fn(); null;
null
learn> cursor.sort({name:1}).limit(2); null;
null
learn> cursor.forEach((el) => print(el.name));
Dunx
Horny

learn>
```

4. Содержание коллекции единорогов unicorns: Практическое задание 8.2.3: вывести количество самок единорогов весом от полутонны до 600 кг.

```
learn> db.unicorns.find({gender:"f", weight:{$gt:500, $lt:600}}).count();
2
learn>
```

Практическое задание 8.2.4: вывести список предпочтений.

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

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 } ]
learn>
```

Практическое задание 8.2.6:

1. Выполнить команду, Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.insertOne({name:"Barny", loves: ['grape'], weight: 340, gender: 'm'});
{
  acknowledged: true,
  insertedId: ObjectId("6473fc7a43503b2143a84194")
}
```

```
learn> db.unicorns.findOne({name:"Barny"});
{
  _id: ObjectId("6473fc7a43503b2143a84194"),
  name: 'Barny',
  loves: [ 'grape' ],
  weight: 340,
  gender: 'm'
}
learn>
```

Практическое задание 8.2.7:

1. Для самки единорога Ayna внести изменения в БД: теперь ее вес 800, она убила 51 вампира.

2. Проверить содержимое коллекции unicorns

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.updateOne({name:"Ayna"}, {$set: {weight:800, vampires:51}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn>
```

Практическое задание 8.2.9:

1. Всем самцам единорогов увеличить количество убитых вампиров 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();
[
  {
    _id: ObjectId("6473dfb34826028d9e822b10"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("6473dfd44826028d9e822b11"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("6473dfe24826028d9e822b12"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("6473dff04826028d9e822b13"),
    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.findOne({name:"Portland"});
{
  _id: ObjectId("6473f24343503b2143a84193"),
  name: 'Portland',
  populatiuon: 528000,
  last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
  famous_for: [ 'beer', 'food' ],
  mayor: { name: 'Sam Adams' }
}
learn>
```

```
learn> db.unicorns.update({name:"Pilot"}, {$push:{loves:"chocolate"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({name:"Pilot"});
{
  _id: ObjectId("6473e0454826028d9e822b19"),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  weight: 650,
  gender: 'm',
  vampires: 59
}
learn>
```

Практическое задание 8.2.12:

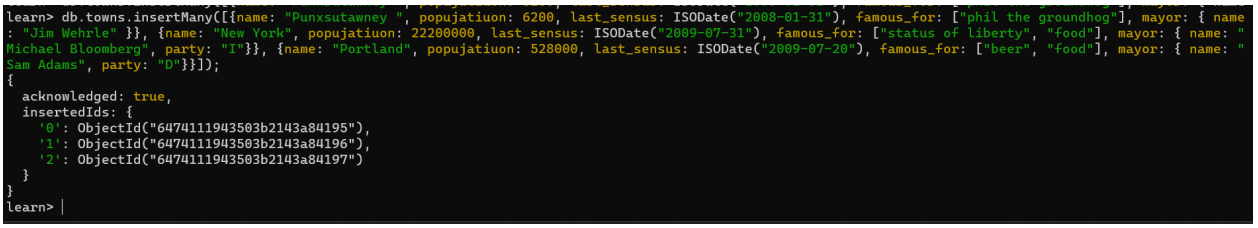
1. Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.

2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.update({name:"Aurora"}, {$push:{loves:"sugar"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name:"Aurora"}, {$push:{loves:"lemon"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({name:"Aurora"});
{
  _id: ObjectId("6473dfd44826028d9e822b11"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn>
```

Практическое задание 8.2.13:

1. Создайте коллекцию towns, включающую следующие документы:



2. Удалите документы с беспартийными мэрами.

3. Проверьте содержание коллекции.

```
learn> db.towns.remove({"mayor.party":{$exists:false}});
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 1 }
learn>
```

```
learn> db.towns.find();
[
  {
    _id: ObjectId("6474111943503b2143a84196"),
    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("6474111943503b2143a84197"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

4. Очистите коллекцию.

5. Просмотрите список доступных коллекций.

```
learn> db.towns.drop()
true
learn> show collections
unicorns
learn>
```

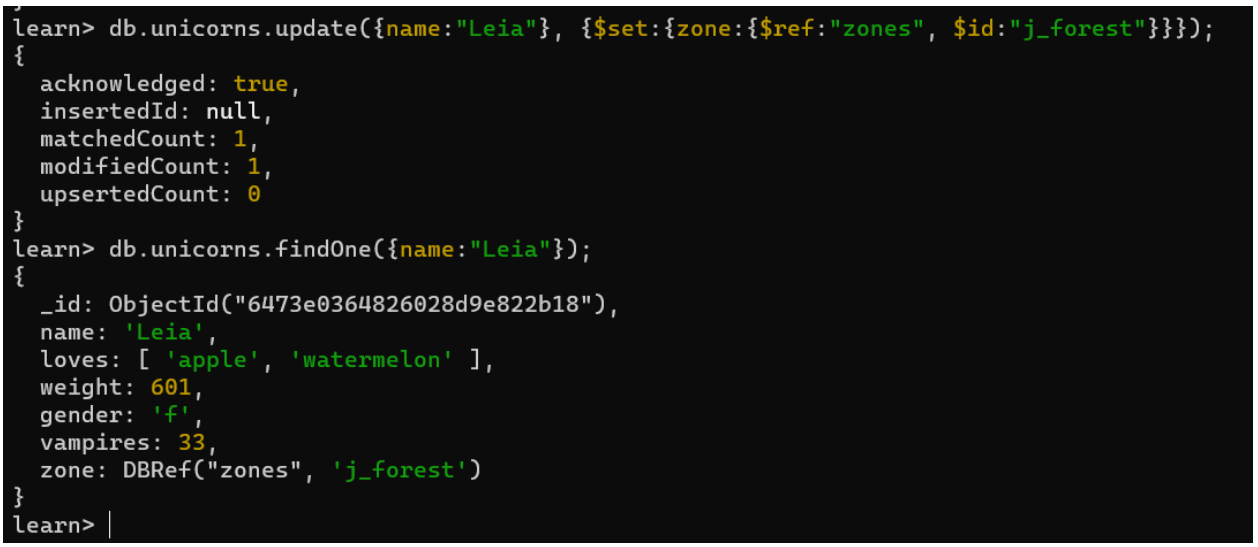
Практическое задание 8.3.1:

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 единорогов.



```
learn> db.createCollection("arials");
{ ok: 1 }
learn> db.arials.insertMany([{name: "forest", cilmate: "average"},{name: "mountains", climate: "cold"},{name: "desert", climate: "hot"}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("647418c143503b2143a841a4"),
    '1': ObjectId("647418c143503b2143a841a5"),
    '2': ObjectId("647418c143503b2143a841a6")
  }
}
learn> db.arials.find();
[
  {
    _id: ObjectId("647418c143503b2143a841a4"),
    name: 'forest',
    cilmate: 'average'
  },
  {
    _id: ObjectId("647418c143503b2143a841a5"),
    name: 'mountains',
    climate: 'cold'
  },
  {
    _id: ObjectId("647418c143503b2143a841a6"),
    name: 'desert',
    climate: 'hot'
  }
]
learn>
```

```
learn> db.unicorns.update({name:"Leia"},{$set:{zone:{$ref:"zones",$climate:"cold"}}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.findOne({name:"Leia"});
{
  _id: ObjectId("6473e0364826028d9e822b18"),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33,
  zone: { '$ref': 'zones', '$climate': 'cold' }
}
learn>
```

Практическое задание 8.3.2:

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

```
learn> db.unicorns.createIndex({"name":1},{"unique":true});
name_1
learn> db.unicorns.insertOne({name:"Pilot"});
MongoServerError: E11000 duplicate key error collection: learn.unicorns index: name_1 dup key: { name: "Pilot" }
learn>
```

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 }
]
learn>
```

2. Удалите все индексы, кроме индекса для идентификатора.

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

3. Попытайтесь удалить индекс для идентификатора.

```
learn> db.unicorns.dropIndexes("_id_");
MongoServerError: cannot drop _id index
learn>
```

Практическое задание 8.3.4:

1. Создайте объемную коллекцию numbers, задействовав курсор:

2. Выберите последних четыре документа.

```
learn> db.numbers.find().sort({$natural:-1}).limit(4);
[
  { _id: ObjectId("64741c6143503b2143a893d8"), value: 21040 },
  { _id: ObjectId("64741c6143503b2143a893d7"), value: 21039 },
  { _id: ObjectId("64741c6143503b2143a893d6"), value: 21038 },
  { _id: ObjectId("64741c6143503b2143a893d5"), value: 21037 }
]
learn>
```

3. Проанализируйте план выполнения запроса

2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis)

```
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: 3,
    totalKeysExamined: 0,
    totalDocsExamined: 4,
    executionStages: {
      stage: 'LIMIT',
      nReturned: 4,
      executionTimeMillisEstimate: 0,
      works: 6,
      advanced: 4,
      needTime: 1,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      limitAmount: 4,
      inputStage: {
```

4. Создайте индекс для ключа value.

5. Получите информацию о всех индексах коллекции numbers.

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

6. Выполните запрос 2

```
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,
    executionStages: {
      stage: 'LIMIT',
      nReturned: 4,
      executionTimeMillisEstimate: 0,
      works: 6,
      advanced: 4,
      needTime: 1,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      limitAmount: 4,
```

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

0 миллисекунд.

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

Время уменьшилось на 3 миллисекунды. По результатам анализа, запрос с индексами оказался эффективнее.

ЗАКЛЮЧЕНИЕ

В ходе работы мы получили теоретические и практические навыки работы с CRUD, вложенными объектами в коллекциях системы управления базы данных MongoDB, агрегации и изменения данных, с ссылками и индексами в базе данных MongoDB.

MongoDB предоставляет мощный CLI интерфейс для выполнения CRUD операций, отличительной особенностью является интеграция полноценного языка программирования: Javascript.