

Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет Инфокоммуникационных технологий

**Лабораторная работа №5 по теме**

**«Работа с БД в СУБД MongoDB»**

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

Выполнил:

студент 2 курса К32421 группы

Козлов Всеволод Денисович

Преподаватель:

Говорова Марина Михайловна

Санкт-Петербург

2023

**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).

## Практическая часть

### 8.1.1

```
// Создайте базу данных learn.
use learn
// Заполните коллекцию единорогов unicorns:
db.unicorns.insertMany([
  {name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender:
'm', vampires: 63},
  {name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender:
'f', vampires: 43},
  {name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984,
gender: 'm', vampires: 182},
  {name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm',
vampires: 99},
  {name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'],
weight: 550, gender: 'f', vampires: 80},
  {name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender:
'f', vampires: 40},
  {name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender:
'm', vampires: 39},
  {name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender:
'm', vampires: 2},
  {name: 'Leia', loves: ['apple', 'watermelon'], weight: 601,
gender: 'f', vampires: 33},
  {name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650,
gender: 'm', vampires: 54},
  {name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender:
'f'}
])

db.unicorns.find();
```

	{_id}	{gender}	{loves}	{name}	{vamp}
1	64745a69cdf3f766d...	m	["carrot", "papaya"]	Horny	
2	64745a69cdf3f766d...	f	["carrot", "grape"]	Aurora	
3	64745a69cdf3f766d...	m	["energon", "redbu"]	Unicrom	
4	64745a69cdf3f766d...	m	["apple"]	Roooooodles	
5	64745a69cdf3f766d...	f	["apple", "carrot"]	Solnara	
6	64745a69cdf3f766d...	f	["strawberry", "le"]	Ayna	
7	64745a69cdf3f766d...	m	["grape", "lemon"]	Kenny	
8	64745a69cdf3f766d...	m	["apple", "sugar"]	Raleigh	
9	64745a69cdf3f766d...	f	["apple", "waterme"]	Leia	
10	64745a69cdf3f766d...	m	["apple", "waterme"]	Pilot	
11	64745a69cdf3f766d...	f	["grape", "carrot"]	Nimue	

// можно и без скобок

```
doc = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704,
gender: 'm', vampires: 165};
db.unicorns.insertOne(doc);
```

```
db.unicorns.find();
```

6	f	["strawberry", "lemon"]	Ayna	40
7	m	["grape", "lemon"]	Kenny	39
8	m	["apple", "sugar"]	Raleigh	2
9	f	["apple", "watermelon"]	Leia	33
10	m	["apple", "watermelon"]	Pilot	54
11	f	["grape", "carrot"]	Nimue	<unset>
12	m	["grape", "watermelon"]	Dunx	165

## 8.1.2

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

```
db.unicorns.find({gender: 'm'}).limit(3).sort({name: 1});
```

	{_id}	{gender}	{loves}	{name}
1	64745aa8cdf3f766d9027013	m	["grape", "watermelon"]	Dunx
2	64745a69cdf3f766d9027007	m	["carrot", "papaya"]	Horny
3	64745a69cdf3f766d902700d	m	["grape", "lemon"]	Kenny

```
db.unicorns.find({gender: 'f'}).limit(3);
```

	{_id}	{gender}	{loves}	{name}
1	64745a69cdf3f766d9027008	f	["carrot", "grape"]	Aurora
2	64745a69cdf3f766d902700b	f	["apple", "carrot", "chocol"]	Solnara
3	64745a69cdf3f766d902700c	f	["strawberry", "lemon"]	Ayna

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

```
db.unicorns.findOne({gender: 'f', loves: 'carrot'});
```

	{_id}	{gender}	{loves}	{name}
1	64745a69cdf3f766d9027008	f	["carrot", "grape"]	Aurora

### 8.1.3

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

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

	{_id}	{name}	{vampires}	{weight}
1	64745a69cdf3f766d9027007	Horny	63	600
2	64745a69cdf3f766d9027009	Unicrom	182	984
3	64745a69cdf3f766d902700a	Roooooodles	99	575
4	64745a69cdf3f766d902700d	Kenny	39	690
5	64745a69cdf3f766d902700e	Raleigh	2	421
6	64745a69cdf3f766d9027010	Pilot	54	650
7	64745aa8cdf3f766d9027013	Dunx	165	704

### 8.1.4

// Вывести список единорогов в обратном порядке добавления.

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

	{ gender ↕	{ loves ↕	{ name ↕	{ vampires ↕
1	m	["grape", "watermelon"]	Dunx	165
2	f	["grape", "carrot"]	Nimue	<unset>
3	m	["apple", "watermelon"]	Pilot	54
4	f	["apple", "watermelon"]	Leia	33
5	m	["apple", "sugar"]	Raleigh	2
6	m	["grape", "lemon"]	Kenny	39
7	f	["strawberry", "lemon"]	Ayna	40
8	f	["apple", "carrot", "choco"]	Solnara	80
9	m	["apple"]	Roooooodles	99
10	m	["energon", "redbull"]	Unicrom	182
11	f	["carrot", "grape"]	Aurora	43
12	m	["carrot", "papaya"]	Horny	63

### 8.1.5

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

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

	{ gender ↕	{ loves ↕	{ name ↕	{ vampires ↕	{ weight ↕
1	m	["carrot"]	Horny	63	600
2	f	["carrot"]	Aurora	43	450
3	m	["energon"]	Unicrom	182	984
4	m	["apple"]	Roooooodles	99	575
5	f	["apple"]	Solnara	80	550
6	f	["strawberry"]	Ayna	40	733
7	f	["grape"]	Kenny	39	690
8	m	["apple"]	Raleigh	2	421
9	f	["apple"]	Leia	33	601
10	m	["apple"]	Pilot	54	650
11	f	["grape"]	Nimue	<unset>	540
12	m	["grape"]	Dunx	165	704

### 8.1.6

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

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

{ loves }	{ name }	{ vampires }	{ weight }
["apple", "carrot", "chocolat"]	Solnara	80	550
["apple", "watermelon"]	Leia	33	601
["grape", "carrot"]	Nimue	<unset>	540

### 8.1.7

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

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

{ gender }	{ loves }	{ name }	{ vampires }	{ weight }
1 m	["grape", "lemon"]	Kenny	39	690

### 8.1.8

```
// Найти всех единорогов, не имеющих ключ vampires.
```

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

{ gender }	{ loves }	{ name }	{ weight }
1 cdf3f766d9027011 f	["grape", "carrot"]	Nimue	540

### 8.1.9

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

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

	{} loves	{} name
1	["grape"]	Dunx
2	["carrot"]	Horny
3	["grape"]	Kenny
4	["apple"]	Pilot
5	["apple"]	Raleigh
6	["apple"]	Roooooodles
7	["energon"]	Unicrom

## 8.2.1

*// Создайте коллекцию towns*

```
db.towns.insertMany([
  {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"}}
]);
```

*// Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре.*

```
db.towns.find({"mayor.party": 'I'}, {mayor: 1, name: 1});
```

	{_id}	{mayor}	{name}
1	64745c18cdf3f766d9027016	{"name": "Michael Bloomberg"}	New York

// Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре.

```
db.towns.find({"mayor.party": {$exists: false}}, {mayor: 1, name: 1});
```

	{_id}	{mayor}	{name}
1	64745c18cdf3f766d9027015	{"name": "Jim Wehrle"}	Punxsutawney

## 8.2.2

```
db.unicorns.deleteMany({});
db.unicorns.insertMany([
  {name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63},
  {name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43},
  {name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182},
  {name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99},
  {name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f', vampires: 80},
  {name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40},
  {name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39},
  {name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2},
  {name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33},
  {name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54},
  {name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'},
  {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
]);
```



```
// Сформировать функцию для вывода списка самцов единорогов.
male = function() {return this.gender == 'm'}

// Создать курсор для этого списка из первых двух особей с сортировкой
в лексикографическом порядке.
var cursor = db.unicorns.find(male); null;
var cursor = db.unicorns.find({gender: 'm'}); null;
var cursor = cursor.limit(2).sort({name:1});null;
// Вывести результат, используя forEach.
cursor.forEach(function (obj){print(obj.name)});
```

	{ } 0
1	Dunx
2	Horny

### 8.2.3

```
// Вывести количество самок единорогов весом от полутонны до 600 кг.
db.unicorns.find({weight: {$gt: 500, $lt: 600}}).count();
```

	{ } result
1	3

### 8.2.4

```
// Вывести список предпочтений.
db.unicorns.distinct("loves")
```

### 8.2.5

```
// Посчитать количество особей единорогов обоих полов.
db.unicorns.aggregate({"$group": {_id: "$gender", count:{$sum:1}}})
```

	{ } _id	{ } count
1	m	7
2	f	5

### 8.2.6

```
// save is deprecated
db.unicorns.save({name: 'Barney', loves: ['grape'], weight:340,
gender: 'm'});
// I will use insertOne instead
```

```
db.unicorns.insertOne({name: 'Barney', loves: ['grape'], weight:340,
gender: 'm'});
db.unicorns.find();
```

{ gender ↕	{ loves ↕	{ name ↕	{ weight ↕
m	["grape"]	Barney	340

### 8.2.7

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

```
db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight: 800, vampires:
51}}, {upsert: false});
// Проверить содержимое коллекции unicorns.
db.unicorns.find();
```

{ loves ↕	{ name ↕	{ vampires ↕	{ weight ↕
["strawberry", "lemon"]	Ayna	51	800

### 8.2.8

// Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.

```
db.unicorns.updateOne({gender: 'm', name: 'Raleigh'}, {$set:{loves:
["redbull"]}}, {upsert: false});
// Проверить содержимое коллекции unicorns.
db.unicorns.find({gender: 'm', name: 'Raleigh'});
```

{ gender ↕	{ loves ↕	{ name ↕	{ vampires ↕	{ weight ↕
m	["redbull"]	Raleigh	2	421

### 8.2.9

// Всем самцам единорогов увеличить количество убитых вампиров на 5.

```
db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
// Проверить содержимое коллекции unicorns.
db.unicorns.find({gender: 'm'});
```

{ gender ↕	{ loves ↕	{ name ↕	{ vampires ↕
m	["carrot", "papaya"]	Horny	68
m	["energon", "redbull"]	Unicrom	187
m	["apple"]	Roooooodles	104
m	["grape", "lemon"]	Kenny	44
m	["redbull"]	Raleigh	7
m	["apple", "watermelon"]	Pilot	59
m	["grape", "watermelon"]	Dunx	170
m	["grape"]	Barney	5

### 8.2.10

// Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.

```
db.towns.updateOne({name: 'Portland'}, {$unset: {"mayor.party": 1}});
```

// Проверить содержимое коллекции towns.

```
db.towns.find({name: 'Portland'}, {mayor: 1});
```

{ _id ↕	{ mayor ↕
1 64745c18cdf3f766d9027017	{"name": "Sam Adams"}

### 8.2.11

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

```
db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
```

// Проверить содержимое коллекции unicorns.

```
db.unicorns.find({name: 'Pilot'});
```

{ loves ↕	{ name ↕	{ vampires ↕	{ weight ↕
["apple", "watermelon", "chocola	Pilot	59	650

### 8.2.12

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

```
db.unicorns.updateOne(
  {name: 'Aurora'},
  {$push: {loves: {$each: ['sugar', 'lemons']}}});
```

// Проверить содержимое коллекции unicorns.

```
db.unicorns.find({name: 'Aurora'});
```

loves	name	vampires	weight
arrot", "grape", "sugar", "lemons"]	Aurora	43	450

### 8.2.13

```

db.towns.deleteMany({});
db.towns.insertMany([
  {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"}}
]);
// Удалите документы с беспартийными мэрами.
db.towns.deleteMany({"mayor.party": {$exists: false}});
// Проверьте содержание коллекции.
db.towns.find({}, {mayor: 1});

```

	_id	mayor
1	64745d7bcd3f766d9027029	{"name": "Michael Bloomberg", "party": "I"}
2	64745d7bcd3f766d902702a	{"name": "Sam Adams", "party": "D"}

```

// Очистите коллекцию.
db.towns.deleteMany({});
// Просмотрите список доступных коллекций.
show collections

```

	<code>{}</code> badge	<code>{}</code> name
1		habitats
2		numbers
3		towns
4		unicorns

### 8.3.1

```
db.unicorns.deleteMany({});
db.unicorns.insertMany([
  {name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender:
'm', vampires: 63},
  {name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender:
'f', vampires: 43},
  {name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984,
gender: 'm', vampires: 182},
  {name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm',
vampires: 99},
  {name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'],
weight: 550, gender: 'f', vampires: 80},
  {name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender:
'f', vampires: 40},
  {name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender:
'm', vampires: 39},
  {name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender:
'm', vampires: 2},
  {name: 'Leia', loves: ['apple', 'watermelon'], weight: 601,
gender: 'f', vampires: 33},
  {name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650,
gender: 'm', vampires: 54},
  {name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender:
'f'},
  {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704,
gender: 'm', vampires: 165}
]);
```

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

```
db.habitats.insertMany([
  {_id: 'ru', name: 'russia', desctiption: 'dangerous place'},
  {_id: 'us', name: 'USA', desctiption: 'tech advanced place'},
```

```

    { _id: 'uk', name: 'united kingdom', desctiption: 'cultural place' }
  ]);
// Включите для нескольких единорогов в документы ссылку на зону
обитания, используя второй способ автоматического связывания.
db.unicorns.updateOne({name: 'Horny'},
  {$set: {habitat: {$ref:"habitats", $id: 'ru'}}});
db.unicorns.updateOne({name: 'Aurora'},
  {$set: {habitat: {$ref:"habitats", $id: 'us'}}});

// Проверьте содержание коллекции едиорогов.
db.unicorns.find();

```

{ habitat	{ loves	{ name
{ "\$ref" : "habitats", "\$id" : "ru" }	["carrot", "papaya"]	Horny
{ "\$ref" : "habitats", "\$id" : "us" }	["carrot", "grape"]	Aurora
<unset>	["energon", "redbull"]	Unicrom
<unset>	["apple"]	Rooodooles
<unset>	["apple", "carrot", "choco"]	Solnara
<unset>	["strawberry", "lemon"]	Ayna
<unset>	["grape", "lemon"]	Kenny
<unset>	["apple", "sugar"]	Raleigh
<unset>	["apple", "watermelon"]	Leia
<unset>	["apple", "watermelon"]	Pilot
<unset>	["grape", "carrot"]	Nimue
<unset>	["grape", "watermelon"]	Dunx

```

// шаманим с получением $id
horny = db.unicorns.findOne({name: 'Horny'});
db.habitats.find({_id: horny.habitat.$id});

```

```

{ map

```

Почему??

## 8.3.2

```

db.unicorns.deleteMany({});

```

```

db.unicorns.insertOne({name: 'Horny', dob: new Date(1992, 2, 13, 7, 47),
loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63});
db.unicorns.insertOne({name: 'Aurora', dob: new Date(1991, 1, 24, 13,
0), loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires:
43});
db.unicorns.insertOne({name: 'Unicrom', dob: new Date(1973, 1, 9, 22,
10), loves: ['energon', 'redbull'], weight: 984, gender: 'm',
vampires: 182});
db.unicorns.insertOne({name: 'Roooooodles', dob: new Date(1979, 7,
18, 18, 44), loves: ['apple'], weight: 575, gender: 'm', vampires:
99});
db.unicorns.insertOne({name: 'Solnara', dob: new Date(1985, 6, 4, 2,
1), loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f',
vampires: 80});
db.unicorns.insertOne({name: 'Ayna', dob: new Date(1998, 2, 7, 8, 30),
loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires:
40});
db.unicorns.insertOne({name: 'Kenny', dob: new Date(1997, 6, 1, 10,
42), loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires:
39});
db.unicorns.insertOne({name: 'Raleigh', dob: new Date(2005, 4, 3, 0,
57), loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires:
2});
db.unicorns.insertOne({name: 'Leia', dob: new Date(2001, 9, 8, 14,
53), loves: ['apple', 'watermelon'], weight: 601, gender: 'f',
vampires: 33});
db.unicorns.insertOne({name: 'Pilot', dob: new Date(1997, 2, 1, 5,
3), loves: ['apple', 'watermelon'], weight: 650, gender: 'm',
vampires: 54});
db.unicorns.insertOne ({name: 'Nimue', dob: new Date(1999, 11, 20,
16, 15), loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
db.unicorns.insertOne({name: 'Dunx', dob: new Date(1976, 6, 18, 18,
18), loves: ['grape', 'watermelon'], weight: 704, gender: 'm',
vampires: 165})

```

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

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

```

{} result
1 name_1

```

### 8.3.3

```
// Получите информацию о всех индексах коллекции unicorns .
```

```
db.unicorns.getIndexes();
```

{ key	{ name	{ v	{ unique
{"_id": new NumberInt("1")}	_id_	2	<unset>
{"name": new NumberInt("1")}	name_1	2	• true

```
// Удалите все индексы, кроме индекса для идентификатора.
```

```
db.unicorns.dropIndex('name_1');
```

	{ nIndexWas	{ ok
1	2	1

```
// Попробуйте удалить индекс для идентификатора.
```

```
db.unicorns.dropIndex('_id');
```

Command failed with error 72 (InvalidOptions): 'cannot drop \_id index' on server localhost:27017. The full response is {"ok": 0.0, "errmsg": "cannot drop \_id index", "code": 72, "codeName": "InvalidOptions"}

### 8.3.4

```
db.numbers.deleteMany({});
```

```
// Создайте объемную коллекцию numbers
```

```
arr = [];
```

```
for (i=0; i<100000; i++)
```

```
{
```

```
    arr.push({value: i})
```

```
}
```

```
db.numbers.insertMany(arr);
```

```
function measureMeanTime(n) {
```

```
    const executionTimes = [];
```

```
    for (i=0; i<n; i++){
```

```
        executionTime = db.numbers.find({}).sort({value:
-1}).limit(4).explain("executionStats").executionStats.executionTimeM
illis;
```

```
        executionTimes.push(executionTime);
```

```
    }
```

```
    const sm = executionTimes.reduce((acc, val) => acc + val, 0);
```

```
    return sm / n;
```

```
}
```

```
print(measureMeanTime(5));
```

	{ 0
1	120



```
// Создайте индекс для ключа value.
db.numbers.createIndex({"value": 1});
db.numbers.getIndexes();
```

	{ } key	{ } name	{ } v
1	{"_id": new NumberInt("1")}	_id_	2
2	{"value": new NumberInt("1")}	value_1	2

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

```
print(measureMeanTime(5));
```

	{ } 0
1	0.4

Вывод: запросы с использованием индексов более эффективны. Эффективность возросла в 300 раз

### Вывод

В ходе выполнения практической работы были успешно овладены практические навыки работы с CRUD-операциями в базе данных MongoDB, а также с вложенными объектами в коллекциях, агрегациями и изменениями данных. Также были изучены ссылки и индексы в MongoDB, что позволяет увеличить производительность и эффективность работы с данными.

Для выполнения работы использовалось программное обеспечение MongoDB 5.0.18, которое демонстрировало высокую надежность и стабильность в работе.

В целом, выполнение данной практической работы позволило приобрести ценный опыт работы с одной из наиболее популярных NoSQL-баз данных, что может быть полезно в дальнейшей профессиональной деятельности.