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 K32392 группы

Никитин Павел Павлович

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.

```
test> use learn
```

// Заполните коллекцию единорогов unicorns:

```
learn> 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'}
... ])
```

// Берем значения из коллекции:

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

// Вывод:

```
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abd7"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
learn>
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abd8"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abd9"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
```

// Создадим

```
learn> doc = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165};
```

// Вставим

```
learn> db.unicorns.insertOne(doc);
```

// Найдем елинорога

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

// Единорог обнаружен

```
{
  _id: ObjectId("6480eeadd545a5fa4791abe1"),
  name: 'Nimue',
  loves: [ 'grape', 'carrot' ],
  weight: 540,
  gender: 'f'
},
{
  _id: ObjectId("6480f013d545a5fa4791abe2"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
```

8.1.2

// 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'}).limit(3).sort({name: 1});
```

```
[
  {
    _id: ObjectId("6480f013d545a5fa4791abe2"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abd7"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abdd"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

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

```
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abd8"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abdb"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abdc"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
]
```

// 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'});
```


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


```
{
  _id: ObjectId("6480eeadd545a5fa4791abd8"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

8.1.3

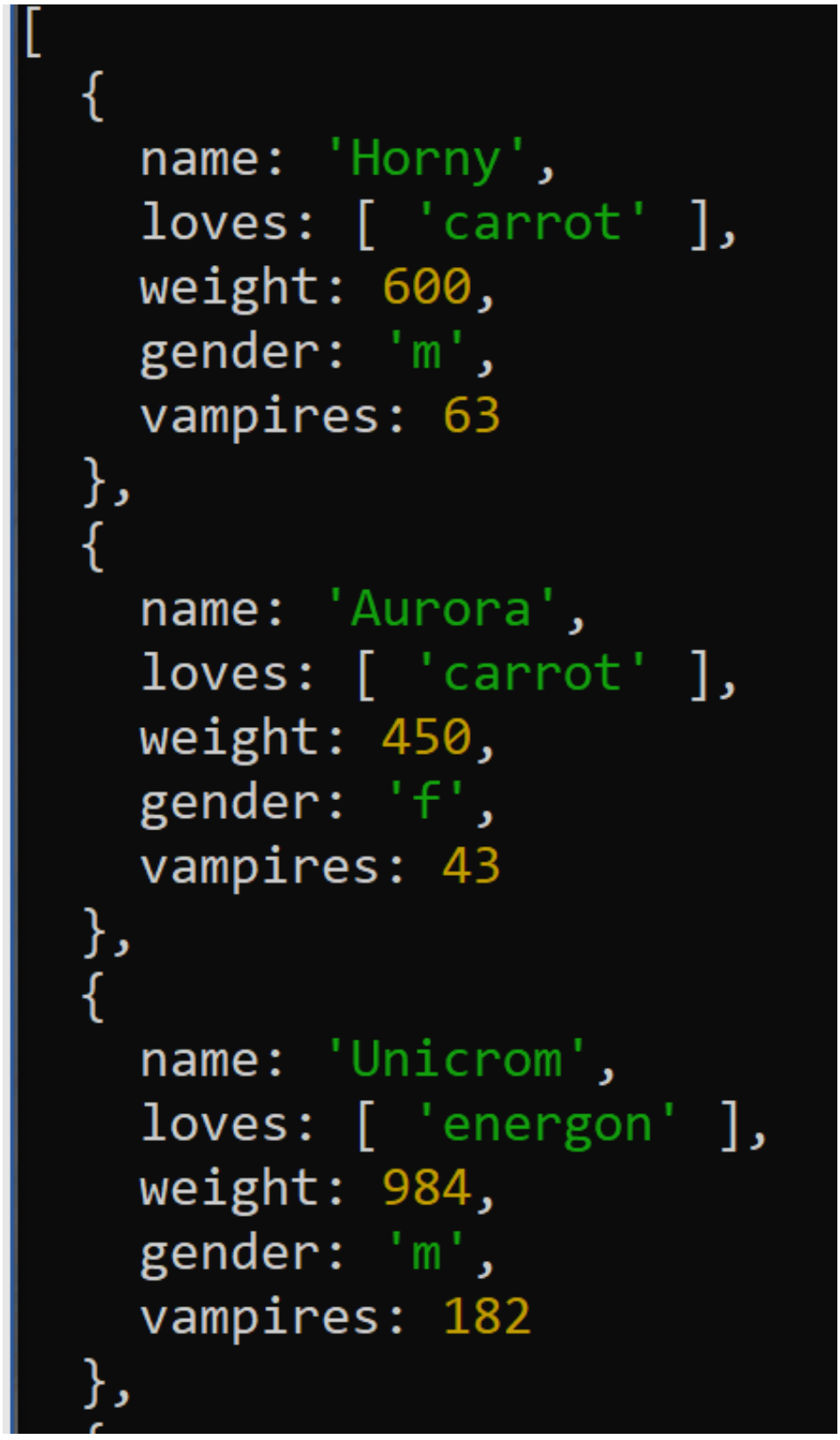
```
// Модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпотениях и поле.
learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender:0});
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abd7"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abd9"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abda"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abdd"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
```

8.1.4

```
// Вывести список единорогов в обратном порядке добавления.
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId("6480f013d545a5fa4791abe2"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abe1"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abe0"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abdf"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
```

8.1.5

```
// Вывести список единорогов с названием первого любимого
предпочтения, исключив идентификатор.
learn> db.unicorns.find({}, {_id:0, loves: {$slice: 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
  },
```

8.1.6

// 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>
```

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

8.1.8

// Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abe1"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

8.1.9

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

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

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.
```

```
learn> db.towns.find({"mayor.party": 'I'}, {mayor: 1, name: 1});
[
  {
    _id: ObjectId("6480fdc5a7612553117aa945"),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

// 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}}, {mayor: 1, name: 1});
[
  {
    _id: ObjectId("6480fdc5a7612553117aa944"),
    name: 'Punxsutawney ',
    mayor: { name: 'Jim Wehrle' }
  }
]
```

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.
learn> male = function() {return this.gender == 'm'}
[Function: male]
// Создать курсор для этого списка из первых двух особей с сортировкой
в лексикографическом порядке.
learn> var cursor = db.unicorns.find(male); null;
null
learn> var cursor = db.unicorns.find({gender: 'm'}); null; var cursor = cursor.limit(2).sort({name:1});null;
null
learn> cursor.forEach(function (obj){print(obj.name);});
Dunx
Horny
```

8.2.3

```
// Вывести количество самок единорогов весом от полутонны до 600 кг.
learn> db.unicorns.find({weight: {$gt: 500, $lt: 600}}).count();
3
```

8.2.4

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

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 } ]
```

8.2.6

```
// save is deprecated
learn> db.unicorns.save({name: 'Barny', loves: ['grape'], weight:340, gender: 'm'});
TypeError: db.unicorns.save is not a function
// I will use insertOne instead

learn> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight:340, gender: 'm'});
{
  acknowledged: true,
  insertedId: ObjectId("6481022aa7612553117aa947")
}
```

```
learn> db.unicorns.find();
[
  {
    _id: ObjectId("6481022aa7612553117aa947"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
learn> _
```

8.2.7

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

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

// Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("6480eeadd545a5fa4791abdc"),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
```

8.2.8

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

```
learn> db.unicorns.updateOne({gender: 'm', name: 'Raleigh'}, {$set:{loves: ["redbull"]}}, {upsert: false});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
learn> // Проверить содержимое коллекции unicorns.

learn> db.unicorns.find({gender: 'm', name: 'Raleigh'});
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abde"),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

8.2.9

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

```
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("6480eeadd545a5fa4791abd7"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("6480eeadd545a5fa4791abd9"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
```

8.2.10

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

```
test> db.towns.updateOne({name: 'Portland'}, {$unset: {"mayor.party": 1}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
test> db.towns.find({name: 'Portland'}, {mayor: 1});
[
  {cknowledged: true,
    _id: ObjectId("6480fdc5a7612553117aa946"),
    mayor: { name: 'Sam Adams' }
  }odifiedCount: 1,
] upsertedCount: 0
```

8.2.11

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

```
learn> db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> b.unicorns.find({name: 'Pilot'});
ReferenceError: b is not defined
learn> db.unicorns.find({name: 'Pilot'});
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abe0"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

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.
```

```
learn> db.unicorns.find({name: 'Aurora'});
[
  {
    _id: ObjectId("6480eeadd545a5fa4791abd8"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemons', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

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.
```

```
learn> db.towns.deleteMany({"mayor.party": {$exists: false}});
{ acknowledged: true, deletedCount: 2 }
learn> db.towns.find({}, {mayor: 1});
[
  {
    _id: ObjectId("6480fdc5a7612553117aa945"),
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn>
```

```
// Очистите коллекцию.
```

```
learn> db.towns.deleteMany({});
{ acknowledged: true, deletedCount: 1 }
learn>
towns
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, использую второй способ автоматического связывания.
```

```
earn> db.unicorns.updateOne({name: 'Horny'},
.. {$set: {habitat: {$ref:"habitats", $id: 'ru'}}}); db.unicorns.updateOne({name: 'Aurora'},
.. {$set: {habitat: {$ref:"habitats", $id: 'us'}}});

 acknowledged: true,
 insertedId: null,
 matchedCount: 1,
 modifiedCount: 1,
 upsertedCount: 0

earn> db.unicorns.find();

 {
   _id: ObjectId("6480eeadd545a5fa4791abd7"),
   name: 'Horny',
   loves: [ 'carrot', 'papaya' ],
   weight: 600,
   gender: 'm',
   vampires: 68,
   habitat: DBRef("habitats", 'ru')
 },
 {
   _id: ObjectId("6480eeadd545a5fa4791abd8"),
   name: 'Aurora',
   loves: [ 'carrot', 'grape', 'sugar', 'lemons', 'sugar', 'lemons' ],
   weight: 450,
   gender: 'f',
   vampires: 43,
   habitat: DBRef("habitats", 'us')
 },
 {
   _id: ObjectId("6480eeadd545a5fa4791abd9"),
```

8.3.2

```
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.

```
learn> db.unicorns.createIndex({name: 1}, {unique: true})
name_1
```

8.3.3

```
// 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 }
]
// Удалите все индексы, кроме индекса для идентификатора.
learn> db.unicorns.dropIndex('name_1');
{ nIndexesWas: 2, ok: 1 }
// Попытайтесь удалить индекс для идентификатора.
learn> db.unicorns.dropIndex('_id_');
MongoServerError: cannot drop _id index
learn>
```

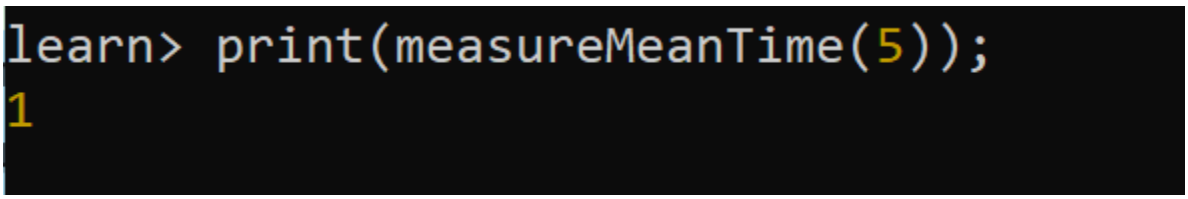
8.3.4

```
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;
}
```

```
learn> print(measureMeanTime(5));
148.4
```

```
// Создайте индекс для ключа value.
learn> db.numbers.createIndex({"value": 1}); db.numbers.getIndexes();
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { value: 1 }, name: 'value_1' }
]
// Сравните время выполнения запросов с индексом и без. Дайте ответ на
вопрос: какой запрос более эффективен?
```



Вывод: запросы с использование индексов более эффективен. Эффективность возросла в 148 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.