САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе №5
по курсу «Проектирование и реализация баз данных»
Тема: Работа с БД в СУБД MongoDB

Выполнил:
Седельников П.В.
К32401

Проверила:
Говорова М.М.

Санкт-Петербург
2023 г.

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

Практическое задание:

1. Изучить функции открытия соединения к базе данных средствами PHP.
2. Изучить основные функции для создания php-скрипта (на базе видео-уроков 1-7).
3. Создать сайт с использованием базовых возможностей PHP (в соответствии с содержанием видео-уроков 1-7).

Выполнение:

1) **Практическое задание 8.1.1:**

Добавление данных:

```
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'});
```

Добавление документа:

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

Просмотр содержимого:

```
db.unicorns.find();
<   {
      _id: ObjectId("646e9b5794597fb029f70fe7"),
      name: 'Horny',
      loves: [
        'carrot',
        'papaya'
      ],
      weight: 600,
      gender: 'm',
```

2) **Практическое задание 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'}).sort({name: 1}).limit(3);
<   {
      _id: ObjectId("646e9ba994597fb029f70ff2"),
      name: 'Dunx',
      loves: [
        'grape',
        'watermelon'
      ],
      weight: 704,
      gender: 'm',
      vampires: 165
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fe7"),
      name: 'Horny',
      loves: [
        'carrot',
        'papaya'
      ],
      weight: 600,
      gender: 'm',
      vampires: 63
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fed"),
      name: 'Kenny',
      loves: [
        'grape',
        'lemon'
      ],
      weight: 690,
      gender: 'm',
      vampires: 39
    }
```

```
> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3);
<   {
      _id: ObjectId("646e9b5794597fb029f70fe8"),
      name: 'Aurora',
      loves: [
        'carrot',
        'grape'
      ],
      weight: 450,
      gender: 'f',
      vampires: 43
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fec"),
      name: 'Ayna',
      loves: [
        'strawberry',
        'lemon'
      ],
      weight: 733,
      gender: 'f',
      vampires: 40
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fef"),
      name: 'Leia',
      loves: [
        'apple',
        'watermelon'
      ],
      weight: 601,
      gender: 'f',
      vampires: 33
    }
```

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: ObjectId("646e9b5794597fb029f70fe8"),
      name: 'Aurora',
      loves: [
        'carrot',
        'grape'
      ],
      weight: 450,
      gender: 'f',
      vampires: 43
    }
> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1);
<   {
      _id: ObjectId("646e9b5794597fb029f70fe8"),
      name: 'Aurora',
      loves: [
        'carrot',
        'grape'
      ],
      weight: 450,
      gender: 'f',
      vampires: 43
    }
```

3) **Практическое задание 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}).sort({name: 1});
<   {
      _id: ObjectId("646e9ba994597fb029f70ff2"),
      name: 'Dunx',
      weight: 704,
      vampires: 165
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fe7"),
      name: 'Horny',
      weight: 600,
      vampires: 63
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fed"),
      name: 'Kenny',
      weight: 690,
      vampires: 39
    }
    {
      _id: ObjectId("646e9b5794597fb029f70ff0"),
      name: 'Pilot',
      weight: 650,
      vampires: 54
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fee"),
      name: 'Raleigh',
      weight: 421,
      vampires: 2
    }
    {
```

4) Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural: -1});
<   {
      _id: ObjectId("646e9ba994597fb029f70ff2"),
      name: 'Dunx',
      loves: [
        'grape',
        'watermelon'
      ],
      weight: 704,
      gender: 'm',
      vampires: 165
    }
    {
      _id: ObjectId("646e9b5794597fb029f70ff1"),
      name: 'Nimue',
      loves: [
        'grape',
        'carrot'
      ],
      weight: 540,
      gender: 'f'
    }
    {
      _id: ObjectId("646e9b5794597fb029f70ff0"),
      name: 'Pilot',
      loves: [
        'apple',
        'watermelon'
      ],
      weight: 650,
      gender: 'm',
      vampires: 54
```

5) **Практическое задание 8.1.5:**

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

6) **Практическое задание 8.1.6:**

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

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

7) Практическое задание 8.1.7:

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

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

8) Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
> db.unicorns.find({vampires: {$exists: false}});
< {
    _id: ObjectId("646e9b5794597fb029f70ff1"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
```

9) Практическое задание 8.1.9:

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

```
> 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'
    ]
  }
  {
```

10) Практическое задание 8.2.1:

Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.insert({name: "Punxsutawney", populatiuon: 6200, last_sensus:
  ISODate("2008-01-31"), famous_for: [""], mayor: {name: "Jim Wehrle"}});
<   {
      acknowledged: true,
      insertedIds: {
        '0': ObjectId("646ea26194597fb029f70ff3")
      }
    }
> db.towns.insert({name: "New York", populatiuon: 22200000, last_sensus:
  ISODate("2009-07-31"), famous_for: ["status of liberty", "food"], mayor: {name:
  "Michael Bloomberg", party: "I"}});
<   {
      acknowledged: true,
      insertedIds: {
        '0': ObjectId("646ea26794597fb029f70ff4")
      }
    }
> db.towns.insert({name: "Portland", populatiuon: 528000, last_sensus:
  ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: {name: "Sam
  Adams", party: "D"}});
⊗ ▸ Error: clone(t={}){const r=t.loc||{};return e({loc:new Position("line"in r?r.l
> db.towns.insert({name: "Portland", populatiuon: 528000, last_sensus:ISODate("200
<   {
      acknowledged: true,
      insertedIds: {
        '0': ObjectId("646ea2d094597fb029f70ff5")
      }
    }
```

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"}, {_id: 0, name: 1, mayor: 1});
<   {
      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.

```
> db.towns.find({"mayor.party": {$exists: false}}, {_id: 0, name: 1, mayor: 1});
<   {
      name: 'Punxsutawney',
      mayor: {
        name: 'Jim Wehrle'
      }
    }
```

11) Практическое задание 8.2.2:

Сформировать функцию для вывода списка самцов единорогов.

```
> function getMales() {return db.unicorns.find({gender: 'm'});};
```

Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.

```
> var cursor = getMales().sort({name: 1}).limit(2);
```

Вывести результат, используя forEach.

```
> cursor.forEach(function(obj) {print(obj);});
<   {
      _id: ObjectId("646e9ba994597fb029f70ff2"),
      name: 'Dunx',
      loves: [ 'grape', 'watermelon' ],
      weight: 704,
      gender: 'm',
      vampires: 165
    }
<   {
      _id: ObjectId("646e9b5794597fb029f70fe7"),
      name: 'Horny',
      loves: [ 'carrot', 'papaya' ],
      weight: 600,
      gender: 'm',
      vampires: 63
    }
learn>
```

12) Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 600}}).count(true);
<   2
```

13) Практическое задание 8.2.4:

Вывести список предпочтений.

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

14) Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

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

15) Практическое задание 8.2.6:

Выполнить команду

```
> db.unicorns.updateOne({name: 'Barny'}, {$set: {name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'}}, {upsert: true});
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 0,
    upsertedCount: 0
  }
```

Проверить содержимое коллекции unicorns

```
    vampires: 165
  }
  {
    _id: ObjectId("646ea84ccbdb931e0601bf34"),
    name: 'Barny',
    gender: 'm',
    loves: [
      'grape'
    ],
    weight: 340
  }
```

16) Практическое задание 8.2.7:

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

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

Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("646e9b5794597fb029f70fec"),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 800,
  gender: 'f',
  vampires: 51
}
{
```

17) Практическое задание 8.2.8:

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

```
db.unicorns.updateOne({name: 'Raleigh'}, {$set: {loves: ['Redbull']}});
  {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("646e9b5794597fb029f70fee"),
  name: 'Raleigh',
  loves: [
    'Redbull'
  ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
```

18) Практическое задание 8.2.9:

Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}});
<   {
      acknowledged: true,
      insertedId: null,
      matchedCount: 8,
      modifiedCount: 8,
      upsertedCount: 0
    }
```

Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("646e9b5794597fb029f70fed"),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 44
}
```

19) Практическое задание 8.2.10:

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

```
> db.towns.updateOne({name: 'Portland'}, {$unset: {'mayor.party': ""}});
<   {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
learn>
```

Проверить содержимое коллекции towns.

```
  _id: ObjectId("646ea2d094597fb029f70ff5"),
  name: 'Portland',
  populatiuon: 528000,
  last_sensus: 2009-07-20T00:00:00.000Z,
  famous_for: [
    'beer',
    'food'
  ],
  mayor: {
    name: 'Sam Adams'
  }
}
```

20) Практическое задание 8.2.11:

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

```
> db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}});
<   {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
```

Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("646e9b5794597fb029f70ff0"),
  name: 'Pilot',
  loves: [
    'apple',
    'watermelon',
    'chocolate'
  ],
  weight: 650,
  gender: 'm',
  vampires: 59
}
```

21) Практическое задание 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']}}});
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Проверить содержимое коллекции unicorns.

```
{
  _id: ObjectId("646e9b5794597fb029f70fe8"),
  name: 'Aurora',
  loves: [
    'carrot',
    'grape',
    'sugar',
    'lemons'
  ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

22) Практическое задание 8.2.13:

Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.insert({name: "Punxsutawney", popujatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil the groundhog"], mayor: {name: "Jim Wehrle"}});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("646eaba994597fb029f70ff6")
    }
  }
> db.towns.insert({name: "New York", popujatiuon: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["status of liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("646eabc294597fb029f70ff7")
    }
  }
> db.towns.insert({name: "Portland", popujatiuon: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayor: {name: "Sam Adams", party: "D"}});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("646eabd394597fb029f70ff8")
    }
  }
```

Удалите документы с беспартийными мэрами.

```
> db.towns.deleteMany({'mayor.party': null});
< {
    acknowledged: true,
    deletedCount: 3
  }
```

Проверьте содержание коллекции.

```
> db.towns.find();
<   {
      _id: ObjectId("646ea26794597fb029f70ff4"),
      name: 'New York',
      populatiuon: 22200000,
      last_sensus: 2009-07-31T00:00:00.000Z,
      famous_for: [
        'status of liberty',
        'food'
      ],
      mayor: {
        name: 'Michael Bloomberg',
        party: 'I'
      }
    }
    {
      _id: ObjectId("646eabc294597fb029f70ff7"),
      name: 'New York',
      popujatiuon: 22200000,
      last_sensus: 2009-07-31T00:00:00.000Z,
      famous_for: [
        'status of liberty',
        'food'
      ],
      mayor: {
        name: 'Michael Bloomberg',
        party: 'I'
      }
    }
```

Очистите коллекцию.

```
> db.towns.deleteMany({});
<   {
      acknowledged: true,
      deletedCount: 3
    }
```

Просмотрите список доступных коллекций.

```
> show collections;
< towns
  unicorns
```

23) Практическое задание 8.3.1:

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

```
> db.zones.insert({_id: 'desert', full_name: 'Dangerous desert', description: 'Very dangerous desert.'});
< {
    acknowledged: true,
    insertedIds: {
      '0': 'desert'
    }
  }
> db.zones.insert({_id: 'valley', full_name: 'Beautiful valley', description: 'Very beautiful valley.'});
< {
    acknowledged: true,
    insertedIds: {
      '0': 'valley'
    }
  }
> db.zones.insert({_id: 'field', full_name: 'Big field', description: 'Very big field.'});
< {
    acknowledged: true,
    insertedIds: {
      '0': 'field'
    }
  }
```

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

```
> db.unicorns.updateOne({name: 'Ayna'}, {$set: {habitat_zone: {$ref: "zones", $id: "desert"}}});
<   {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
> db.unicorns.updateOne({name: 'Kenny'}, {$set: {habitat_zone: {$ref: "zones", $id: "valley"}}});
<   {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 0,
      upsertedCount: 0
    }
> db.unicorns.updateOne({name: 'Horny'}, {$set: {habitat_zone: {$ref: "zones", $id: "field"}}});
<   {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
```

Проверьте содержание коллекции едиорогов.

```
> db.unicorns.find();
<   {
      _id: ObjectId("646e9b5794597fb029f70fe7"),
      name: 'Horny',
      loves: [
        'carrot',
        'papaya'
      ],
      weight: 600,
      gender: 'm',
      vampires: 73,
      habitat_zone: DBRef("zones", 'field')
    }
    {
      _id: ObjectId("646e9b5794597fb029f70fe8"),
      name: 'Aurora',
```

24) Практическое задание 8.3.2:

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

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

- Можно.

25) Практическое задание 8.3.3:

Получите информацию о всех индексах коллекции unicorns .

```
> 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.

```
> var indexes = db.unicorns.getIndexes();
  indexes.forEach((index) => {
    if (index.name != '_id_') {
      db.unicorns.dropIndex(index.name);
    }
  });
learn>
```

Попытайтесь удалить индекс для идентификатора.

```
> var indexes = db.unicorns.getIndexes();
  indexes.forEach((index) => {
    db.unicorns.dropIndex(index.name);
  });
▸ MongoServerError: cannot drop _id index
learn>
```

26) Практическое задание 8.3.4:

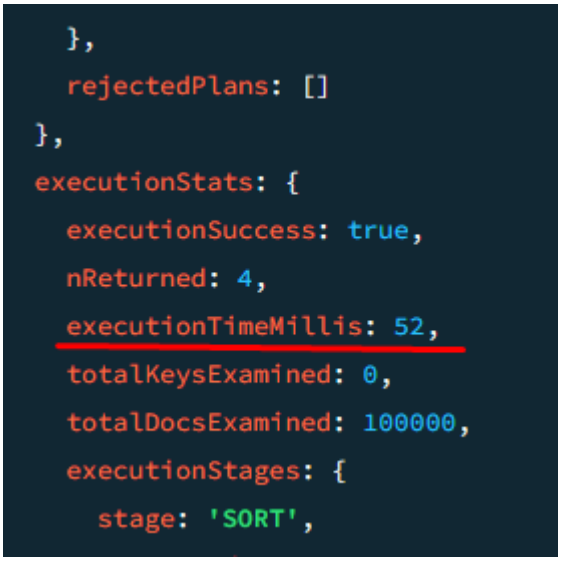
Создайте объемную коллекцию numbers, задействовав курсор

```
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
<   {
      acknowledged: true,
      insertedIds: {
        '0': ObjectId("646eb1c994597fb029f89698")
      }
    }
learn>
```

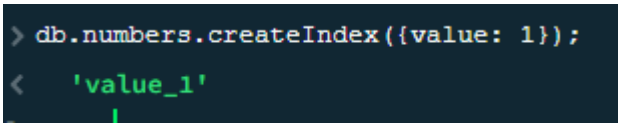
Выберите последних четыре документа.

```
> db.numbers.find().sort({value: -1}).limit(4);
<   {
      _id: ObjectId("646eb1c994597fb029f89698"),
      value: 99999
    }
    {
      _id: ObjectId("646eb1c994597fb029f89697"),
      value: 99998
    }
    {
      _id: ObjectId("646eb1c994597fb029f89696"),
      value: 99997
    }
    {
      _id: ObjectId("646eb1c994597fb029f89695"),
      value: 99996
    }
```

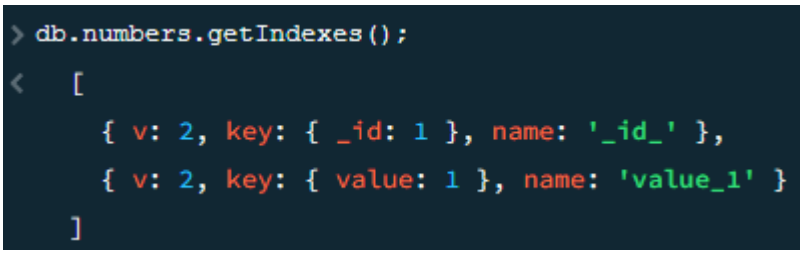
Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis)



Создайте индекс для ключа value.



Получите информацию о всех индексах коллекции numbres.



Выполните запрос 2.

```
> db.numbers.find().sort({value: -1}).limit(4);
<   {
      _id: ObjectId("646eb1c994597fb029f89698"),
      value: 99999
    }
    {
      _id: ObjectId("646eb1c994597fb029f89697"),
      value: 99998
    }
    {
      _id: ObjectId("646eb1c994597fb029f89696"),
      value: 99997
    }
    {
      _id: ObjectId("646eb1c994597fb029f89695"),
      value: 99996
    }
```

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

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

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

52 и <0. Запрос с индексом оказался эффективнее.

Вывод: овладел 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.