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

Отчет

по лабораторной работе №6.2 «Работа с БД в СУБД

MongoDB»по дисциплине **«Проектирование и**

реализация баз данных»

Автор: Морозов А. А.

Факультет: ИКТ

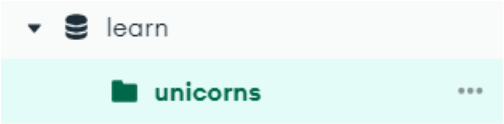
Группа: К3241

Преподаватель: Говорова М. М.

Санкт-Петербург 2023

Практическое задание 2.1.1:

Создайте базу данных learn.



Заполните коллекцию единорогов unicorns:

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

```
>_MONGOSH
  }
> 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'});
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('65a9685a81fef5452ebf2a19')
    }
  }
```

Проверьте содержимое коллекции с помощью метода find.

```
> db.unicorns.find()
< {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a10'),
```

Практическое задание 2.2.1

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find({gender: "m"}).limit(3).sort({name: 1})
< {
    _id: ObjectId('65a9692f81fef5452ebf2a1a'),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a15'),
    name: 'Kenny',
    loves: [
      'grape',
      'lemon'
    ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
```

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('65a9685a81fef5452ebf2a10'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Практическое задание 2.2.2

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

```
> db.unicorns.find({gender: "m"}, {loves: 0, gender: 0}).limit(3).sort({name: 1})
< {
    _id: ObjectId('65a9692f81fef5452ebf2a1a'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a15'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  }
```

Практическое задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural: -1})
< {
    _id: ObjectId('65a9692f81fef5452ebf2a1a'),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a19'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a18'),
```

Практическое задание 2.1.4

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор

```
> 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
  }
```

Практическое задание 2.3.1

Вывести список самок единорогов весом от полтонны 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
  }
  {
```

Практическое задание 2.3.2

```
> db.unicorns.find({gender: "m", weight: {$gte: 500}, love: {$in: ["grape", "lemon"]}}, {_id: 0})
<
```

Практическое задание 2.3.3

```
> db.unicorns.find({vampires: {$exists: false}})
< {
    _id: ObjectId('65a9685a81fef5452ebf2a19'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
```

Практическое задание 2.3.4

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

```
> db.unicorns.find({gender: "m"}, {loves: {$slice: 1}})
< {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    loves: [
      'carrot'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a11'),
    name: 'Unicrom',
    loves: [
      'energon'
    ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
```

Практическое задание 3.1.1

Создайте коллекцию towns, включающую следующие документы

```
> 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"}}
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('65ad62b54f76388cf8a9b9ad'),
      '1': ObjectId('65ad62b54f76388cf8a9b9ae'),
      '2': ObjectId('65ad62b54f76388cf8a9b9af')
    }
  }
```

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"}, {name: 1, mayor: 1})
< {
    _id: ObjectId('65ad62b54f76388cf8a9b9ae'),
    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: 0}}, {name: 1, mayor: 1})
< {
    _id: ObjectId('65ad62b54f76388cf8a9b9ad'),
    name: 'Punxsutawney ',
    mayor: {
      name: 'Jim Wehrle'
    }
  }
```

Практическое задание 3.1.2

Сформировать функцию для вывода списка самцов единорогов

```
> function MaleUnicorns() {return db.unicorns.find({gender: "m"})}
< [Function: MaleUnicorns]
> MaleUnicorns()
< {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
```

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

```
> var UnicornCursor = MaleUnicorns().sort({name: 1}).limit(2)
learn>
```

Вывести результат, используя forEach

```
> UnicornCursor.forEach(function(obj){print(obj.name)})
< Dunx
< Horny
```

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг

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

Практическое задание 3.2.2

Вывести список предпочтений

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

Практическое задание 3.2.3

Посчитать количество особей единорогов обоих полов

```
> db.unicorns.find().count()
< 12
```

Практическое задание 3.3.1

Выполнить команду
Проверить содержимое коллекции `unicorns`

```
  _id: ObjectId('65ad7c744f76388cf8a9b9b0'),
  name: 'Barny',
  loves: [
    'grape'
  ],
  weight: 340,
  gender: 'm'
}
```

Практическое задание 3.3.2

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
  }
```

```
> db.unicorns.find({name:"Ayna"})
< {
    _id: ObjectId('65a9685a81fef5452ebf2a14'),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
```

Практическое задание 3.3.3

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`

```
> db.unicorns.find({name: "Raleigh"})
< {
    _id: ObjectId('65a9685a81fef5452ebf2a16'),
    name: 'Raleigh',
    loves: [
      'redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```

Практическое задание 3.3.4

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
  }
> db.unicorns.findOne({gender: "m"})
< {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
```

Практическое задание 3.3.5

Изменить информацию о городе Портланд: мэр этого города теперь беспартийный

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

Практическое задание 3.3.6

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

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

Практическое задание 3.3.7

Изменить информацию о самке единорога `Aurora`*: теперь она любит еще и сахар, и лимоны*

```
> db.unicorns.updateOne({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.findOne({name: "Aurora"})
< {
    _id: ObjectId('65a9685a81fef5452ebf2a10'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Практическое задание 3.4.1

Создайте коллекцию towns, включающую следующие документы

```
db.towns.insertMany([{name: "Punxsutawney ",
popujatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: ["phil the groundhog"],
mayor: {
   name: "Jim Wehrle"
}}
, {name: "New York",
popujatiuon: 22200000,
last_sensus: ISODate("2009-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
   name: "Michael Bloomberg",
party: "I"}}
, {name: "Portland",
popujatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
party: "D"}}
])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65ad8ce54f76388cf8a9b9b4'),
    '1': ObjectId('65ad8ce54f76388cf8a9b9b5'),
    '2': ObjectId('65ad8ce54f76388cf8a9b9b6')
  }
}
```

Проверьте содержание коллекции

```
> db.towns.find()
< {
    _id: ObjectId('65ad8ce54f76388cf8a9b9b4'),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: 2008-01-31T00:00:00.000Z,
    famous_for: [
      'phil the groundhog'
    ],
    mayor: {
      name: 'Jim Wehrle'
    }
  }
  {
    _id: ObjectId('65ad8ce54f76388cf8a9b9b5'),
    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.remove({})
< {
    acknowledged: true,
    deletedCount: 3
  }
```

Просмотрите список доступных коллекций

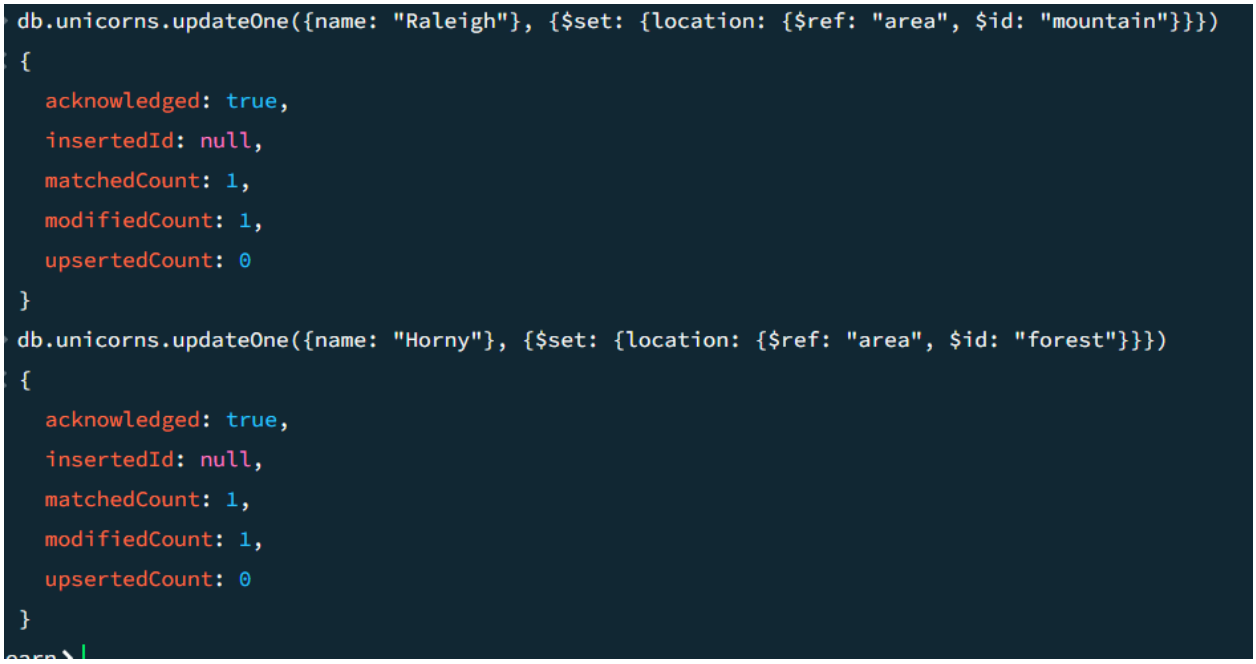
```
> show collections
< towns
  unicorns
```

Практическое задание 4.1.1

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

```
> db.Area.insertMany([{_id: "field", description: "Sunny place"}, {_id: "forest", description: "Misterious place"}, {_id: "mountain", description: "High place"}])
< {
    acknowledged: true,
    insertedIds: {
      '0': 'field',
      '1': 'forest',
      '2': 'mountain'
    }
  }
```

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



```
db.unicorns.updateOne({name: "Raleigh"}, {$set: {location: {$ref: "area", $id: "mountain"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
db.unicorns.updateOne({name: "Horny"}, {$set: {location: {$ref: "area", $id: "forest"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Проверьте содержание коллекции единорогов

```
> db.unicorns.find()
< {
    _id: ObjectId('65a9685a81fef5452ebf2a0f'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    location: DBRef('area', 'forest')
  }
  {
    _id: ObjectId('65a9685a81fef5452ebf2a10'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
```

Практическое задание 4.2.1

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

```
> db.unicorns.createIndex({name: 1}, {unique: true})
< name_1
> db.unicorns.getIndexes()
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
  ]
```

Практическое задание 4.3.1

Получите информацию обо 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

```
> db.unicorns.dropIndexes()
< {
    nIndexesWas: 2,
    msg: 'non-_id indexes dropped for collection',
    ok: 1
  }
```

Попытайтесь удалить индекс для идентификатора

```
> db.unicorns.dropIndex({ _id: 1 })
✖ ▸ MongoServerError: cannot drop _id index
```

Практическое задание 4.4.1

Создайте объемную коллекцию `numbers`, задействовав курсор

```
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
< DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('65afa54436dbd2a19701ab97')
    }
  }
```

Выберите последних четыре документа

```
> db.numbers.find().sort({value: -1}).limit(4)
< {
    _id: ObjectId('65afa54436dbd2a19701ab97'),
    value: 99999
  }
  {
    _id: ObjectId('65afa54436dbd2a19701ab96'),
    value: 99998
  }
  {
    _id: ObjectId('65afa54436dbd2a19701ab95'),
    value: 99997
  }
  {
    _id: ObjectId('65afa54436dbd2a19701ab94'),
    value: 99996
  }
```

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

```
executionTimeMillis: 61,
```

Создайте индекс для ключа `value`

```
> db.numbers.createIndex({value: 1})
< value_1
```

Получите информацию обо всех индексах коллекции `numbres`

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

Выполните запрос 2

```
> db.numbers.find().sort({value: -1}).limit(4)
< {
    _id: ObjectId('65afa54436dbd2a19701ab97'),
    value: 99999
  }
  {
    _id: ObjectId('65afa54436dbd2a19701ab96'),
    value: 99998
  }
  {
    _id: ObjectId('65afa54436dbd2a19701ab95'),
    value: 99997
  }
  {
    _id: ObjectId('65afa54436dbd2a19701ab94'),
    value: 99996
  }
```

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

```
executionTimeMillis: 2,
```

Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?
Более эффективен запрос с индексом (~в 30 раз)

Вывод

В ходе лабораторной работы были изучены основные методы работы с MongoDB.