

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

Отчет

по лабораторной работе №6.2 «Работа с БД в СУБД MongoDB»

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

Автор: Волжева М. И.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

Цель работы	3
Практическое задание 2.1.1	3
Практическое задание 2.2.1	4
Практическое задание 2.2.2:	6
Практическое задание 2.2.3	6
Практическое задание 2.1.4	7
Практическое задание 2.3.1	8
Практическое задание 2.3.2	9
Практическое задание 2.3.3	9
Практическое задание 2.3.4	10
Практическое задание 2.3.4	10
Практическое задание 3.1.1	10
Практическое задание 3.1.2	11
Практическое задание 3.2.1	12
Практическое задание 3.2.2	12
Практическое задание 3.2.3	12
Практическое задание 3.3.1	12
Практическое задание 3.3.2	12
Практическое задание 3.3.3	13
Практическое задание 3.3.4	13
Практическое задание 3.3.5	14
Практическое задание 3.3.6	14
Практическое задание 3.3.7	15
Практическое задание 3.4.1	15
Практическое задание 4.1.1	17
Практическое задание 4.2.1	18
Практическое задание 4.3.1	18
Практическое задание 4.4.1	19
Вывод.....	20

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 2.1.1

- 1) Создайте базу данных learn.
- 2) 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: 'Rooooooodles', 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'});
```

- 3) Используя второй способ, вставьте в коллекцию единорогов документ:

```
{name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
```

```
test> db.unicorns.insert({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657c56c45c98a111516a86bc') }
}
```

- 4) Проверьте содержимое коллекции с помощью метода find.

```
test> db.unicorns.find({"name": "Dunx"})
[
  {
    _id: ObjectId('657c56c45c98a111516a86bc'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

Практическое задание 2.2.1

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

```
test> db.unicorns.find({"gender": "m"}).sort({"name": 1}).limit(3);
[
  {
    _id: ObjectId('657c56c45c98a111516a86bc'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657c53a25c98a111516a86b1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657c53a35c98a111516a86b7'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

```
test> db.unicorns.find({"gender": "f"}).sort({"name": 1}).limit(3);
[
  {
    _id: ObjectId('657c53a25c98a111516a86b2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657c53a25c98a111516a86b6'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657c53a35c98a111516a86b9'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

```
test> db.unicorns.findOne({"gender": "f", "loves": "carrot"});
{
  _id: ObjectId('657c53a25c98a111516a86b2'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
test> db.unicorns.find({"gender": "f", "loves": "carrot"}).limit(1);
[
  {
    _id: ObjectId('657c53a25c98a111516a86b2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 2.2.2:

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

```
test> db.unicorns.find({"gender": "f", "loves": "carrot"}, {loves:0, gender:0}).limit(1);
\[
  {
    _id: ObjectId('657c53a25c98a111516a86b2'),
    name: 'Aurora',
    weight: 450,
    vampires: 43
  }
]
```

Практическое задание 2.2.3

- 1) Вывести список единорогов в обратном порядке добавления.

```
test> db.unicorns.find().sort({ $natural: -1});
[
  {
    _id: ObjectId('657c56c45c98a111516a86bc'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657c53a35c98a111516a86bb'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('657c53a35c98a111516a86ba'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657c53a35c98a111516a86b9'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',

```

Практическое задание 2.1.4

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

```
test> 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
  },
  {
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 88
  }
]
```

Практическое задание 2.3.1

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


```
test> db.unicorns.find({weight: { $gte: 500, $lte: 700 }}, { _id: 0 })
[
  {
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Практическое задание 2.3.2

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

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

Практическое задание 2.3.3

- 1) Найти всех единорогов, не имеющих ключ vampires.

```
test> db.unicorns.find({vampires: {$exists:false}})
[
  {
    _id: ObjectId('657c53a35c98a111516a86bb'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.4

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

```
test> db.unicorns.find({ "gender": "m" }, { name: 1, loves: { $slice: 1 }, _id: 0 }).sort({ name: 1 })
[
  { name: 'Dunx', loves: [ 'grape' ] },
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Rooooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
```

Практическое задание 2.3.4

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

```
test> db.unicorns.find({ "gender": "m" }, {name:1, loves: { $slice: 1 }, _id: 0 }).sort({ name: 1 })
[
  { name: 'Dunx', loves: [ 'grape' ] },
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Rooooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
```

Практическое задание 3.1.1

- 1) Создайте коллекцию towns, включающую следующие документы:

```
{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"}}
```

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

```
test> db.towns.find({"mayor.party": "I"}, {_id: 0, mayor: 1, name: 1})
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

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

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

Практическое задание 3.1.2

- 1) Сформировать функцию для вывода списка самцов единорогов.
- 2) Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.
- 3) Вывести результат, используя forEach.

```
test> fn2 = function() {return db.unicorns.find({gender: "m"})}
[Function: fn2]

test> var cursor = fn2().sort({name: 1}).limit(2);

test> cursor.forEach(function (unicorns) {
... print(unicorns.name);
... });
Dunx
Horny
```

Практическое задание 3.2.1

- 1) Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2

- 1) Вывести список предпочтений.

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

Практическое задание 3.2.3

- 1) Посчитать количество особей единорогов обоих полов.

```
test> db.unicorns.aggregate({"$group":{_id: "$gender", count: {$sum : 1}}})
[ { _id: 'f', count: 5 }, { _id: 'm', count: 8 } ]
```

Практическое задание 3.3.1

Практическое задание 3.3.2

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

- 2) Проверить содержимое коллекции unicorns.

```
test> db.unicorns.replaceOne({ name: "Ayna" }, { name: "Ayna", loves: ["strawberry", "lemon"], weight: 800, gender: "f",
vampires: 51 }, { upsert: true })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}

{
  _id: ObjectId('657c53a25c98a111516a86b6'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
```

Практическое задание 3.3.3

- 1) Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.
2) Проверить содержимое коллекции unicorns.

```
test> db.unicorns.replaceOne({ name: "Raleigh", gender: "m" }, { name: "Raleigh", loves: ['apple', 'sugar', 'redbull'], w
eight: 421, gender: 'm', vampires: 2 }, { upsert: true })
{
  acknowledged: true,
  insertedId: ObjectId('657f14ce41ae529d77d59547'),
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 1
}
test> db.unicorns.find({name: "Raleigh"})
[
  {
    _id: ObjectId('657f14ce41ae529d77d59547'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

Практическое задание 3.3.4

- 1) Всем самцам единорогов увеличить количество убитых вапмиров на 5.

- 2) Проверить содержимое коллекции unicorns.

```
test> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires:5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
test> db.unicorns.find()
[
  {
    _id: ObjectId('657c53a25c98a111516a86b1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
```

Практическое задание 3.3.5

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

```
test> db.towns.update({ name: "Portland" }, { $unset: { "mayor.party": 1 } })
```

- 2) Проверить содержимое коллекции towns.

```
test> db.towns.find({ name: "Portland" })
[
  {
    _id: ObjectId('657eca27e1aed3a41fc1c6d4'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

Практическое задание 3.3.6

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

- 2) Проверить содержимое коллекции unicorns.

```
test> db.unicorns.update({ "name": "Pilot"}, { $push: { "loves": "chocolate" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
test> db.unicorns.find({ "name": "Pilot"})
[
  {
    _id: ObjectId('657c53a35c98a111516a86ba'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

Практическое задание 3.3.7

- 1) Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.
- 2) Проверить содержимое коллекции unicorns.

```
test> db.unicorns.update({name : "Aurora"}, {$addToSet: {loves: {$each: ["lemon", "sugar"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
test> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId('657c53a25c98a111516a86b2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'lemon', 'sugar' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 3.4.1

- 1) Создайте коллекцию towns, включающую следующие документы:

```
test> db.towns.insert({name: "Punxsutawney ", popujatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: "the groundhog"}, mayor: { name: "Jim Wehrle"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f1c4ff19d242fff59ec59') }
}
test> 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('657f1ce2f19d242fff59ec5a') }
}
test> 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('657f1d06f19d242fff59ec5b') }
}
```

- 2) Удалите документы с беспартийными мэрами.

```
test> db.towns.remove({"mayor.party": { $exists:false }})
{ acknowledged: true, deletedCount: 3 }
```

- 3) Проверьте содержание коллекции.

```
test> db.towns.find()
[
  {
    _id: ObjectId('657f1d06f19d242fff59ec5b'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  },
  {
    _id: ObjectId('657f1dc6f19d242fff59ec5c'),
    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' }
  }
]
```

- 4) Очистите коллекцию.

```
test> db.towns.remove({})
{ acknowledged: true, deletedCount: 2 }
```


- 5) Просмотрите список доступных коллекций.

```
test> show collections
towns
unicorns
test> db.towns.find()

test>
```

Практическое задание 4.1.1

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

```
test> db.zones.insert([
...   {
...     _id: "forest",
...     name: "Unicorns' Forest",
...     description: "Special forest for unicorns"
...   },
...   {
...     _id: "mountains",
...     name: "Unicorns' Mountains",
...     description: "Special mountains for unicorns"
...   },
... ])
{
  acknowledged: true,
  insertedIds: { '0': 'forest', '1': 'mountains' }
}
test>
```

- 2) Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания.

```
test> db.unicorns.update({name: 'Horny'}, {$set: {habital: {$ref: 'zones', $id: "forest"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
test> db.unicorns.update({name: 'Aurora'}, {$set: {habital: {$ref: 'zones', $id: "mountains"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

- 3) Проверьте содержание коллекции единорогов.

```
test> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId('657c53a25c98a111516a86b2'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'lemon', 'sugar' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habital: DBRef('zones', 'mountains')
  }
]
test> db.unicorns.find({name: 'Horny'})
[
  {
    _id: ObjectId('657c53a25c98a111516a86b1'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habital: DBRef('zones', 'forest')
  }
]
```

Практическое задание 4.2.1

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

```
test> db.unicorns.ensureIndex({"name" : 1}, {"unique" : true})
[ 'name_1' ]
```

Практическое задание 4.3.1

- 1) Получите информацию о всех индексах коллекции unicorns.

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

- 2) Удалите все индексы, кроме индекса для идентификатора.

```
test> db.unicorns.dropIndex("name_1");
{ nIndexesWas: 2, ok: 1 }
```

- 3) Попробуйте удалить индекс для идентификатора.

```
test> db.unicorns.dropIndex("_id");
MongoServerError: cannot drop _id index
```

Практическое задание 4.4.1

- 1) Создайте объемную коллекцию numbers, задействовав курсор:

```
for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
```

```
test> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657f2757f19d242fff5b86fb') }
}
```

- 2) Выберите последних четыре документа.

```
test> db.numbers.find().sort({value: -1}).limit(4).explain("executionStats")
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 117,
  totalKeysExamined: 0,
  totalDocsExamined: 105119,
```

- 4) Создайте индекс для ключа value.

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

- 5) Получите информацию о всех индексах коллекции numbers.

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

- 6) Выполните запрос 2.

```
test> db.numbers.find().sort({value: -1}).limit(4).explain("executionStats")
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 2,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
```

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

Эффективнее был запрос с индексом, скорость выполнения увеличилась более, чем в 50 раз.

Вывод

В ходе лабораторной работы я овладела 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.