

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Кафедра вычислительных систем

ОТЧЕТ
по лабораторной работе № 6 на тему:
«Работа с БД в СУБД MongoDB»

Выполнил: студент группы К3239

ФИО: Субботин Александр

Станиславович

Проверил: преподаватель М.М. Говорова

Санкт-Петербург
2025

Овладеть практическими навыками работы с CRUD-операциями, вложенными объектами, агрегацией, изменением данных, ссылками и индексами в MongoDB.

Создание базы данных и заполнение коллекции

Практическое задание 2.2.1

Сформируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени. Найдите всех самок, которые любят carrot. Ограничьте этот список первой особью с помощью функций `findOne` и `limit`

Команды MongoDB:

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

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

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

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

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn> db.unicorns.find().sort({name: 1})
[
  {
    _id: ObjectId('6835de7f999b3dce856c4ba7'),
    name: 'Muma',
    loves: [ 'lemon', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 45
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba8'),
    name: 'Anna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 735,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba9'),
    name: 'Anna',
    loves: [ 'grape', 'watermelon' ],
    weight: 104,
    gender: 'm',
    vampires: 105
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba1'),
    name: 'Mump',
    loves: [ 'peach', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 65
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba5'),
    name: 'Mump',
    loves: [ 'grape', 'lemon' ],
    weight: 600,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba6'),
    name: 'Lola',
    loves: [ 'apple', 'watermelon' ],
    weight: 681,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba4'),
    name: 'Mama',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba3'),
    name: 'Lisa',
    loves: [ 'apple', 'watermelon' ],
    weight: 450,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba2'),
    name: 'Malign',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba0'),
    name: 'Momonodias',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba5'),
    name: 'Mama',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f'
  }
]
```

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('6835de7f999b3dce856c4ba7'),
    name: 'Muma',
    loves: [ 'lemon', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 45
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba8'),
    name: 'Anna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 735,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba9'),
    name: 'Lola',
    loves: [ 'apple', 'watermelon' ],
    weight: 681,
    gender: 'f',
    vampires: 33
  }
]
```

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    name: 'Muma',
    loves: [ 'grape', 'carrot' ],
    _id: ObjectId('6835de7f999b3dce856c4ba5'),
    name: 'Mump',
    loves: [ 'grape', 'watermelon' ],
    weight: 104,
    gender: 'm',
    _id: '6835de7f999b3dce856c4ba5',
    vampires: 105a,
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    _id: ObjectId('6835de7f999b3dce856c4ba1'),
    name: 'Mama',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 65
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba7'),
    name: 'Mump',
    loves: [ 'grape', 'lemon' ],
    weight: 600,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba3'),
    name: 'Lisa',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba0'),
    name: 'Momonodias',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('6835de7f999b3dce856c4ba5'),
    name: 'Malign',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

Практическое задание 2.2.2

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

Команды MongoDB:

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

```
]
learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0})
[
  {
    _id: ObjectId('6835de7f599b3dcc056c4bd1'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('6835de7f599b3dcc056c4bd3'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId('6835de7f599b3dcc056c4bd4'),
    name: 'Rooooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('6835de7f599b3dcc056c4bd7'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('6835de7f599b3dcc056c4bd8'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('6835de7f599b3dcc056c4bda'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('6835df56599b3dcc056c4bdc'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
]
```

Практическое задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

Команды MongoDB:

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

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
> use test
> db.unicorns.find().sort({$natural: -1})
[
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d2"),
    "name": "Anna",
    "loves": [ "apple", "watermelon" ],
    "weight": 380,
    "gender": "w",
    "vampires": 155
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d3"),
    "name": "Miaou",
    "loves": [ "apple", "carrot" ],
    "weight": 360,
    "gender": "f",
    "vampires": 9
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d4"),
    "name": "Piliu",
    "loves": [ "apple", "watermelon" ],
    "weight": 450,
    "gender": "w",
    "vampires": 54
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d5"),
    "name": "Jia",
    "loves": [ "apple", "watermelon" ],
    "weight": 681,
    "gender": "f",
    "vampires": 33
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d6"),
    "name": "Miaou",
    "loves": [ "apple", "sugar" ],
    "weight": 421,
    "gender": "w",
    "vampires": 2
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d7"),
    "name": "Jiaou",
    "loves": [ "apple", "lemon" ],
    "weight": 480,
    "gender": "w",
    "vampires": 39
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d8"),
    "name": "Jiaou",
    "loves": [ "strawberry", "lemon" ],
    "weight": 731,
    "gender": "f",
    "vampires": 48
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0d9"),
    "name": "Jiaou",
    "loves": [ "apple", "carrot", "chocolate" ],
    "weight": 550,
    "gender": "f",
    "vampires": 88
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0da"),
    "name": "Miaou",
    "loves": [ "apple" ],
    "weight": 375,
    "gender": "w",
    "vampires": 95
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0db"),
    "name": "Miaou",
    "loves": [ "orange", "redbull" ],
    "weight": 484,
    "gender": "w",
    "vampires": 133
  },
  {
    "_id": ObjectId("6835ae7f990b3cc8564e0dc"),
    "name": "Miaou",
    "loves": [ "carrot", "grape" ],
    "weight": 450,
    "gender": "w",
    "vampires": 133
  }
]
```

Практическое задание 2.2.4

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

Команды MongoDB:

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

```
learn> 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
  },
  {
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Ayna',
    loves: [ 'strawberry' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    name: 'Leia',
    loves: [ 'apple' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    name: 'Nimue', loves: [ 'grape' ], weight: 540, gender: 'f' },
  {
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

Практическое задание 2.3.1

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

Команды MongoDB:

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

```
]
learn> 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'
  }
]
```

Практическое задание 2.3.2

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

Команды MongoDB:

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

```
]
learn> 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
  }
]
```

Практическое задание 2.3.3

Найти всех единорогов, не имеющих ключ vampires.

Команды MongoDB:

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

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId('6835de7f599b3dcc056c4bdb'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Практическое задание 2.3.4

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

Команды MongoDB:

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

```
learn> 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: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Rooooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
```


Практическое задание 3.1.1

Создайте коллекцию towns и выполните выборки по мэрам с party="I" и без party.

Команды MongoDB:

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

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

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6835e266599b3dcc056c4bdd'),
    '1': ObjectId('6835e266599b3dcc056c4bde'),
    '2': ObjectId('6835e266599b3dcc056c4bdf')
  }
}
learn> db.towns.find({"mayor.party": "I"}, {name: 1, mayor: 1, _id: 0});
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn> db.towns.find({"mayor.party": {$exists: false}}, {name: 1, mayor: 1, _id: 0});
[ { name: 'Punxsutawney', mayor: { name: 'Jim Wehrle' } } ]
```

Практическое задание 3.1.2

Сформировать функцию для вывода списка самцов единорогов и вывести первых двух.

Команды MongoDB:

```
var cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2);  
cursor.forEach(function(unicorn) { print(unicorn.name); });
```

```
learn> var cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2);  
learn> cursor.forEach(function(unicorn) { print(unicorn.name); });  
Dunx  
Horny
```

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

Команды MongoDB:

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

Практическое задание 3.2.2

Вывести список предпочтений.

Команды MongoDB:

```
db.unicorns.distinct("loves")
```

```
learn> 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.

Команды MongoDB:

```
db.unicorns.aggregate([{$group: {_id: "$gender", count: {$sum: 1}}}}])
```

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

Практическое задание 3.3.1

Добавить самца Barny.

Команды MongoDB:

```
db.unicorns.save({name: "Barny", loves: ["grape"], weight: 340, gender:
"m"})
```

В моей версии (8.09) команда save была удалена, поэтому:

```
db.unicorns.insertOne({name: "Barny", loves: ["grape"], weight: 340,
gender: "m"})
```

```
learn> db.unicorns.insertOne({name: "Barny", loves: ["grape"], weight: 340,
... gender: "m"})
{
  acknowledged: true,
  insertedId: ObjectId('6835e357599b3dcc056c4be0')
}
```

Практическое задание 3.3.2

Обновить Ayna: вес 800, вампиры 51.

Команды MongoDB:

```
db.unicorns.update({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
```

```
learn> db.unicorns.update({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Практическое задание 3.3.3

Обновить Raleigh: добавить redbull в loves.

Команды MongoDB:

```
db.unicorns.update({name: "Raleigh"}, {$push: {loves: "redbull"}})
```

```
learn> db.unicorns.update({name: "Raleigh"}, {$push: {loves: "redbull"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Практическое задание 3.3.4

Увеличить количество убитых вампиров у всех самцов на 5.

Команды MongoDB:

```
db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true})
```

```
learn> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
```

Практическое задание 3.3.5

Убрать партию у мэра Портланда.

Команды MongoDB:

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

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

Практическое задание 3.3.6

Обновить Pilot: добавить chocolate в loves.

Команды MongoDB:

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

```
learn> db.unicorns.update({name: "Pilot"}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Практическое задание 3.3.7

Обновить Aurora: добавить sugar и lemon в loves.

Команды MongoDB:

```
db.unicorns.update({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
```

```
learn> db.unicorns.update({name: "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Практическое задание 3.4.1

Удалить беспартийных мэров, очистить коллекцию, просмотреть коллекции.

Команды MongoDB:

```
db.towns.remove({"mayor.party": {$exists: false}})
```

```
db.towns.remove({})
```

```
show collections
```

```
learn> db.towns.remove({"mayor.party": {$exists: false}})
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 2 }
learn> db.towns.remove({})
{ acknowledged: true, deletedCount: 1 }
learn> 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. Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания. Проверьте содержание коллекции единорогов.

Команды MongoDB:

```
db.habitats.insert({_id: "nw", name: "Northwest", desc: "Forests and rivers"});

db.habitats.insert({_id: "desert", name: "Desert", desc: "Hot and sandy"});

db.unicorns.update({name: "Horny"}, {$set: {habitat: {$ref: "habitats", $id: "nw"}}});

db.unicorns.update({name: "Aurora"}, {$set: {habitat: {$ref: "habitats", $id: "desert"}}});

db.unicorns.find();
```

```
learn> db.habitats.insert({_id: "nw", name: "Northwest", desc: "Forests and rivers"});
{ acknowledged: true, insertedIds: { '0': 'nw' } }
learn> db.habitats.insert({_id: "desert", name: "Desert", desc: "Hot and sandy"});
{ acknowledged: true, insertedIds: { '0': 'desert' } }
learn> db.unicorns.update({name: "Horny"}, {$set: {habitat: {$ref: "habitats", $id: "nw"}}});
{ acknowledged: true, insertedId: null, matchedCount: 1, modifiedCount: 0, upsertedCount: 0 }
learn> db.unicorns.update({name: "Horny"}, {$set: {habitat: {$ref: "habitats", $id: "nw"}}});
{ acknowledged: true, insertedId: null, matchedCount: 1, modifiedCount: 0, upsertedCount: 0 }
learn> db.unicorns.find();
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd1"),
  "name": "Horny",
  "loves": [ "carrot", "papaya" ],
  "weight": 600,
  "gender": "m",
  "vampires": 60,
  "habitat": DBRef("habitats", "nw")
}
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd2"),
  "name": "Aurora",
  "loves": [ "carrot", "grape", "sugar", "lemon" ],
  "weight": 450,
  "gender": "f",
  "vampires": 45
}
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd3"),
  "name": "Midcrow",
  "loves": [ "mango", "redbull" ],
  "weight": 900,
  "gender": "m",
  "vampires": 107
}
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd4"),
  "name": "Hoooodoodie",
  "loves": [ "apple" ],
  "weight": 375,
  "gender": "m",
  "vampires": 104
}
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd5"),
  "name": "Mileena",
  "loves": [ "apple", "carrot", "chocolate" ],
  "weight": 550,
  "gender": "f",
  "vampires": 80
}
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd6"),
  "name": "Raya",
  "loves": [ "strawberry", "lemon" ],
  "weight": 800,
  "gender": "f",
  "vampires": 51
}
{
  "_id": ObjectId("6835de7f599b3dcc056c4bd7"),
  "name": "Kenny",
  "loves": [ "grape", "lemon" ],

```

Практическое задание 4.2.1

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

Команды MongoDB:

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

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

Практическое задание 4.3.1

Получите информацию о всех индексах коллекции unicorns. Удалите все индексы, кроме индекса для идентификатора. Попытайтесь удалить индекс для идентификатора.

Команды MongoDB:

```
db.unicorns.getIndexes();
db.unicorns.dropIndexes();
db.unicorns.dropIndex("_id_");
```

```
learn> db.unicorns.createIndex({name: 1}, {unique: true})
name_1
learn> db.unicorns.getIndexes();
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
learn> db.unicorns.dropIndexes();
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
learn> db.unicorns.dropIndex("_id_");
MongoServerError[InvalidOptions]: cannot drop _id index
learn>
```


Практическое задание 4.4.1

Создайте объемную коллекцию numbers, задействовав курсор. Выберите последние четыре документа. Проанализируйте план выполнения запроса. Сколько потребовалось времени на выполнение запроса? Создайте индекс для ключа value. Получите информацию о всех индексах коллекции numbers. Выполните запрос 2. Проанализируйте план выполнения запроса с установленным индексом. Сравните время выполнения запросов с индексом и без.

Команды MongoDB:

```
for(i = 0; i < 100000; i++){ db.numbers.insert({value: i}) }  
db.numbers.find().sort({$natural: -1}).limit(4)  
db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)  
db.numbers.createIndex({value: 1})  
db.numbers.getIndexes()  
db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
```

Результат с индексом:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn for (let i = 0; i < 100000; i++) { db.numbers.insert({ value: i }); }
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6835e652599b3dc05701148') }
}
learn db.numbers.find().sort({ $natural: -1 }).limit(4);
{
  _id: ObjectId('6835e652599b3dc05701148'), value: 99999 },
  { _id: ObjectId('6835e652599b3dc05701147'), value: 99998 },
  { _id: ObjectId('6835e652599b3dc05701146'), value: 99997 },
  { _id: ObjectId('6835e652599b3dc05701145'), value: 99996 }
}
learn db.numbers.explain("executionStats").find().sort({ $natural: -1 }).limit(4);
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BF23B3EE',
    planCacheShapeHash: 'BF23B3EE',
    planCacheKey: 'FOR350ET',
    optimizationTimeMillis: 0,
    maxIndexedSolutionsReached: false,
    maxIndexesToSolutionsReached: false,
    maxScansToExploreReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: { stage: 'COLLSCAN', direction: 'backward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 4,
    executionStages: {
      isCached: false,
      stage: 'LIMIT',
      nReturned: 4,
      executionTimeMillisEstimate: 0,
      works: 5,
      advanced: 4,
      needTime: 0,
      needField: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      limitAmount: 4,
      inputStage: {
        stage: 'COLLSCAN',
        nReturned: 4,
        executionTimeMillisEstimate: 0,
        works: 4,
        advanced: 4,
        needTime: 0,
        needField: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 0,
        direction: 'backward',
        docsExamined: 4
      }
    }
  },
  queryShapeHash: '71C04F1800E2018344A2D08A520E31838C280F488D6A5638C7886E8478220E4',
  command: {
    find: 'numbers',
    filter: {},
    sort: { '$natural': -1 },
    limit: 4,
    '$db': 'learn'
  },
  serverInfo: {
    host: 'localhost:27017',
    port: 27017,
    version: '4.0.3',
    gitVersion: 'f8b3e781d51ec7b0593843dc554fde96c416'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857200,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    ...
  }
}
```

Результат без индекса:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn db.numbers.explain("executionStats").find().sort({ $natural: -1 }).limit(4);
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BF23B3EE',
    planCacheShapeHash: 'BF23B3EE',
    planCacheKey: 'FOR350ET',
    optimizationTimeMillis: 0,
    maxIndexedSolutionsReached: false,
    maxIndexesToSolutionsReached: false,
    maxScansToExploreReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: { stage: 'COLLSCAN', direction: 'backward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 4,
    executionStages: {
      isCached: false,
      stage: 'LIMIT',
      nReturned: 4,
      executionTimeMillisEstimate: 0,
      works: 5,
      advanced: 4,
      needTime: 0,
      needField: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      limitAmount: 4,
      inputStage: {
        stage: 'COLLSCAN',
        nReturned: 4,
        executionTimeMillisEstimate: 0,
        works: 4,
        advanced: 4,
        needTime: 0,
        needField: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 0,
        direction: 'backward',
        docsExamined: 4
      }
    }
  },
  queryShapeHash: '71C04F1800E2018344A2D08A520E31838C280F488D6A5638C7886E8478220E4',
  command: {
    find: 'numbers',
    filter: {},
    sort: { '$natural': -1 },
    limit: 4,
    '$db': 'learn'
  },
  serverInfo: {
    host: 'localhost:27017',
    port: 27017,
    version: '4.0.3',
    gitVersion: 'f8b3e781d51ec7b0593843dc554fde96c416'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857200,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    ...
  }
}
```

Без индекса происходит полный просмотр коллекции, который занял 3мс `executionTimeMillis`, а с индексом происходит индексный поиск, который занял менее 1мс.

Выводы

В ходе выполнения лабораторной работы были изучены основные возможности MongoDB: вставка, выборка, изменение и удаление документов (CRUD), работа с вложенными документами, агрегация данных, создание и использование индексов, а также связи между коллекциями. Полученные навыки позволяют эффективно использовать MongoDB для хранения и обработки данных в реальных

