

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

Отчет

по лабораторной работе №5

«Работа с БД в СУБД MongoDB»

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

Автор: Бабичев Л.А.

Факультет: ИКТ

Группа: K32422

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

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:

Вывод:

Практическое задание 8.1.1:

1) *Создайте базу данных learn.*

```
test> db.version()
6.0.6
test> use learn
switched to db learn
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf487dbeda811a881a6e9") }
```


2) *Заполните коллекцию единорогов unicorns:*

```
} 2 | db.unicorns.insertOne({name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63});
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf7b6dbeda811a881a6ec") }
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf7c8dbeda811a881a6ed") }
learn> db.unicorns.insert({name: 'Rooodooles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf7ecdbeda811a881a6ee") }
learn> db.unicorns.insert({name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f', vampires: 80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf852dbeda811a881a6ef") }
learn> db.unicorns.insert({name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf865dbeda811a881a6f0") }
learn> db.unicorns.insert({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf865dbeda811a881a6f1") }
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf865dbeda811a881a6f2") }
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf865dbeda811a881a6f3") }
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf865dbeda811a881a6f4") }
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("646cf865dbeda811a881a6f5") }
```

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

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

4) Проверьте содержимое коллекции с помощью метода *find*.

 mongosh mongod --port 27017 --directConnection=true --serverSelectionTimeoutMS=2000

```
learn> db.unicorns.insertOne({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165});
{
  acknowledged: true,
  insertedId: ObjectId("646cf99adbeda811a881a6f6")
}
learn> db.unicorns.find()
[
  {
    _id: ObjectId("646cf789dbeda811a881a6eb"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("646cf7b6dbeda811a881a6ec"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646cf7c8dbeda811a881a6ed"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("646cf7ecdbeda811a881a6ee"),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("646cf852dbeda811a881a6ef"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f0"),
```

Практическое задание 8.1.2:

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

mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSe

```
learn> db.unicorns.find({gender:'m'}).sort({name:1});
[
  {
    _id: ObjectId("646cf99adbeda811a881a6f6"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("646cf789dbeda811a881a6eb"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f1"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f4"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f2"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("646cf7ecdbeda811a881a6ee"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  }
]
```

```
learn> db.unicorns.find({gender:'f'}).sort({name:1}).limit(3);
[
  {
    _id: ObjectId("646cf7b6dbeda811a881a6ec"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f0"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f3"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn> _
```

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

```
learn> db.unicorns.findOne({loves:'carrot', gender: 'f'}, {name: 1, loves:1});
{typeError: db.unicorns.f ... es:1}).limit is not a function
  _id: ObjectId("646cf7b6dbeda811a881a6ec"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ]
}
learn> db.unicorns.find({loves:'carrot', gender: 'f'}, {name: 1, loves:1}).limit(1);
[
  {
    _id: ObjectId("646cf7b6dbeda811a881a6ec"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ]
  }
]
learn>
```

Практическое задание 8.1.3:

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

```
learn> db.unicorns.find({gender: 'm'}, {loves:0,gender:0});
[
  {
    _id: ObjectId("646cf789dbeda811a881a6eb"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("646cf7c8dbeda811a881a6ed"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("646cf7ecdbeda811a881a6ee"),
    name: 'Rooooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f1"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f2"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f4"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("646cf99adbda811a881a6f6"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
]
```

В этом запросе `{ loves: 0, gender: 0 }` является параметром проекции, где `0` указаны поля, которые должны быть исключены из результата.

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
}
]
learn> db.unicorns.find().sort({ $natural: -1});
[
  {
    _id: ObjectId("646cf99adbeda811a881a6f6"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f5"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f4"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f3"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f2"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId("646cf865dbeda811a881a6f1"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```



В этом запросе `sort({ $natural: -1 })` сортирует документы в порядке, обратном порядку вставки, используя `$natural` параметр сортировки. `-1` значение указывает на порядок убывания.

Практическое задание 8.1.5:

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

```
]
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
  },
]
learn>
```



Поиск



Практическое задание 8.1.6:

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

```
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'  
  }  
]  
learn> █
```

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

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}});
[
  {
    _id: ObjectId("646cf865dbeda811a881a6f5"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

Практическое задание 8.1.9:

```
learn> db.unicorns.find({gender: 'm'}, {loves: {$slice: 1 }, name: true, _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: 'Roooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
learn> _
```

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

Практическое задание 8.2.1:

1) Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.insert({name: "Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: [""], mayor: { name: "Jim Mehrle" }});
{ acknowledged: true, insertedIds: { '0': ObjectId("646d099ddbda811a881a6f7") } }
learn> db.towns.insert({name: "New York", populatiuon: 22200000, last_sensus: ISODate("2009-07-31"), famfamous_for: ["status of liberty", "food"], mayor: { name: "Michael Bloomberg", party: "I" }});
{ acknowledged: true, insertedIds: { '0': ObjectId("646d09d2dbda811a881a6f8") } }
learn> db.towns.insert({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("646d0a96dbda811a881a6f9") } }
learn> _
```

- 2) 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'}, {name: 1, mayor:1, _id:0});
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn>
```

- 3) 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}}, {name: 1, mayor:1, _id:0});
[ { name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } } ]
learn> █
```

Практическое задание 8.2.2:

- 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`.

```
learn> function Male() {db.unicorns.find({ gender:'m'}).forEach(function(unicorn) { print(unicorn.name)});}
[Function: Male]
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 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 8.2.4:

Вывести список предпочтений.

```
learn> db.unicorns.distinct('loves')

'apple',      'carrot',
'chocolate', 'energon',
'grape',      'lemon',
'papaya',     'redbull',
'strawberry', 'sugar',
'watermelon'

learn>
```

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

```

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

```

Практическое задание 8.2.6:

1. Выполнить команду:
`> db.unicorns.save({ name: 'Barney', loves: ['grape'],
weight: 340, gender: 'm'})`
2. Проверить содержимое коллекции *unicorns*.

```

learn> db.unicorns.insertOne({name: 'Barney', loves: ['grape'],
... weight: 340, gender: 'm'});
{
  acknowledged: true,
  insertedId: ObjectId("646d1389dbeda811a881a6fa")
}

```

В MongoDB команда `save()` устарела и рекомендуется использовать команду `insertOne()` для добавления нового документа в коллекцию.

Практическое задание 8.2.7:

1. Для самки единорога Айна внести изменения в БД: теперь ее вес 800, она убила 51 вампира.
2. Проверить содержимое коллекции *unicorns*.

 `mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000`

```

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

```

В MongoDB команда `update()` устарела и рекомендуется использовать команду `updateOne()` для добавления нового документа в коллекцию.

Практическое задание 8.2.8:

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

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

Практическое задание 8.2.9:

1. Всем самцам единорогов увеличить количество убитых вампиров на 5.
2. Проверить содержимое коллекции `unicorn`

mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000

```
learn> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn> db.unicorns.find()
[
  {
    _id: ObjectId("646cf789dbeda811a881a6eb"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("646cf7b6dbeda811a881a6ec"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646cf7c8dbeda811a881a6ed"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("646cf7ecdbeda811a881a6ee"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId("646cf852dbeda811a881a6ef"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
]
```

Практическое задание 8.2.10:

1. Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.
2. Проверить содержимое коллекции towns.

```
learn> db.towns.updateOne({name: 'Portland'}, {$unset: {'mayor.party':1}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find();
[
  {
    _id: ObjectId("646d099ddbda811a881a6f7"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("646d09d2dbda811a881a6f8"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId("646d0a96dbda811a881a6f9"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
learn> _
```

Практическое задание 8.2.11:

1. Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.
2. Проверить содержимое коллекции unicorns.

```

]
learn> db.unicorns.updateOne({name: 'Pilot'}, {$addToSet: {loves: ['chocolate']}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find()
[
  {
    _id: ObjectId("646cf789dbeda811a881a6eb"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("646cf7b6dbeda811a881a6ec"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("646cf7c8dbeda811a881a6ed"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("646cf7ecdbeda811a881a6ee"),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  },
  {
    _id: ObjectId("646cf852dbeda811a881a6ef"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
  }
]

```



Поиск



Практическое задание 8.2.12:

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

```
Ctrl+M mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeo

weight: 550,
gender: 'f',
vampires: 80
},
{
  _id: ObjectId("646cf865dbeda811a881a6f0"),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
{
  _id: ObjectId("646cf865dbeda811a881a6f1"),
  name: 'Kenny',
  loves: [ 'grape', 'lemon' ],
  weight: 690,
  gender: 'm',
  vampires: 44
},
{
  _id: ObjectId("646cf865dbeda811a881a6f2"),
  name: 'Raleigh',
  loves: [ 'redbull' ],
  weight: 421,
  gender: 'm',
  vampires: 7
},
{
  _id: ObjectId("646cf865dbeda811a881a6f3"),
  name: 'Leia',
  loves: [ 'apple', 'watermelon' ],
  weight: 601,
  gender: 'f',
  vampires: 33
},
{
  _id: ObjectId("646cf865dbeda811a881a6f4"),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  weight: 650,
  gender: 'm',
  vampires: 59
},
{
  _id: ObjectId("646cf865dbeda811a881a6f5"),
  name: 'Nimue',
  loves: [ 'grape', 'carrot' ],
  weight: 540,
```



Практическое задание 8.2.13:

- 1) Удалите документы с беспартийными мэрами.
- 2) Проверьте содержание коллекции.

```
arn> db.towns.deleteMany({ 'mayor.party':{ $exists: false }});
acknowledged: true, deletedCount: 2 }
arn> db.towns.find()

{
  _id: ObjectId("646d09d2dbeda811a881a6f8"),
  name: 'New York',
  populatiuon: 22200000,
  last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
  famous_for: [ 'status of liberty', 'food' ],
  mayor: { name: 'Michael Bloomberg', party: 'I' }
}

arn> _
```

- 3) Очистите коллекцию.
- 4) Просмотрите список доступных коллекций.

```
]
learn> db.towns.deleteMany({ 'mayor.party':{ $exists: false }});
{ acknowledged: true, deletedCount: 2 }
learn> db.towns.find()
[
  {
    _id: ObjectId("646d09d2dbeda811a881a6f8"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn> db.towns.deleteMany({});
{ acknowledged: true, deletedCount: 1 }
learn>
towns
unicorns
```

Практическое задание 8.3.1:

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

```
learn> db.habitats.find();
[
  { _id: 'area 1', name: 'area 1', desription: 'big area.' },
  { _id: 'area 2', name: 'area 2', desription: 'small area.' }
]
learn>
```

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

```
learn> db.unicorns.updateMany({}, {$set: { habitat: 'zone1' } });
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 13,
  modifiedCount: 13,
  upsertedCount: 0
}
```

```
learn> db.unicorns.find();
[
  {
    _id: ObjectId("646cf789dbeda811a881a6eb"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habitat: 'zone1'
  },
  {
    _id: ObjectId("646cf7b6dbeda811a881a6ec"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat: 'zone1'
  },
  {
    _id: ObjectId("646cf7c8dbeda811a881a6ed"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187,
    habitat: 'zone1'
  },
  {
    _id: ObjectId("646cf7ecdbeda811a881a6ee"),
    name: 'Rooooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104,
    habitat: 'zone1'
  },
  {
    _id: ObjectId("646cf852dbeda811a881a6ef"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80,
    habitat: 'zone1'
  },
  {
    id: ObjectId("646cf865dbeda811a881a6f0"),

```

Практическое задание 8.3.2:

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

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

Практическое задание 8.3.3:

- 1) Получите информацию о всех индексах коллекции `unicorns`.
- 2) Удалите все индексы, кроме индекса для идентификатора.
- 3) Попробуйте удалить индекс для идентификатора.

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

```
learn> db.unicorns.getIndexes();
[
  { v: 2, key: { _id: 1 }, name: '_id_ },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
learn> db.unicorns.dropIndex('name_1');
{ nIndexesWas: 2, ok: 1 }
learn> db.unicorns.dropIndex('_id_');
MongoServerError: cannot drop _id index
learn> _
```

Практическое задание 8.3.4:

- 1) Создайте объемную коллекцию `numbers`, задействовав курсор:

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

```
Stopping execution...
learn> for (i = 0; i < 100000; i++) {db.numbers.insertOne({value: i});}
{
  acknowledged: true,
  insertedId: ObjectId("646dc321dbeda811a8867c50")
}
learn> db.numbers.find().sort({_id: -1}).limit(4);
[
  { _id: ObjectId("646dc321dbeda811a8867c50"), value: 99999 },
  { _id: ObjectId("646dc321dbeda811a8867c4f"), value: 99998 },
  { _id: ObjectId("646dc321dbeda811a8867c4e"), value: 99997 },
  { _id: ObjectId("646dc321dbeda811a8867c4d"), value: 99996 }
]
learn> _
```

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

105 ms

4) Создайте индекс для ключа *value*.

```
    }
  },
  rejectedPlans: []
},
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 105,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'LIMIT',
    nReturned: 4,
    executionTimeMillisEstimate:
learn> db.numbers.explain('executionStats').find().sort({_id: -1}).limit(1)

explainVersion: '1', dc321dbeda811a8867c50", value: 99999 },
queryPlanner: {"646dc321dbeda811a8867c4f", value: 99998 },
namespace: 'learn.numbers', a811a8867c4e", value: 99997 },
indexFilterSet: false, 1dbeda811a8867c4d", value: 99996 }
parsedQuery: {},
queryHash: '51B6F510',
planCacheKey: '51B6F510',
maxIndexedOrSolutionsReached: false,
maxIndexedAndSolutionsReached: false,
maxScansToExplodeReached: false,
winningPlan: {
  stage: 'LIMIT',
  limitAmount: 4,
  inputStage: {
    stage: 'FETCH',
    inputStage: {
      stage: 'IXSCAN',
      keyPattern: { _id: 1 },
      indexName: '_id_',
      isMultiKey: false,
      multiKeyPaths: { _id: [] },
      isUnique: true,
      isSparse: false,
      isPartial: false,
      indexVersion: 2,
      direction: 'backward',
      indexBounds: { _id: [ '[MaxKey, MinKey]' ] }
    }
  }
},
rejectedPlans: []
},
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 0,
  totalKeysExamined: 4,
  totalDocsExamined: 4,
  executionStages: {
    stage: 'LIMIT',
    nReturned: 4,
    executionTimeMillisEstimate: 0,
    works: 5,
    advanced: 4,
    needTime: 0,
```

- 5) Получите информацию о всех индексах коллекции *nombres*.
- 6) Выполните запрос 2.
- 7) Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

0 ms

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

Запрос с индексацией обычно выполняется быстрее, потому что индекс позволяет базе данных эффективно найти и выбрать нужные документы, минимизируя количество проверок и сравнений.

Когда индекс не используется, СУБД должна просмотреть все документы в коллекции и выполнить полное сканирование (*full scan*), чтобы найти и выбрать нужные документы. Это требует больше времени и ресурсов, особенно при большом объеме данных.

Вывод: в ходе данной лабораторной работе были приобретены практические навыки работы с базой данных MongoDB, используя интерфейс DataGrip. Основные задачи, с которыми я столкнулся, включали создание, чтение, обновление и удаление данных (CRUD операции), работу с вложенными объектами в коллекциях, выполнение агрегации и изменение данных.