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 «**Базы данных**»

Автор: Демша Евгения

Факультет: ИКТ

Группа: К32422

Преподаватель: Говорова М.М.

Санкт-Петербург, 2023

Цель: 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.

Задание 1.1:

Создайте базу данных learn.
Заполните коллекцию единорогов unicorns:

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

Используя второй способ, вставьте в коллекцию единорогов документ:

```
> document = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165};
< {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
> db.unicorns.insertOne(document);
< {
    acknowledged: true,
    insertedId: ObjectId("647432df30951bc247d6a147")
  }
```

Проверьте содержимое коллекции с помощью метода find.

```
> db.unicorns.find()
< {
    _id: ObjectId("64742f1130951bc247d6a13a"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId("6474317f30951bc247d6a13b"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
```

Задание 1.2:

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

```
> db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3);
< {
    _id: ObjectId("6474317f30951bc247d6a13c"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("6474317f30951bc247d6a140"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
    _id: ObjectId("6474317f30951bc247d6a143"),
    name: 'Leia',
    loves: [
```

```
> db.unicorns.find({gender: "m"}).sort({name: 1}).limit(3);
< {
    _id: ObjectId("647435dc30951bc247d6a148"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId("647435dc30951bc247d6a14e"),
    name: 'Kenny',
    loves: [
      'grape',
      'lemon'
    ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
  {
    _id: ObjectId("647435dc30951bc247d6a151"),
    name: 'Pilot',
```

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

```
> db.unicorns.findOne({ loves: "carrot"});
< {
    _id: ObjectId("647435dc30951bc247d6a148"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
learn>
```

Задание 1.3:

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

```
> db.unicorns.find({gender: "m"},{"loves": 0, "gender": 0}).sort({name: 1}).limit(3);
< {
    _id: ObjectId("64743e2d30951bc247d6a153"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
  {
    _id: ObjectId("647435dc30951bc247d6a148"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
  {
    _id: ObjectId("647435dc30951bc247d6a14e"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  }
learn>
```

Задание 1.4:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({ $natural: -1 });
< {
    _id: ObjectId("64743e2d30951bc247d6a153"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("647435dc30951bc247d6a152"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("647435dc30951bc247d6a151"),
    name: 'Pilot',
    loves: [
      'apple',
```

Практическое задание 1.5:

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

```
> db.unicorns.find ({}, {loves: {$slice : 1},_id: false});
< {
    name: 'Horny',
    loves: [
      'carrot'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    name: 'Aurora',
    loves: [
      'carrot'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    name: 'Unicrom',
    loves: [
      'energon'
    ],
```

Задание 1.6:

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

```
> db.unicorns.find ({weight: {$gt : 500, $lt: 700}, gender: "f"},{_id: false});
< {
    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'
```

Задание 1.7:

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

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

Задание 1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
> db.unicorns.find ({vampires: {$exists:false}});
< {
    _id: ObjectId("647435dc30951bc247d6a152"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
learn>
```

Задание 1.9:

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

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

Задание 2.1

Создайте коллекцию towns:

```
> 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,
```

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:0});
< {
    name: 'New York',
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
learn>
```

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

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

Задание 2.2:

Сформировать функцию для вывода списка самцов единорогов.

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

Вывести результат, используя forEach.

```
> var cursor = db.unicorns.find({gender: 'm'}); null;
< null
> cursor.sort({name:1}).limit(3);null;
< null
> cursor.forEach(function(obj){
  print(obj.name);
  })
< Dunx
< Horny
< Kenny
learn>
```

Задание **2.3:**

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Задание 2.4:

Вывести список предпочтений.

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

Задание 2.5:

Посчитать количество особей единорогов обоих полов.

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

Задание 2.6:

Выполнить команду:

> db.unicorns.save({name: 'Barny', loves: ['grape'],

weight: 340, gender: 'm'})

Проверить содержимое коллекции unicorns.

```
> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
< {
    acknowledged: true,
    insertedId: ObjectId("6474649e323909a8cfeefe9b")
  }
> db.unicorns.find().sort({ $natural: -1 })
< {
    _id: ObjectId("6474649e323909a8cfeefe9b"),
    name: 'Barny',
    loves: [
      'grape'
    ],
    weight: 340,
    gender: 'm'
  }
  {
    _id: ObjectId("64743e2d30951bc247d6a153"),
    name: 'Dunx',
```

Задание 2.7:

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

Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateOne({ name : "Ayna" },{$set:{ name : "Ayna",weight: 800, vampires: 51}},{ upsert: true } )
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: "Ayna"})
< {
    _id: ObjectId("647435dc30951bc247d6a14d"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
```

Задание 2.8:

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

Проверить содержимое коллекции unicorns.

Задание 2.9:

Всем самцам единорогов увеличить количество убитых вапмиров на 5.

Проверить содержимое коллекции unicorns.

```
> db.unicorns.find({},{_id:0,loves:0,weight:0})
< {
    name: 'Horny',
    gender: 'm',
    vampires: 63
  }
  {
    name: 'Aurora',
    gender: 'f',
    vampires: 43
  }
  {
    name: 'Unicrom',
    gender: 'm',
    vampires: 182
  }
  {
    name: 'Roooooodles',
    gender: 'm',
    vampires: 99
  }
  {
    name: 'Solnara',
    gender: 'f',
    vampires: 80
  }
```

```
> db.unicorns.update({gender : "m"}, {$inc: {vampires:5}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

```
> db.unicorns.find({},{_id:0,loves:0,weight:0})
< {
    name: 'Horny',
    gender: 'm',
    vampires: 68
  }
  {
    name: 'Aurora',
    gender: 'f',
    vampires: 43
  }
  {
    name: 'Unicrom',
    gender: 'm',
    vampires: 182
  }
  {
    name: 'Roooooodles',
    gender: 'm',
    vampires: 99
  }
  {
    name: 'Solnara',
    gender: 'f',
    vampires: 80
  }
```

Задание 2.10:

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

```
> db.towns.update({name : "Portland"}, {$unset: {"mayor.party": 1}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.towns.find({name : "Portland"})
< {
    _id: ObjectId("6474551a30951bc247d6a156"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams'
    }
  }
learn>
```

Задание 2.11:

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

```
> db.unicorns.update({name : "Pilot"}, {$push: {loves: "chocolate"}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name : "Pilot"})
< {
    _id: ObjectId("647435dc30951bc247d6a151"),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon',
      'chocolate'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
learn>
```

Задание 2.12:

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

```
> db.unicorns.update({name : "Aurora"},{$addToSet: {loves: {$each: ["sugar", "lemons"]}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name : "Aurora"})
< {
    _id: ObjectId("647435dc30951bc247d6a149"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemons'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
learn>
```

Задание 2.13:

Удалите документы с беспартийными мэрами.

```
> db.towns.remove({"mayor.party": {$exists:false}})
< DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
< {
    acknowledged: true,
    deletedCount: 1
  }
```

Проверьте содержание коллекции.

```
> db.towns.find()
< {
    _id: ObjectId("6474551a30951bc247d6a155"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: 2009-07-31T00:00:00.000Z,
    famous_for: [
      'status of liberty',
      'food'
    ],
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
  {
    _id: ObjectId("6474551a30951bc247d6a156"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams',
      party: 'D'
    }
  }
```

Очистите коллекцию.
Просмотрите список доступных коллекций.

```
> db.towns.drop()
< true
> show dbs
< admin    40.00 KiB
  config  108.00 KiB
  learn    72.00 KiB
  local    40.00 KiB
> show collections
< unicorns
learn>
```

Задание 3.1:

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

```
{
  _id: ObjectId("647821b94a9ce524dcc51730"),
  id: 'valley',
  name: 'Rainbow Valley',
  description: 'This habitat area is characterized by lush green meadows, sparkling streams, an
}
{
  _id: ObjectId("647821b94a9ce524dcc51731"),
  id: 'cave',
  name: 'Crystal Caverns',
  description: 'Located deep underground, this habitat area is home to unicorns that have adapt
}
{
  _id: ObjectId("647821b94a9ce524dcc51732"),
  id: 'forest',
  name: 'Enchanted Forest',
  description: 'This habitat area is filled with towering trees, babbling brooks, and mystical
}
{
  _id: ObjectId("647821b94a9ce524dcc51733"),
  id: 'meadow',
  name: 'Starry Sky Meadows',
  description: 'This habitat area is located on a high plateau, where the air is thin and the s
}
{
  _id: ObjectId("647821b94a9ce524dcc51734"),
  id: 'ocean',
  name: " Ocean's Edge",
```

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

```
  }
> db.unicorns.updateOne({name: "Aurora"},{$set:{habitat:{$ref:"habitats", $id: "meadow"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 0,
    upsertedCount: 0
  }
> db.unicorns.updateOne({name: "Roooooodles"},{$set:{habitat:{$ref:"habitats", $id: "valley"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.updateOne({name: "Nimue"},{$set:{habitat:{$ref:"habitats", $id: "ocean"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.updateOne({name: "Raleigh"},{$set:{habitat:{$ref:"habitats", $id: "forest"}}})
< {
    acknowledged: true,
```

Проверьте содержание коллекции единорогов.

```
> db.unicorns.find({habitat: {$exists:true}},{name:1, habitat:1, _id:0})
< {
    name: 'Aurora',
    habitat: DBRef("habitats", 'meadow')
  }
  {
    name: 'Roooooodles',
    habitat: DBRef("habitats", 'valley')
  }
  {
    name: 'Raleigh',
    habitat: DBRef("habitats", 'forest')
  }
  {
    name: 'Nimue',
    habitat: DBRef("habitats", 'ocean')
  }
learn>
```

Задание 3.2:

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

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

Можно

Задание 3.3:

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

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

Задание 3.4:

1. Создайте объемную коллекцию numbers, задействовав курсор:

```
> for(i = 0; i < 100000; i++){db.numbers.insertOne({value: i})}
< {
    acknowledged: true,
    insertedId: ObjectId("6478347f774985c7ccbe5c63")
  }
learn>
```

2. Выберите последних четыре документа.

```
> db.numbers.find().sort({$natural: -1 }).limit(4)
< {
    _id: ObjectId("6478347f774985c7ccbe5c63"),
    value: 99999
  }
  {
    _id: ObjectId("6478347f774985c7ccbe5c62"),
    value: 99998
  }
  {
    _id: ObjectId("6478347f774985c7ccbe5c61"),
    value: 99997
  }
  {
    _id: ObjectId("6478347f774985c7ccbe5c60"),
    value: 99996
  }
learn>
```

3. Проанализируйте план выполнения запроса

4. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis)

```
executionTimeMillis: 0,
```

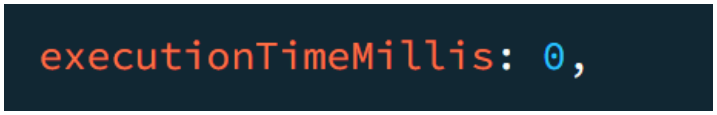
5. Создайте индекс для ключа value.

6. Получите информацию о всех индексах коллекции numbres.

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

7. Выполните запрос 2.

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



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

При запросе в коллекцию из десяти тысяч документов, содержащих одно поле, индекс не дал заметного улучшения. Однако ясно, что по проиндексированному полю эффективнее и быстрее.

Выводы:

NoSQL-субд пользоваться очень удобно и весело, мне понравилось!