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

Отчет

по Лабораторной Работе № 5 по дисциплине

«**Базы Данных**»

Автор: Сахно Ярослав Александрович

Факультет: ФИКТ

Группа: K3241

Преподаватель: Говорова Марина Михайловна

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.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4+, 6.0.6 (текущая).

Практическое задание 2.1.1:

1) *Создайте базу данных learn.*
2) *Заполните коллекцию единорогов unicorns.*
3) *Используя второй способ, вставьте в коллекцию единорогов документ.*
4) *Проверьте содержимое коллекции с помощью метода find.*

```
use learn



db.unicorns.insertOne({name:           'Horny',           loves:
['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insertOne({name:    'Aurora',    loves:    ['carrot',
'grape'], weight: 450, gender: 'f', vampires: 43});

db.unicorns.insertOne({name:   'Unicrom',   loves:   ['energon',
'redbull'], weight: 984, gender: 'm', vampires: 182});

db.unicorns.insertOne({name:  'Roooooodles',  loves:  ['apple'],
weight: 575, gender: 'm', vampires: 99});

db.unicorns.insertOne({name:      'Solnara',      loves:['apple',
'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

db.unicorns.insertOne({name:'Ayna',     loves:     ['strawberry',
'lemon'], weight: 733, gender: 'f', vampires: 40});

db.unicorns.insertOne({name:'Kenny', loves: ['grape', 'lemon'],
weight: 690,  gender: 'm', vampires: 39});

db.unicorns.insertOne({name:    'Raleigh',    loves:    ['apple',
'sugar'], weight: 421, gender: 'm', vampires: 2});

db.unicorns.insertOne({name:     'Leia',     loves:     ['apple',
'watermelon'], weight: 601, gender: 'f', vampires: 33});

db.unicorns.insertOne({name:     'Pilot',     loves:     ['apple',
'watermelon'], weight: 650, gender: 'm', vampires: 54});

db.unicorns.insertOne({name:     'Nimue',     loves:     ['grape',
'carrot'], weight: 540, gender: 'f'});
```

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


db.unicorns.insertOne(doc)


db.unicorns.find()
```

Практическое задание 2.2.1:

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

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

```
db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3)
db.unicorns.find({gender: "m"}).sort({name: 1}).limit(3)
db.unicorns.findOne({loves: "carrot", gender: "f"})
```

Практическое задание 2.2.2:

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

```
db.unicorns.find({gender: "m"},{loves: 0}).sort({name: 1,}).limit(3)
```

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

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

Практическое задание 2.1.4:

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

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

Практическое задание 2.3.1:

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

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

Практическое задание 2.3.2:

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

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

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.

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

Практическое задание 2.3.4:

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

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

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

```
db.towns.insertMany([
   {
       name: "Punxsutawney",
       population: 6200,
       last_sensus: ISODate("2008-01-31"),
       famous_for: [],
       mayor: {
           name: "Jim Wehrle"
```

```
        }
    },
    {
        name: "New York",
        population: 22200000,
        last_sensus: ISODate("2009-07-31"),
        famous_for: ["Statue of Liberty", "food"],
        mayor: {
            name: "Michael Bloomberg",
            party: "I"
        }
    },
    {
        name: "Portland",
        population: 528000,
        last_sensus: ISODate("2009-07-20"),
        famous_for: ["beer", "food"],
        mayor: {
            name: "Sam Adams",
            party: "D"
        }
    }
]);


db.towns.find({"mayor.party": "I"}, {name: 1, "mayor.name": 1,
_id: 0})
db.towns.find({ "mayor.party": { $exists: false } }, { name: 1,
"mayor.name": 1, _id: 0 })
```

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.*

```
function printMaleUnicorns() {
    var cursor = db.unicorns.find({ gender: "m" });null;
    cursor.sort({ name: 1 }).limit(2);null;
```

```
    cursor.forEach(function(unicorn) {
        print(unicorn.name);
    });
}
printMaleUnicorns()
```

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2:

Вывести список предпочтений.

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

Практическое задание 3.2.3:

Подсчитать количество особей единорогов обоих полов.

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

Практическое задание 3.3.2:

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

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

Практическое задание 3.3.3:

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

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

Практическое задание 3.3.4:

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

```
db.unicorns.updateMany({gender: "m"}, {$inc: {kills: 5}})
db.unicorns.find()
```

Практическое задание 3.3.5:

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

```
db.towns.update({name: "Portland"}, {$set: {mayor: "I"}})
db.towns.find()
```

Практическое задание 3.3.6:

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

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

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`.

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

Практическое задание 3.4.1:

1) Создайте коллекцию towns, включающую следующие документы:

```
{name: "Punxsutawney ",
popujatiuon: 6200,
last_sensus: ISODate("2008-01-31"),
famous_for: ["phil the groundhog"],
mayor: {
   name: "Jim Wehrle"
   }}

{name: "New York",
popujatiuon: 22200000,
last_sensus: ISODate("2009-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
   name: "Michael Bloomberg",
   party: "I"}}

{name: "Portland",
popujatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
   party: "D"}}
```

2) *Удалите документы с беспартийными мэрами.*
3) *Проверьте содержание коллекции.*
4) *Очистите коллекцию.*
5) *Просмотрите список доступных коллекций.*

```
db.towns.insertMany([
    {
        name: "Punxsutawney",
        population: 6200,
        last_sensus: new Date("2008-01-31"),
        famous_for: ["Phil the groundhog"],
        mayor: {
            name: "Jim Wehrle"
        }
    },
    {
        name: "New York",
        population: 22200000,
        last_sensus: new Date("2009-07-31"),
        famous_for: ["Statue of Liberty", "food"],
        mayor: {
            name: "Michael Bloomberg",
            party: "I"
        }
    },
    {
        name: "Portland",
        population: 528000,
        last_sensus: new Date("2009-07-20"),
        famous_for: ["beer", "food"],
        mayor: {
            name: "Sam Adams",
            party: "D"
```

```
        }
    }
])

db.towns.deleteMany({ "mayor.party": { $exists: false } })
db.towns.find()
db.towns.deleteMany({})
show collections
```

Практическое задание 4.1.1:

1) *Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.*
2) *Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания.*
3) *Проверьте содержание коллекции единорогов.*

```
db.zones.insert([
    {
        _id: "forest",
        name: "Magic Forest",
        description: "A mystical forest where unicorns live peacefully."
    },
    {
        _id: "meadow",
        name: "Enchanted Meadow",
        description: "A beautiful meadow where unicorns roam and play."
    },
    {
        _id: "mountain",
        name: "Majestic Mountain",
```

```
       description: "A grand mountain where unicorns graze and
rest."
   }
])

var forestZoneId = db.zones.findOne({_id: "forest"})._id;
var meadowZoneId = db.zones.findOne({_id: "meadow"})._id;
var mountainZoneId = db.zones.findOne({_id: "mountain"})._id;

db.unicorns.updateMany({}, [
   { $set: { habitat: { $ref: "zones", $id: forestZoneId }}},
   { $set: { habitat: { $ref: "zones", $id: meadowZoneId }}},
   { $set: { habitat: { $ref: "zones", $id: mountainZoneId }}}
])
```

Практическое задание 4.2.1:

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

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

Практическое задание 4.3.1:

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

```
db.unicorns.getIndexes()
db.unicorns.dropIndexes()
```

Практическое задание 4.4.1:

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

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

2) *Выберите последних четыре документа.*
3) *Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра* `executionTimeMillis`*)*
4) *Создайте индекс для ключа* `value`.
5) *Получите информацию о всех индексах коллекции* `numbers`.
6) *Выполните запрос 2.*

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

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

```
for(i = 0; i < 100000; i++){

   db.numbers.insert({value: i})

}

db.numbers.find().sort({value:-1}).limit(4).explain("executionS
tats")

db.numbers.createIndex({value: 1})

db.numbers.getIndexes()

db.numbers.find().sort({value:-1}).limit(4).explain("executionS
tats")


//До создания индекса - 56, после - 4. Эффективнее с индексом
```

Вывод: В результате выполнения данной лабораторной работы я овладел практическими навыками работы с CRUD-операциями в базе данных MongoDB. Я освоил работу с вложенными объектами в коллекциях, осуществлял агрегации и изменения данных, а также работал со ссылками и индексами в базе данных MongoDB. Этот опыт поможет мне в дальнейшей работе с базами данных и разработке программных продуктов, где требуется использование MongoDB.