Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

Лабораторная работа №5
«Работа с БД в СУБД MongoDB»

Выполнила:
студентка II курса ИКТ
группы К3243
Махнева Анастасия Денисовна

Проверила:
Говорова Марина Михайловна

Санкт-Петербург
2022

Цель лабораторной 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.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 5.0.8.

Выполнение:

Задание 8.1.1:

1) Создайте базу данных learn.
2) Заполните коллекцию единорогов unicorns.
3) Используя второй способ, вставьте в коллекцию единорогов документ.

```
> use learn
switched to db learn
> db.createCollection("unicorns")
{ "ok" : 1 }
> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
WriteResult({ "nInserted" : 1 })
> doc=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
{
        "name" : "Dunx",
        "loves" : [
                "grape",
                "watermelon"
        ],
        "weight" : 704,
        "gender" : "m",
        "vampires" : 165
}
> db.unicorns.insert(doc)
WriteResult({ "nInserted" : 1 })
```

Рисунок 1 – Создание и наполнение данными БД learn

4) Проверьте содержимое коллекции с помощью метода find.

Рисунок 2 – Результат проверки содержимого коллекции unicorns с помощью метода find

Задание 8.1.2:

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

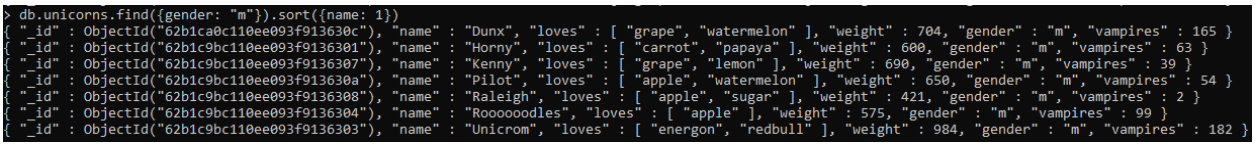

Рисунок 3 – Список самцов, отсортированный по имени

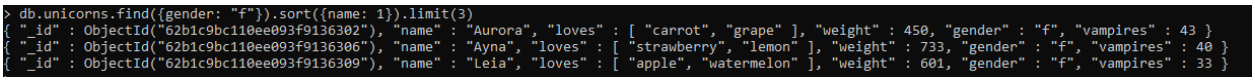

Рисунок 4 – Первые три элемента списка самок, отсортированного по имени

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

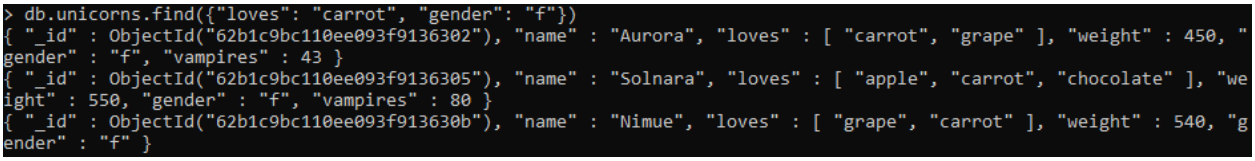

Рисунок 5 – Список всех самок, которые любят carrot

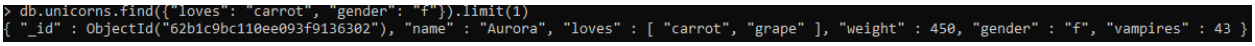

Рисунок 6 – Первая особь из списка самок, которые любят carrot с ограничением limit

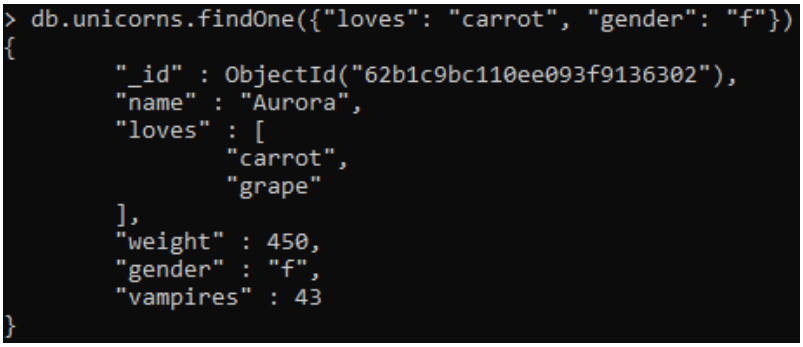

Рисунок 7 – Поиск одной самки, любящей carrot, с ограничением findOne

Задание 8.1.3: модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпочтениях и поле.

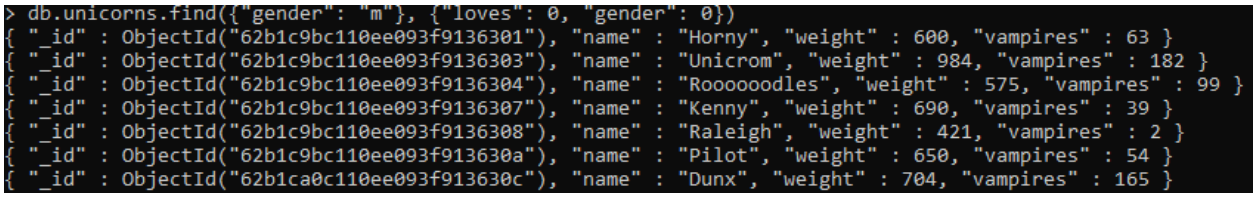

Рисунок 8 – Модифицированный запрос

Задание 8.1.4: вывести список единорогов в обратном порядке добавления.

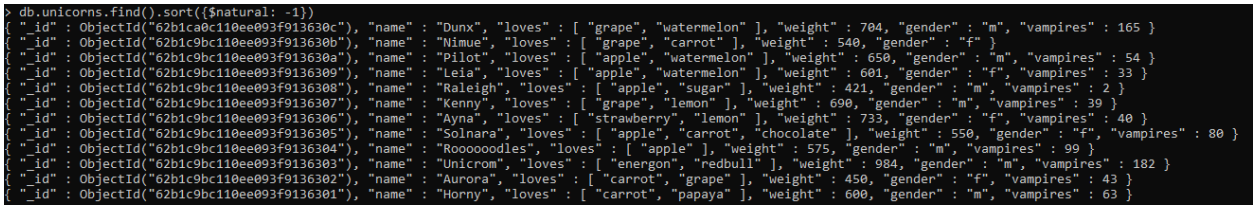

Рисунок 9 – Список единорогов в обратном порядке добавления

Задание 8.1.5: вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор.

Рисунок 10 – Список единорогов с названием первого любимого предпочтения

Задание 8.1.6: вывести список самок единорогов весом от полутонны до 700 кг, исключив вывод идентификатора.

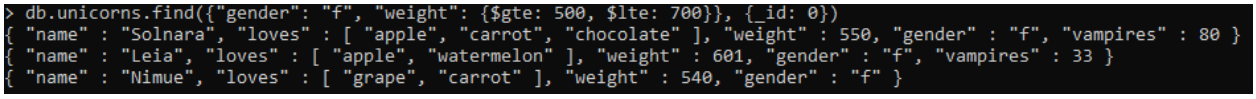

Рисунок 11 – Список самок единорогов весом от 500 до 700 кг

Задание 8.1.7: вывести список самцов единорогов весом от полутонны и предпочитающих grape и lemon, исключив вывод идентификатора.

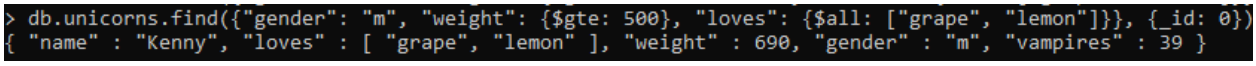

Рисунок 12 – Список самцов единорогов весом от полутонны и предпочитающих grape и lemon

Задание 8.1.8: найти всех единорогов, не имеющих ключ vampires.

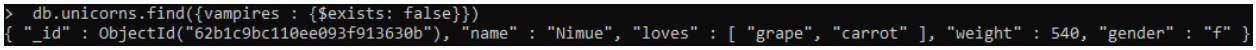

Рисунок 13 – Единороги без ключа vampires

Задание 8.1.9: вывести упорядоченный список имен самцов единорогов с информацией об их первом предпочтении.

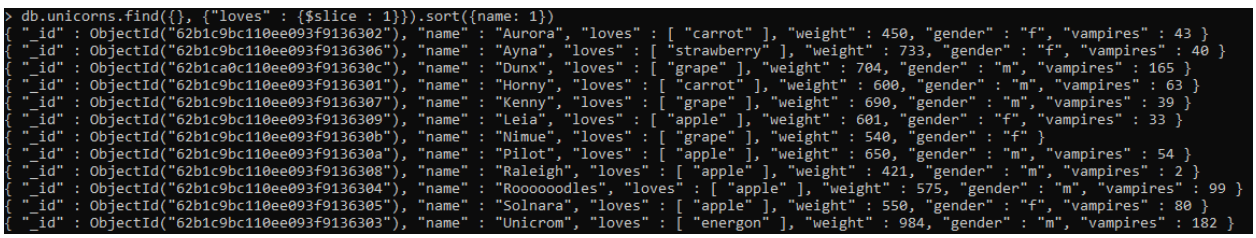

Рисунок 14 – упорядоченный список имен самцов с информацией об их первом предпочтении

Задание 8.2.1:

1) Создайте коллекцию towns, включающую некоторые документы.

```
> db.createCollection("towns")
{ "ok" : 1 }
> doc1={name: "Punxsutawney ",
... population: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...     name: "Jim Wehrle"
... }}
{
        "name" : "Punxsutawney ",
        "population" : 6200,
        "last_sensus" : ISODate("2008-01-31T00:00:00Z"),
        "famous_for" : [
                ""
        ],
        "mayor" : {
                "name" : "Jim Wehrle"
        }
}
> db.towns.insert(doc1)
WriteResult({ "nInserted" : 1 })
```

Рисунок 15 – Часть скрипта создания коллекции towns с документами из текста лабораторной работы

2) 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"}, {_id: 0, "population": 0, "last_sensus": 0, "famous_for": 0})
{ "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
```

Рисунок 16 – Результат выполнения запроса

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

```
> db.towns.find({"mayor.party": {$exists: false}}, {_id: 0 , population: 0, last_sensus: 0, famous_for: 0})
{ "name" : "Punxsutawney ", "mayor" : { "name" : "Jim Wehrle" } }
```

Рисунок 17 – Результат выполнения запроса

Задание 8.2.2:

1) Сформировать функцию для вывода списка самцов единорогов.

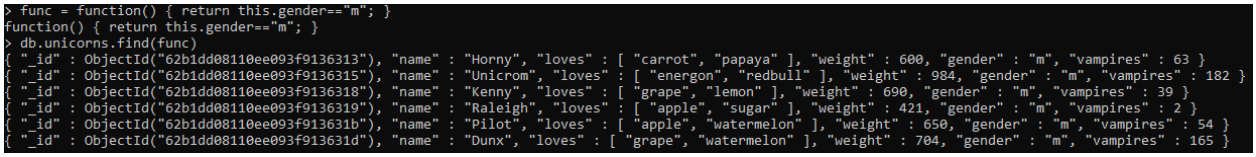

Рисунок 18 – Функция и результат ее работы

2) Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке.
3) Вывести результат, используя forEach.

```
> var cursor = db.unicorns.find(func);null;
null
> cursor.limit(2);null;
null
> cursor.sort({name:1});null;
null
> cursor.forEach(function(obj) {print(obj.name);})
Dunx
Horny
```

Рисунок 19 – Созданный курсор и вывод результата с использованием forEach

Задание 8.2.3: вывести количество самок единорогов весом от полутонны до 600 кг.

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

Рисунок 20 – Количество самок с весом в заданном диапазоне

Задание 8.2.4: вывести список предпочтений.

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

Рисунок 21 – Список предпочтений

Задание 8.2.5: посчитать количество особей единорогов обоих полов.

```
> db.unicorns.aggregate({$group:{_id:"$gender", count:{$sum:1}}})
{ "_id" : "m", "count" : 6 }
{ "_id" : "f", "count" : 5 }
```

Рисунок 22 – количество особей разных полов

Задание 8.2.6:

1) Выполнить команду:

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

2) Проверить содержимое коллекции unicorns.

```
> db.unicorns.aggregate({$group:{_id:"$gender", count:{$sum:1}}})
{ "_id" : "m", "count" : 6 }
{ "_id" : "f", "count" : 5 }
> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("62b1dd08110ee093f9136313"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m",
 "vampires" : 63 }
{ "_id" : ObjectId("62b1dd08110ee093f9136314"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f",
 "vampires" : 43 }
{ "_id" : ObjectId("62b1dd08110ee093f9136315"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" :
"m", "vampires" : 182 }
{ "_id" : ObjectId("62b1dd08110ee093f9136316"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "
gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("62b1dd08110ee093f9136317"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f
", "vampires" : 40 }
{ "_id" : ObjectId("62b1dd08110ee093f9136318"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "
vampires" : 39 }
{ "_id" : ObjectId("62b1dd08110ee093f9136319"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m",
 "vampires" : 2 }
{ "_id" : ObjectId("62b1dd08110ee093f913631a"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f
", "vampires" : 33 }
{ "_id" : ObjectId("62b1dd08110ee093f913631b"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "
m", "vampires" : 54 }
{ "_id" : ObjectId("62b1dd08110ee093f913631c"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }

{ "_id" : ObjectId("62b1dd08110ee093f913631d"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m
", "vampires" : 165 }
{ "_id" : ObjectId("62b1e314110ee093f913631e"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Рисунок 23 – Выполнение команды и проверка содержимого коллекции

Задание 8.2.7:

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"}, {name: "Ayna", weight: 800, vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({"name": "Ayna"})
{ "_id" : ObjectId("62b1dd08110ee093f9136317"), "name" : "Ayna", "weight" : 800, "vampires" : 51 }
```

Рисунок 24 – Модификация данных о единороге Ayna

Задание 8.2.8:

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"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>  db.unicorns.find({"name": "Raleigh"})
{ "_id" : ObjectId("62b1dd08110ee093f9136319"), "name" : "Raleigh", "loves" : "RedBull", "weight" : 421, "gender" : "
m", "vampires" : 2 }
```

Рисунок 25 – Модификация данных о единороге Raleigh

Задание 8.2.9:

1) Всем самцам единорогов увеличить количество убитых вапмиров на 5.
2) Проверить содержимое коллекции unicorns.

```
> db.unicorns.update({gender: "f"}, {$inc: {vampires: 5}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find()
{ "_id" : ObjectId("62b1dd08110ee093f9136313"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62b1dd08110ee093f9136314"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 48 }
{ "_id" : ObjectId("62b1dd08110ee093f9136315"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("62b1dd08110ee093f9136316"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("62b1dd08110ee093f9136317"), "name" : "Ayna", "weight" : 800, "vampires" : 51 }
{ "_id" : ObjectId("62b1dd08110ee093f9136318"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62b1dd08110ee093f9136319"), "name" : "Raleigh", "loves" : "RedBull", "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62b1dd08110ee093f913631a"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62b1dd08110ee093f913631b"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62b1dd08110ee093f913631c"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62b1dd08110ee093f913631d"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("62b1e314110ee093f913631e"), "name" : "Barny", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Рисунок 26 – Модификация данных о количестве убитых вампиров

Задание 8.2.10:

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

```
> db.towns.update({name: "Portland"}, {$unset: {mayor: 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find()
{ "_id" : ObjectId("62b1dacb110ee093f9136310"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" :
 ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("62b1db74110ee093f9136311"), "name" : "New York", "population" : 22200000, "last_sensus" :
ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael
 Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("62b1db94110ee093f9136312"), "name" : "Portland", "population" : 528000, "last_sensus" : IS
ODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ] }
```

Рисунок 27 – Модификация данных о городе Портланд

Задание 8.2.11:

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"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>  db.unicorns.find({name: "Pilot"})
{ "_id" : ObjectId("62b1dd08110ee093f913631b"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolat
e" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
```

Рисунок 28 – Модификация данных о единороге Pilot

Задание 8.2.12:

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", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.find({name: "Aurora"})
{ "_id" : ObjectId("62b1dd08110ee093f9136314"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar",
 "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 48 }
```

Рисунок 29 – Модификация данных о единороге Aurora

Задание 8.2.13:

1) Создать коллекцию towns, включающую некоторые документы.

```
> db.createCollection("towns")
{ "ok" : 1 }
> doc1={name: "Punxsutawney ",
... popujatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: ["phil the groundhog"],
... mayor: {
...     name: "Jim Wehrle"
... }}
{
        "name" : "Punxsutawney ",
        "popujatiuon" : 6200,
        "last_sensus" : ISODate("2008-01-31T00:00:00Z"),
        "famous_for" : [
                "phil the groundhog"
        ],
        "mayor" : {
                "name" : "Jim Wehrle"
        }
}
> db.towns.insert(doc1)
WriteResult({ "nInserted" : 1 })
```

Рисунок 30 – Часть скрипта создания коллекции towns с документами из текста лабораторной работы

2) Удалить документы с беспартийными мэрами.
3) Проверить содержание коллекции.

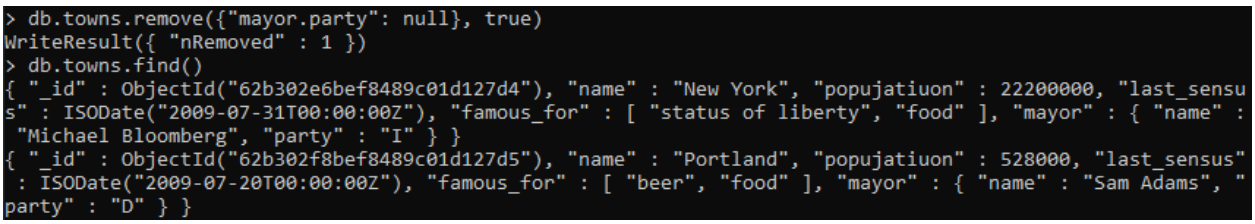

Рисунок 31 – Удаление беспартийных мэров и проверка содержания

4) Очистить коллекцию.
5) Просмотрите список доступных коллекций.

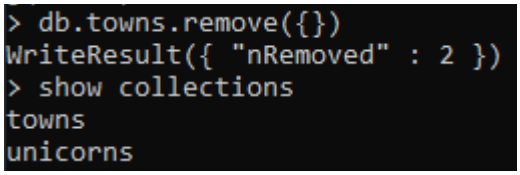

Рисунок 32 – Очистка коллекции и список доступных коллекций

Задание 8.3.1:

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

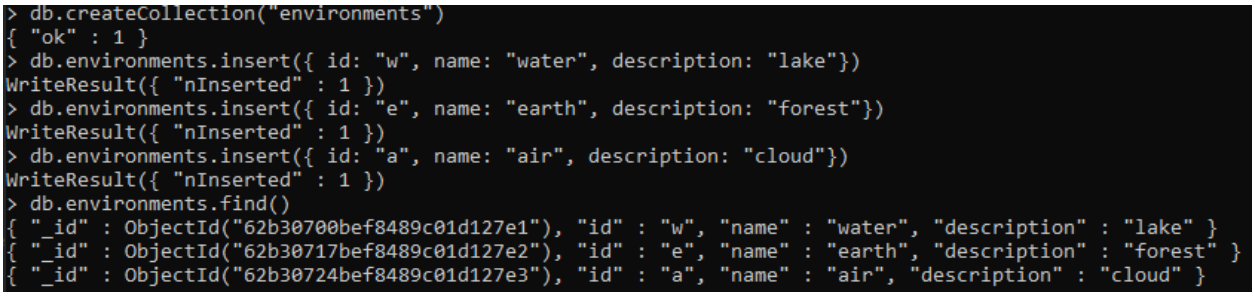

Рисунок 33 – Коллекция с зонами обитания

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

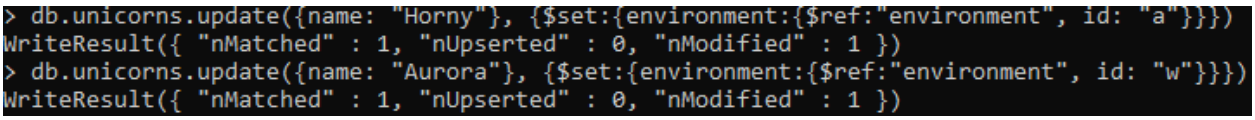

Рисунок 34 – Создание ссылки на зону обитания

3) Проверить содержание коллекции единорогов.

Рисунок 35 – Содержание коллекции unicorns

Задание 8.3.2: проверить, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique.

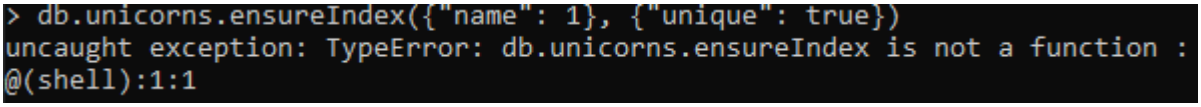

Рисунок 36 – Попытка задать индекс

Создать предлагаемый в задании индекс не удалось.

Задание 8.3.3:

1) Получить информацию о всех индексах коллекции unicorns.
2) Удалить все индексы, кроме индекса для идентификатора.

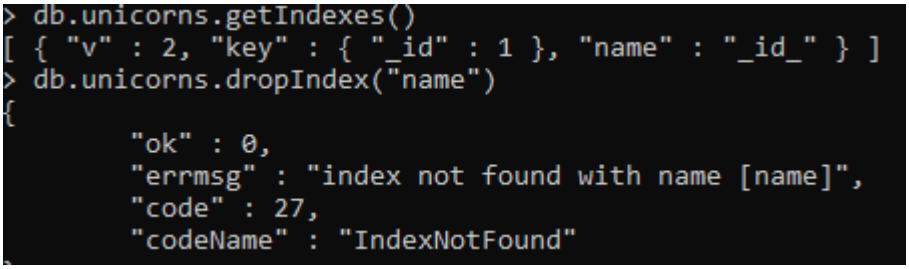

Рисунок 37 – Получение информации об индексах и попытка удаления индексов

3) Попытаться удалить индекс для идентификатора.

```
> db.unicorns.dropIndex("_id_")
{
        "ok" : 0,
        "errmsg" : "cannot drop _id index",
        "code" : 72,
        "codeName" : "InvalidOptions"
}
```

Рисунок 38 – Попытка удаления индекса идентификатора

Задание 8.3.4:

1) Создать объемную коллекцию numbers, задействовав курсор:

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

```
> db.createCollection("numbers")
{ "ok" : 1 }
> var cursor = db.numbers.find();null;
null
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
WriteResult({ "nInserted" : 1 })
```

Рисунок 39 – Создание коллекции numbers

2) Выбрать последних четыре документа.

```
> db.numbers.find().sort({value:-1}).limit(4)
{ "_id" : ObjectId("62b30e41bef8489c01d2ae83"), "value" : 99999 }
{ "_id" : ObjectId("62b30e41bef8489c01d2ae82"), "value" : 99998 }
{ "_id" : ObjectId("62b30e41bef8489c01d2ae81"), "value" : 99997 }
{ "_id" : ObjectId("62b30e41bef8489c01d2ae80"), "value" : 99996 }
```

Рисунок 40 – Выбор последних четырех документов

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

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

Рисунок 41 – Анализ плана выполнения запроса 2

Время выполнения составило 78 мс.

4) Создать индекс для ключа value.

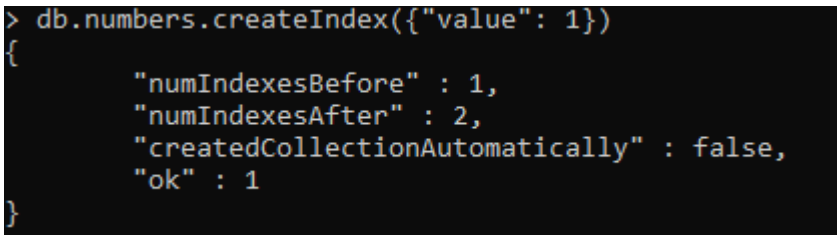

Рисунок 42 – Создание индекса для ключа value

5) Получить информацию о всех индексах коллекции numbers.

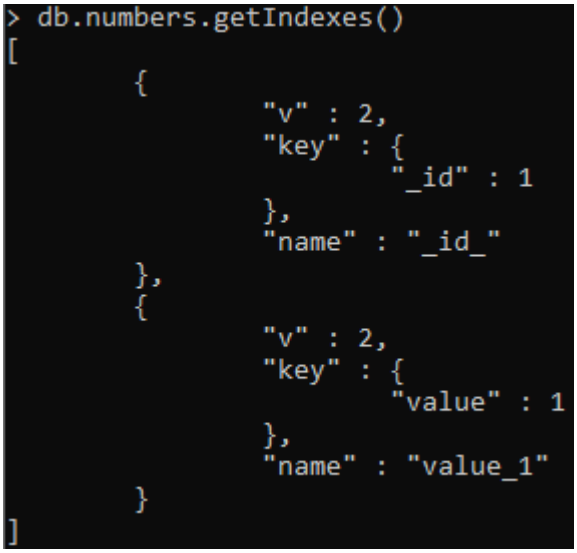

Рисунок 43 – Информация о всех индексах коллекции numbers

6) Выполнить запрос 2.
7) Проанализировать план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

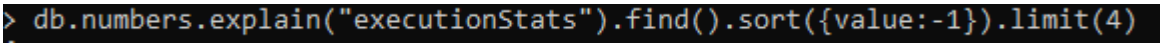

Рисунок 44 – Повторный анализ плана выполнения запроса 2

Время выполнения составило 20 мс.

8) Сравнить время выполнения запросов с индексом и без. Дать ответ на вопрос: какой запрос более эффективен?

Время выполнения запроса индексом меньше на 58 мс. Очевидно, запрос с индексом более эффективен.

Выводы: я познакомилась с СУБД MongoDB и изучила различные средства работы с данными.