Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

Факультет **Инфокоммуникационных технологий**

Образовательная программа **Мобильные и сетевые технологии**

Направление подготовки **09.03.03 Прикладная информатика**

О Т Ч Е Т

по лабораторной работе №6

Ребров Сергей Андреевич, № группы K3239

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.createCollection('unicorns');

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

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

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

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

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

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

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

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

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

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

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

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

db.unicorns.insert(document);

db.unicorns.find();

2.2.1:

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

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

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

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

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

db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1);

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, gender: 0}).sort({name: 1}).limit(3);

2.2.3:

Вывести список единорогов в обратном порядке добавления

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

2.2.4:

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор

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

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});

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});

2.3.3:

Найти всех единорогов, не имеющих ключ vampires

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

2.3.4:

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

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

3.1.1:

1) *Создайте коллекцию towns, включающую следующие документы*

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*

use towns;

db.createCollection('towns');

db.towns.insert({name: 'Punxsutawney', population: 6200, last_sensus: ISODate('2008-01-31'), famous_for: [''], mayor: {name: 'Jim Wehrle'}});

db.towns.insert({name: 'New York', population: 22200000, last_sensus: ISODate('2009-07-31'), famous_for: ['status of liberty', 'food'], mayor: {name: 'Michael Bloomberg', party: 'I'}});

db.towns.insert({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: 1, _id: 0});

db.towns.find({'mayor.party': {$exists: false}}, {name: 1, mayor: 1, _id: 0});

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*
4) *Содержание коллекции единорогов* `unicorns`

var cursor = db.unicorns.find({gender: 'm'}); null;

cursor.sort({name: 1}).limit(2); null;

cursor.forEach(function(uni) { print(uni.name); });

3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг

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

3.2.2:

Вывести список предпочтений

db.unicorns.distinct('loves');

3.2.3:

Подсчитать количество особей единорогов обоих полов

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

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();
```

3.3.3:

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

```
db.unicorns.update({name: 'Raleigh'}, {$set: {loves: ['redbull']}});

db.unicorns.find();
```

3.3.4:

1) *Самцам единорогов увеличить количество убитых вампиров на 5*
2) *Проверить содержимое коллекции* `unicorns`

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

db.unicorns.find();
```

3.3.5:

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

```
use towns;

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

db.towns.find();
```

3.3.6:

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

use learn;

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

db.unicorns.find();

3.3.7:

1) *Изменить информацию о самке единорога* `Aurora`. *Теперь она любит еще сахар и лимоны*
2) *Проверить содержимое коллекции* `unicorns`

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

db.unicorns.find();

3.4.1:

1) *Создайте коллекцию towns, включающую следующие документы*
2) *Удалите документы с беспартийными мэрами*
3) *Проверьте содержание коллекции*
4) *Очистите коллекцию*
5) *Просмотрите список доступных коллекций*

use towns;

db.towns.insert({name: 'Punxsutawney', population: 6200, last_sensus: ISODate('2008-01-31'), famous_for: ['phil the groundhog'], mayor: {name: 'Jim Wehrle'}});

db.towns.insert({name: 'New York', population: 22200000, last_sensus: ISODate('2009-07-31'), famous_for: ['status of liberty', 'food'], mayor: {name: 'Michael Bloomberg', party: 'I'}});

db.towns.insert({name: 'Portland', popujatiuon: 528000, last_sensus: ISODate('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;

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, использую второй способ автоматического связывания*
3) *Проверьте содержание коллекции единорогов*

db.places.insert({_id: 'castle', name: 'rainbow castle', description: 'the castle that is located on the rainbow'});

db.places.insert({_id: 'field', name: 'magic field', description: 'the magical field enchanted by an ancient witch'});

db.unicorns.updateMany({gender: 'f'}, {$set: {place: {$ref: 'places', $id: 'castle'}}});

db.unicorns.updateMany({gender: 'm'}, {$set: {place: {$ref: 'places', $id: 'field'}}});

db.unicorns.find();

4.2.1:

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

db.unicorns.ensureIndex({name: 1}, {unique: true});

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();

db.unicorns.dropIndex('_id_');

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('executionStats');

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

Первый способ - 53, второй способ - 7, более эффективен второй способ.

Вывод:

В результате лабораторной работы с MongoDB были освоены основы работы с данными, включая создание, чтение, обновление и удаление (CRUD), а также использование вложенных объектов и агрегаций. Эксперименты с различными моделями данных и индексами обогатили опыт работы с этой базой данных, что будет полезно в будущих проектах.