

Министерство науки и высшего образования Российской Федерации

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

Факультет инфокоммуникационных технологий

Практическая работа № 8 «Работа с БД в СУБД MongoDB»

по дисциплине «Проектирование и реализация баз данных»

Выполнила:

студентка II курса ИКТ
группы К3242

Тостановская Анна Евгеньевна

Проверила:

Говорова Марина Михайловна

Санкт-Петербург
2021

Цель работы: овладение 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:

1. Проанализировать предметную область согласно варианту задания.
2. Выполнить инфологическое моделирование базы данных по заданной предметной области с использованием метода ER-диаграмм («сущность-связь») в комбинированной нотации Питера Чена - Кириллова.
3. Реализовать разработанную ИЛМ в нотации IDEF1X.

Ход работы:

8.1 CRUD-ОПЕРАЦИИ В СУБД MONGODB. ВСТАВКА ДАННЫХ. ВЫБОРКА ДАННЫХ

8.1.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

Практическое задание 8.1.1:

- 1) Создайте базу данных learn.
- 2) Заполните коллекцию единорогов unicorns:

```
> 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: 'Rooooooodles', 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 })
> document=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
> db.users.insert(document)
WriteResult({ "nInserted" : 1 })
```

- 3) 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.users.insert(document)
WriteResult({ "nInserted" : 1 })
```

4) Проверьте содержимое коллекции с помощью метода find.

```
> db.users.find()
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b69cba62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b69cee62e607a3e9f4b822"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b69d4562e607a3e9f4b828"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

8.2.2 ВЫБОРКА ДАННЫХ ИЗ БД

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.

```
> db.users.find({gender: 'm'}).sort({name: 1});
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

```
> db.users.find({gender: 'f'}).limit(3).sort({name: 1});
{ "_id" : ObjectId("60b69cba62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
```

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

```
> db.users.find({gender: 'f', loves: "carrot"}).limit(1);
{ "_id" : ObjectId("60b69cba62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

```
> db.users.findOne({gender: 'f', loves: "carrot"});
{
  "_id" : ObjectId("60b69cba62e607a3e9f4b81f"),
  "name" : "Aurora",
  "loves" : [
    "carrot",
    "grape"
  ],
  "weight" : 450,
  "gender" : "f",
  "vampires" : 43
}
```

Практическое задание 8.1.3:

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

```
> db.users.find({gender: 'm'}, {gender: 0, loves: 0});
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "weight" : 704, "vampires" : 165 }
```

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
> db.users.find().sort({$natural: -1});
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b69d4562e607a3e9f4b828"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b69cee62e607a3e9f4b822"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60b69c9d562e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

Практическое задание 8.1.5:

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

```
> db.users.find({}, { _id: 0, loves:{$slice: 1}})
{ "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" : "Rooodooles", "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 }
{ "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "name" : "Leia", "loves" : [ "apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape" ], "weight" : 540, "gender" : "f" }
{ "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

8.2.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 8.1.6:

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

```
> db.users.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" }
```

Практическое задание 8.1.7:

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

```
> db.users.find({gender:'m', weight: {$gte: 500}, loves: {$all: ["grape", "lemon"]}}, {_id: 0});
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
```

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
> db.users.find({vampires: {$exists:true}});
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b69c9d562e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooodooles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b69cee62e607a3e9f4b822"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```


Практическое задание 8.1.9:

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

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

8.2 ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB.

ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ

8.2.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 8.2.1:

1) Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.find()
{ "_id" : ObjectId("60b797e85dc408ebab6796d6"), "name" : "Punxsutawney ", "populatioun" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrle" } }
{ "_id" : ObjectId("60b798ec5dc408ebab6796d7"), "name" : "New York", "populatioun" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
{ "_id" : ObjectId("60b799215dc408ebab6796d8"), "name" : "Portland", "populatioun" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "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.

```
> db.towns.find({"mayor.party": "I"}, {name: 1, mayor: 1, _id: 0})
{ "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
```

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}}, {name: 1, mayor: 1, _id: 0})
{ "name" : "Punxsutawney ", "mayor" : { "name" : "Jim Wehrle" } }
```

8.2.2 ИСПОЛЬЗОВАНИЕ JAVASCRIPT

8.2.3 КУРСОРЫ

Практическое задание 8.2.2:

1) Сформировать функцию для вывода списка самцов единорогов.

```
> fn = function() {return this.gender=="m";}
function() {return this.gender=="m";}
```

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

```
> var cursor = db.users.find(fn);null;
null
> cursor.limit(2).sort({name: 1})
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
```

3) Вывести результат, используя forEach.

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

8.2.4 АГРЕГИРОВАННЫЕ ЗАПРОСЫ

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 8.2.4:

Вывести список предпочтений.

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

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

```
> db.users.aggregate([{$group:{_id:"gender", count:{$sum:1}}}]
{ "_id" : "gender", "count" : 12 }
```

8.2.5 РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 8.2.6:

1. Выполнить команду:

```
> db.users.save({name: 'Barney', loves: ['grape'], weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
```

2. Проверить содержимое коллекции unicorns.

```
> db.users.find()
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("60b69c3a62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("60b69cee62e607a3e9f4b822"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("60b69d4562e607a3e9f4b828"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("60b7a7375dc408ebab6796da"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Практическое задание 8.2.7:

1. Для самки единорога Айна внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.

```
> db.users.update({name: 'Ayna'}, {name: 'Ayna', loves: ['strawberry', 'lemon'], gender: 'f', weight: 800, vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

2. Проверить содержимое коллекции unicorn (проверены отдельно значения для Айна)

```
> db.users.find({name: 'Ayna'})
{ "_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "gender" : "f", "weight" : 800, "vampires" : 51 }
```

Практическое задание 8.2.8:

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

```
> db.users.update({name: 'Raleigh', gender: 'm'}, {$set: {loves: ['redbull']}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

2. Проверить содержимое коллекции unicorns (проверены отдельно значения для Raleigh)

```
> db.users.find({name:"Raleigh"})
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
```

Практическое задание 8.2.9:

1. Всем самцам единорогов увеличить количество убитых вампиров на 5.

```
> db.users.update({gender:'m'}, {$inc:{vampires:5}},{multi:true})
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
```

2. Проверить содержимое коллекции unicorns.

```
> db.users.find({gender:'m'})
{ "_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Rooodoodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("60b69d0662e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("60b7a7375dc408ebab6796da"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Практическое 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"}, {$set: {mayor.party: 1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.towns.find()
{ "_id" : ObjectId("60b797e85dc408ebab6796d6"), "name" : "Punxsutawney ", "populatiuon" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Wehrl", "party" : 1 } }
{ "_id" : ObjectId("60b798ec5dc408ebab6796d7"), "name" : "New York", "populatiuon" : 22200000, "last_sensus" : ISODate("2009-07-31T00:00:00Z"), "famous_for" : [ "status of liberty", "food" ], "mayor" : { "name" : "Michael Bloomberg", "party" : 1 } }
{ "_id" : ObjectId("60b799215dc408ebab6796d8"), "name" : "Portland", "populatiuon" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : 1 } }
```

Практическое задание 8.2.11:

1. Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.
2. Проверить содержимое коллекции unicorns (проверены отдельно значения для Pilot)

```
> db.users.find({name:"Pilot"})
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
```

Практическое задание 8.2.12:

1. Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.

```
> db.users.update({name:"Aurora"}, {$addToSet:{loves:{each:["sugar", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

2. Проверить содержимое коллекции unicorns (проверены отдельно значения для Aurora)

```
> db.users.find({name:"Aurora"})
{ "_id" : ObjectId("60b69d3a62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

8.2.6 УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

Практическое задание 8.2.13:

- 1) Создайте коллекцию towns, включающую следующие документы:


```

db.towns.insert({name: "Punxsutawney ",
... population: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: ["phil the groundhog"],
... mayor: {
...   name: "Jim Mehrle"
... }})
writeResult({ "nInserted" : 1 })
db.towns.find()
{ "_id" : ObjectId("60b797e85dc408ebab6796d6"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Mehrle" } }
{ "_id" : ObjectId("60b798ec5dc408ebab6796d7"), "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("60b799215dc408ebab6796d8"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams" } }
{ "_id" : ObjectId("60b7b1a85dc408ebab6796db"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "phil the groundhog" ], "mayor" : { "name" : "Jim Mehrle" } }
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"
... }})
writeResult({ "nInserted" : 1 })
db.towns.insert({name: "Portland",
... population: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...   name: "Sam Adams",
...   party: "D"
... }})
writeResult({ "nInserted" : 1 })
db.towns.find()
{ "_id" : ObjectId("60b797e85dc408ebab6796d6"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "" ], "mayor" : { "name" : "Jim Mehrle" } }
{ "_id" : ObjectId("60b798ec5dc408ebab6796d7"), "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("60b799215dc408ebab6796d8"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams" } }
{ "_id" : ObjectId("60b7b1a85dc408ebab6796db"), "name" : "Punxsutawney ", "population" : 6200, "last_sensus" : ISODate("2008-01-31T00:00:00Z"), "famous_for" : [ "phil the groundhog" ], "mayor" : { "name" : "Jim Mehrle" } }
{ "_id" : ObjectId("60b7b1e55dc408ebab6796dc"), "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("60b7b1f55dc408ebab6796dd"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }

```

2) Удалите документы с беспартийными мэрами.

```

> db.towns.remove({"mayor.party":{"$exists:false}})
WriteResult({ "nRemoved" : 3 })

```

3) Проверьте содержание коллекции.

```

db.towns.find()
{ "_id" : ObjectId("60b798ec5dc408ebab6796d7"), "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("60b7b1e55dc408ebab6796dc"), "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("60b7b1f55dc408ebab6796dd"), "name" : "Portland", "population" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams", "party" : "D" } }

```

4) Очистите коллекцию.

```

> db.towns.remove({})
WriteResult({ "nRemoved" : 3 })

```

5) Просмотрите список доступных коллекций.

```

> show collections
towns
unicorns
users

```

8.3 ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

8.3.1 ССЫЛКИ В БД

Практическое задание 8.3.1:

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

```

> db.areas.insert({_id:"fromds", name: "Treg fromds", desc:"In the world"})
WriteResult({ "nInserted" : 1 })
> db.areas.insert({_id:"qwerty", name: "Qwerty uhi", desc:"In the world"})
WriteResult({ "nInserted" : 1 })

```

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

```

> db.users.update({name:"Horny"},{$set:{area:{$ref:"areas", $id:"fromds"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.users.update({name:"Aurora"},{$set:{area:{$ref:"areas", $id:"qwerty"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.users.update({name:"Pilot"},{$set:{area:{$ref:"areas", $id:"qwerty"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

```


3) Проверьте содержание коллекции едиорогов.

```
db.users.find()
{"_id" : ObjectId("60b69bfff62e607a3e9f4b81e"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68, "area" : DBRef("areas", "fromds" ) }
{"_id" : ObjectId("60b69cba62e607a3e9f4b81f"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43, "area" : DBRef("areas", "qerty" ) }
{"_id" : ObjectId("60b69cc562e607a3e9f4b820"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{"_id" : ObjectId("60b69cd562e607a3e9f4b821"), "name" : "Roocooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{"_id" : ObjectId("60b69cee62e607a3e9f4b822"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{"_id" : ObjectId("60b69cf862e607a3e9f4b823"), "name" : "Aynna", "loves" : [ "strawberry", "lemon" ], "gender" : "f", "weight" : 800, "vampires" : 51 }
{"_id" : ObjectId("60b69d062e607a3e9f4b824"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{"_id" : ObjectId("60b69d1162e607a3e9f4b825"), "name" : "Raleigh", "loves" : [ "redbull" ], "weight" : 421, "gender" : "m", "vampires" : 7 }
{"_id" : ObjectId("60b69d1a62e607a3e9f4b826"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{"_id" : ObjectId("60b69d3a62e607a3e9f4b827"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59, "area" : DBRef("areas", "qerty" ) }
{"_id" : ObjectId("60b69d4562e607a3e9f4b828"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{"_id" : ObjectId("60b69d5162e607a3e9f4b829"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{"_id" : ObjectId("60b7a7375dc488ebab6796da"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

8.3.2 НАСТРОЙКА ИНДЕКСОВ

Практическое задание 8.3.2:

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

```
> db.users.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
```

8.3.3 УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 8.3.3:

1) Получите информацию о всех индексах коллекции unicorns .

```
> db.users.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
```

2) Удалите все индексы, кроме индекса для идентификатора.

```
> db.users.dropIndex("name_1")
{
  "ok" : 0,
  "errmsg" : "index not found with name [name_1]",
  "code" : 27,
  "codeName" : "IndexNotFound"
}
```

3) Попробуйте удалить индекс для идентификатора.

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

8.3.4 ПЛАН ЗАПРОСА

Практическое задание 8.3.4:

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?

```
> db.numbers.explain("executionStats").find({value:{$gte:99996}})
{
  "explainVersion" : "2",
  "queryPlanner" : {
    "namespace" : "learn.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "value" : {
        "$gte" : 99996
      }
    },
    "maxIndexedOrSolutionsReached" : false,
    "maxIndexedAndSolutionsReached" : false,
    "maxScansToExplodeReached" : false,
    "winningPlan" : {
      "queryPlan" : {
        "stage" : "EOF",
        "planNodeId" : 1
      },
      "slotBasedPlan" : {
        "slots" : "$$RESULT=s4 $$RID=s3 env: { s1 = TimeZoneDatabase(America/Lima...Etc/GMT+9) (timeZoneDB), s2 = 1622655882735 (NOW) }",
        "stages" : "[1] project [s3 = Nothing, s4 = Nothing] \n[1] limit 0 \n[1] coscan "
      }
    },
    "rejectedPlans" : [ ]
  },
}
```

```

"executionStats" : {
  "executionSuccess" : true,
  "nReturned" : 0,
  "executionTimeMillis" : 11,
  "totalKeysExamined" : 0,
  "totalDocsExamined" : 0,
  "executionStages" : {
    "stage" : "project",
    "planNodeId" : 1,
    "nReturned" : 0,
    "executionTimeMillisEstimate" : 0,
    "opens" : 1,
    "closes" : 1,
    "saveState" : 0,
    "restoreState" : 0,
    "isEOF" : 1,
    "projections" : {
      "3" : "Nothing ",
      "4" : "Nothing "
    },
    "inputStage" : {
      "stage" : "limit",
      "planNodeId" : 1,
      "nReturned" : 0,
      "executionTimeMillisEstimate" : 0,
      "opens" : 1,
      "closes" : 1,
      "saveState" : 0,
      "restoreState" : 0,
      "isEOF" : 1,
      "limit" : 0,
      "inputStage" : {
        "stage" : "coscan",
        "planNodeId" : 1,
        "nReturned" : 0,
        "executionTimeMillisEstimate" : 0,
        "opens" : 1,
        "closes" : 1,
        "saveState" : 0,
        "restoreState" : 0,
        "isEOF" : 0
      }
    }
  }
}

```

```

"command" : {
  "find" : "numbers",
  "filter" : {
    "value" : {
      "$gte" : 99996
    }
  },
  "$db" : "learn"
},
"serverInfo" : {
  "host" : "DESKTOP-GS600L6",
  "port" : 27017,
  "version" : "5.0.0-rc0",
  "gitVersion" : "9a324616c6557efc15ee6bf0c42587c0f475e573"
},
"serverParameters" : {
  "internalQueryFacetBufferSizeBytes" : 104857600,
  "internalQueryFacetMaxOutputDocSizeBytes" : 104857600,
  "internalLookupStageIntermediateDocumentMaxSizeBytes" : 104857600,
  "internalDocumentSourceGroupMaxMemoryBytes" : 104857600,
  "internalQueryMaxBlockingSortMemoryUsageBytes" : 104857600,
  "internalQueryProhibitBlockingMergeOnMongoS" : 0,
  "internalQueryMaxAddToSetBytes" : 104857600,
  "internalDocumentSourceSetWindowFieldsMaxMemoryBytes" : 104857600
},
"ok" : 1

```

```

> db.numbers.createIndex({"value": 1})
{
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "createdCollectionAutomatically" : true,
  "ok" : 1
}
> db.numbers.explain("executionStats").find({"value":{"$gte:99996}})
{
  "explainVersion" : "2",
  "queryPlanner" : {
    "namespace" : "learn.numbers",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "value" : {
        "$gte" : 99996
      }
    },
    "maxIndexedOrSolutionsReached" : false,
    "maxIndexedAndSolutionsReached" : false,
    "maxScansToExplodeReached" : false,
    "winningPlan" : {
      "queryPlan" : {
        "stage" : "FETCH",
        "planNodeId" : 2,
        "inputStage" : {
          "stage" : "IXSCAN",
          "planNodeId" : 1,
          "keyPattern" : {
            "value" : 1
          },
          "indexName" : "value_1",
          "isMultiKey" : false,
          "multiKeyPaths" : {
            "value" : [ ]
          },
          "isUnique" : false,
          "isSparse" : false,
          "isPartial" : false,
          "indexVersion" : 2,
          "direction" : "forward",
          "indexBounds" : {
            "value" : [
              "[99996.0, inf.0]"
            ]
          }
        }
      }
    },
    "executionStats" : {
      "executionSuccess" : true,
      "nReturned" : 0,
      "executionTimeMillis" : 1,
      "totalKeysExamined" : 0,
      "totalDocsExamined" : 0,
      "executionStages" : {
        "stage" : "nlj",
        "planNodeId" : 2,
        "nReturned" : 0,
        "executionTimeMillisEstimate" : 0,
        "opens" : 1,
        "closes" : 1,
        "saveState" : 0,
        "restoreState" : 0,
        "isEOF" : 1,
        "innerOpens" : 0,
        "innerCloses" : 0,
        "outerProjects" : [ ],
        "outerCorrelated" : [
          NumberLong(7),
          NumberLong(3),
          NumberLong(4),
          NumberLong(5),
          NumberLong(6)
        ]
      }
    }
  }
}

```

После создания индекса value значение executionTimeMillis изменилось с 11 до 1. Значит, для ускорения работы стоит задавать индексы.

Вывод: выполняя данную лабораторную работу, мне удалось познакомиться с CRUD операциями, а также с инструментами работы с базами данных в MongoDB.