Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
Кафедра экономической информатики

Отчет
По лабораторным работам №6-7

ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB. ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ. ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

Выполнила:
Студент ФКП, гр.914302
Воробей Д.А.,
Проверила: Лукашевич А.Э.

Минск 2022

Цель: изучить процесс создания запросов к базе данных MongoDB, формирование выборок данных и агрегированных запросов, использование курсоров и рассмотреть принципы работы с индексами в NOSQL базах данных.

Часть 1

1. 1) Создайте коллекцию towns, включающую следующие документы

```
{ acknowledged: true,
  insertedIds:
   { '0': ObjectId("638370401afd7086efe5f8bc"),
     '1': ObjectId("638370401afd7086efe5f8bd"),
     '2': ObjectId("638370401afd7086efe5f8be") } }
```

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: false, populatiuon: false, last_sensus: false, famous_for: false})
{ 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}}, {_id: false, populatiuon: false, last_sensus: false, famous_for: false})
{ name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } }
```

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.

```
> var cursor = db.unicorns.find({gender:"m"});null;
  cursor.limit(2).sort({name:1});null;
  cursor.forEach(function(obj){print(obj);})
{
  _id: ObjectId("638371be1afd7086efe5f8cd"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId("638371be1afd7086efe5f8c2"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

3. Вывести количество самок единорогов весом от 500 до 600 кг.

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

4. Вывести список предпочтений

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

5. Посчитать количество особей единорогов обоих полов.

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

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

```
> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
< { acknowledged: true,
    insertedId: ObjectId("638373fd1afd7086efe5f8ce") }
```

2) Проверить содержимое коллекции unicorns.

```
{ _id: ObjectId("638373fd1afd7086efe5f8ce"),
  name: 'Barny',
  loves: [ 'grape' ],
  weight: 340,
  gender: 'm' }
```

7. 1) Для самки единорога Ayna внести изменения в БД:

```
> db.unicorns.updateOne({name : "Ayna"}, {$set: {weight: 800, vampires : 51}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

2) Проверить содержимое коллекции unicorns

```
{ _id: ObjectId("638371be1afd7086efe5f8c7"),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51 }
```

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

```
> db.unicorns.updateOne({name : "Raleigh"}, {$set: {loves: ["redbull"]}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

2) Проверить содержимое коллекции unicorns.

```
{ _id: ObjectId("638371be1afd7086efe5f8c9"),
  name: 'Raleigh',
  loves: [ 'redbull' ],
  weight: 421,
  gender: 'm',
  vampires: 2 }
```

9. 1) Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
> db.unicorns.updateMany({}, {$inc: {vampires: 5}}, {multi:true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 13,
    modifiedCount: 13,
    upsertedCount: 0 }
```

2) Проверить содержимое коллекции unicorns.

```
{ _id: ObjectId("638371be1afd7086efe5f8c9"),
  name: 'Raleigh',
  loves: [ 'redbull' ],
  weight: 421,
  gender: 'm',
  vampires: 7 }
```

10. 1) Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.

```
> db.towns.updateOne({name: "Portland"}, {$unset: {"mayor.party": 1}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

2) Проверить содержимое коллекции towns.

```
{ _id: ObjectId("638370401afd7086efe5f8be"),
  name: 'Portland',
  populatiuon: 528000,
  last_sensus: 2022-07-20T00:00:00.000Z,
  famous_for: [ 'beer', 'food' ],
  mayor: { name: 'Sam Adams' } }
```

11. 1) Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.

```
> db.unicorns.updateOne({name : "Pilot"}, {$push: {loves: "chocolate"}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

2) Проверить содержимое коллекции unicorns.

```
{ _id: ObjectId("638371be1afd7086efe5f8cb"),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  weight: 650,
  gender: 'm',
  vampires: 59 }
```

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

```
> db.unicorns.updateOne({name : "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "limons"]}}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

2) Проверить содержимое коллекции unicorns.

```
{ _id: ObjectId("638371be1afd7086efe5f8c3"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape', 'sugar', 'limons' ],
  weight: 450,
  gender: 'f',
  vampires: 48 }
```

13. 1) Создайте коллекцию towns, включающую следующие документы

```
{ acknowledged: true,
  insertedIds:
   { '0': ObjectId("638377a91afd7086efe5f8cf"),
     '1': ObjectId("638377a91afd7086efe5f8d0"),
     '2': ObjectId("638377a91afd7086efe5f8d1") } }
```

2) Удалите документы с беспартийными мэрами.

```
> db.towns.deleteMany({"mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
```

3) Проверьте содержание коллекции.

```
> db.towns.find()
{ _id: ObjectId("638377a91afd7086efe5f8d0"),
  name: 'New York',
  popujatiuon: 22200000,
  last_sensus: 2022-07-31T00:00:00.000Z,
  famous_for: [ 'status of liberty', 'food' ],
  mayor: { name: 'Michael Bloomberg', party: 'I' } }
{ _id: ObjectId("638377a91afd7086efe5f8d1"),
  name: 'Portland',
  popujatiuon: 528000,
  last_sensus: 2022-07-20T00:00:00.000Z,
  famous_for: [ 'beer', 'food' ],
  mayor: { name: 'Sam Adams', party: 'D' } }
```

4) Очистите коллекцию.

```
> db.towns.drop()
true
```

5) Просмотрите список доступных коллекций.

```
> show collections
unicorns
```

Часть 2

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

```
> db.createCollection("areas")
          db.areas.insertOne({_id:"field", name:"Magic Fileds"})
          db.areas.insertOne({_id:"clouds", name:"Magic Clouds"})
< { acknowledged: true, insertedId: 'clouds' }
```

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

```
> db.unicorns.updateOne({_id:ObjectId("638371be1afd7086efe5f8cd")},{$set: {area:{$ref:"areas", $id: "clouds"}}})
  db.unicorns.updateOne({_id:ObjectId("638371be1afd7086efe5f8c9")},{$set: {area:{$ref:"areas", $id: "fields"}}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

3) Проверьте содержание коллекции единорогов.

```
  { _id: ObjectId("638371be1afd7086efe5f8cd"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170,
    area: DBRef("areas", 'clouds') }
```

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

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

3. 1) Получите информацию о всех индексах коллекции unicorns

```
> db.unicorns.dropIndex("name_1")
< { nIndexesWas: 2, ok: 1 }
```

2) Удалите все индексы, кроме индекса для идентификатора.

```
> db.unicorns.dropIndex("name_1")
< { nIndexesWas: 2, ok: 1 }
```

3) Попытайтесь удалить индекс для идентификатора.

```
> db.unicorns.dropIndex("_id_")
✖ ▸ MongoServerError: cannot drop _id index
```

4. 1) Создайте объемную коллекцию numbers, задействовав курсор

```
> db.createCollection("numbers")
      for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
< { acknowledged: true,
    insertedIds: { '0': ObjectId("63837e021afd7086efe77f71") } }
```

2) Выберите последних четыре документа.

```
> db.numbers.find().sort({$natural:-1}).limit(4)
< { _id: ObjectId("63837e021afd7086efe77f71"), value: 99999 }
  { _id: ObjectId("63837e021afd7086efe77f70"), value: 99998 }
  { _id: ObjectId("63837e021afd7086efe77f6f"), value: 99997 }
  { _id: ObjectId("63837e021afd7086efe77f6e"), value: 99996 }
```

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

69

4) Создайте индекс для ключа value.

```
> db.numbers.ensureIndex({"value" : 1})
< [ 'value_1' ]
```

5) Получите информацию о всех индексах коллекции numbres.

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

6) Выполните запрос 2.

```
> db.numbers.find().sort({$natural:-1}).limit(4)
< { _id: ObjectId("63837e021afd7086efe77f71"), value: 99999 }
  { _id: ObjectId("63837e021afd7086efe77f70"), value: 99998 }
  { _id: ObjectId("63837e021afd7086efe77f6f"), value: 99997 }
  { _id: ObjectId("63837e021afd7086efe77f6e"), value: 99996 }
```

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

56

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

Эффективнее с индексом

Вывод: в ходе выполнения лабораторной работы изучен процесс написания запросов и формирования выборок в рамках нереляционных баз данных. Так же изучена работа с курсорами и индексацией на базе вендора MongoDB.