

Национальный исследовательский Университет ИТМО
Мегафакультет информационных и трансляционных технологий
Факультет мобильных и сетевых технологий

Проектирование и реализация баз данных

Лабораторная работа №6

Работа с БД в СУБД MongoDB

Работу выполнил:

Дущенко Д.А.

Группа: К3240

Преподаватель:

Говорова М.М.

Санкт-Петербург
2023

Цель 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.

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

ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

2.1.1

- Создайте базу данных learn.

```
> use learn  
< switched to db learn
```

- Используя второй способ, вставьте в коллекцию единорогов документ:

```

> document={name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', var
< {
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
> db.unicorns.insert(document)
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6603df1cbe5555ac16f0b2d1')
  }
}

```

- Проверьте содержимое коллекции с помощью метода find.

```

> db.unicorns.find()

```

ВЫБОРКА ДАННЫХ ИЗ БД

2.2.1

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

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

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

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

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

2.2.2

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

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

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

2.3.1

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

```
> db.unicorns.find({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({weight: {$gte: 500}, gender: 'm', loves: {$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'}, {_id: 0, name: 1, loves: {$slice: 1}}).sort({name: 1})
```

ЗАПРОСЫ К БАЗЕ ДАННЫХ MONGODB. ВЫБОРКА ДАННЫХ. ВЛОЖЕННЫЕ ОБЪЕКТЫ. ИСПОЛЬЗОВАНИЕ КУРСОРОВ. АГРЕГИРОВАННЫЕ ЗАПРОСЫ. ИЗМЕНЕНИЕ ДАННЫХ

ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

3.1.1

- Создайте коллекцию towns, включающую следующие документы

- 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, name: 1, mayor: 1})
```

- 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, name: 1, mayor: 1,})
```

3.1.2

- Сформировать функцию для вывода списка самцов единорогов

```
> function PrintMaleUnicorns(){return db.unicorns.find({gender:  
< [Function: PrintMaleUnicorns]  
> PrintMaleUnicorns()}
```

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

```
> var cursor = PrintMaleUnicorns().sort({name: 1}).limit(2)  
> cursor.forEach(doc => printjson(doc))
```

АГРЕГИРОВАННЫЕ ЗАПРОСЫ

3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

3.2.2

Вывести список предпочтений.

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

3.2.3

Посчитать количество особей единорогов обоих полов.

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

РЕДАКТИРОВАНИЕ ДАННЫХ

3.3.2

- Для самки единорога Айна внести изменения в БД: теперь ее вес 800, она убила 51 вампира.
- Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateOne({name: 'Ayna'}, {$set: {weight: 800, vampires: 51}})
> db.unicorns.find({name: 'Ayna'})
< {
  _id: ObjectId('6603deb2be5555ac16f0b2cb'),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 800,
  gender: 'f',
  vampires: 51
}
```

3.3.3

- Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.
- Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateOne({name: 'Raleigh'}, {$push: {loves: 'redbull'}})
```

3.3.4

- Всем самцам единорогов увеличить количество убитых вампиров на 5.
- Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires:  
< {  
    acknowledged: true,  
    insertedId: null,  
    matchedCount: 7,  
    modifiedCount: 7,  
    upsertedCount: 0  
  }  
}
```

3.3.5

- Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.
- Проверить содержимое коллекции unicorns.


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

3.3.6

- Изменить информацию о самце единорога Pilot: теперь он любит и шоколад.
- Проверить содержимое коллекции unicorns.

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

3.3.7

- Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.
- Проверить содержимое коллекции unicorns.

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

УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

3.4.1

- Создайте коллекцию towns, включающую следующие документы:
- Удалите документы с беспартийными мэрами.
- Проверьте содержание коллекции.
- Очистите коллекцию.
- Просмотрите список доступных коллекций.

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

```

> db.towns.find()
< {
  _id: ObjectId('6603ecffbe5555ac16f0b2d6'),
  name: 'New York',
  popujatiuon: 22200000,
  last_sensus: 2009-07-31T00:00:00.000Z,
  famous_for: [
    'status of liberty',
    'food'
  ],
  mayor: {
    name: 'Michael Bloomberg',
    party: 'I'
  }
}
{
  _id: ObjectId('6603ed0fbe5555ac16f0b2d7'),
  name: 'Portland',
  popujatiuon: 528000,
  last_sensus: 2009-07-20T00:00:00.000Z,

```

```
> db.towns.deleteMany({})  
< {  
  acknowledged: true,  
  deletedCount: 2  
}  
> show collections  
< towns  
unicorns
```

ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

ССЫЛКИ В БД

4.1.1

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

```
> db.habitat.insert({_id: 'Zone 1', name: 'Zone 1', description: 'D 1'})
< {
  acknowledged: true,
  insertedIds: {
    '0': 'Zone 1'
  }
}
> db.habitat.insert({_id: 'Zone 2', name: 'Zone 2', description: 'D 2'})
< {
  acknowledged: true,
  insertedIds: {
    '0': 'Zone 2'
  }
}
> db.habitat.insert({_id: 'Zone 3', name: 'Zone 3', description: 'D 3'})
< {
  acknowledged: true,
  insertedIds: {
    '0': 'Zone 3'
  }
}
> var zone2 = db.habitat.findOne({_id: 'Zone 2'})._id
> var zone1 = db.habitat.findOne({_id: 'Zone 1'})._id
> db.unicorns.update({name: 'Horny'}, {$set: {habitat: {$ref: 'habitat', $id: zone1}}})
> db.unicorns.update({name: 'Unicrom'}, {$set: {habitat: {$ref: 'habitat', $id: zone2}}})
```

```
> db.unicorns.find({gender: 'm'}).limit(2)
< {
  _id: ObjectId('6603dddabe5555ac16f0b2c6'),
  name: 'Horny',
  loves: [
    'carrot',
    'papaya'
  ],
  weight: 600,
  gender: 'm',
  vampires: 73,
  habitat: DBRef('habitat', 'Zone 1')
}
```

НАСТРОЙКА ИНДЕКСОВ

4.2.1

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

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

УПРАВЛЕНИЕ ИНДЕКСАМИ

4.3.1

- Получите информацию о всех индексах коллекции unicorns .
- Удалите все индексы, кроме индекса для идентификатора
- Попробуйте удалить индекс для идентификатора.

```
> db.unicorns.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
> db.unicorns.dropIndex('name_1')
< { nIndexesWas: 2, ok: 1 }
```

ПЛАН ЗАПРОСА

4.4.1

- Создайте объемную коллекцию numbers, задействовав курсор:
- Выберите последних четыре документа.
- Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis)
- Создайте индекс для ключа value.
- Получите информацию о всех индексах коллекции numbers.
- Выполните запрос 2.
- Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?
- 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({values: i})}
> db.numbers.find().sort({values: -1}).limit(4).explain('executionStats')
```

```
executionTimeMillis: 81,
```

```
> db.numbers.ensureIndex({'values': 1})
< [ 'values_1' ]
> db.numbers.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { values: 1 }, name: 'values_1' }
]
> db.numbers.find().sort({values: -1}).limit(4).explain('executionStats')
executionTimeMillis: 8,
```

Вывод

В ходе выполнения лабораторной работы были освоены практические навыки по применению MongoDB