Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №6
«Работа с БД в СУБД MongoDB»

по дисциплине **«Проектирование и реализация баз данных»**

Автор:Будунов Б. С.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М. М.

Санкт-Петербург 2024

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.

Практическое задание 2.1.1

1. Создайте базу данных learn.
2. Заполните коллекцию единорогов unicorns

```
test> use learn
switched to db learn
learn> db.unicorns.insertMany([
... {name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63},
... {name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43},
... {name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182},
... {name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99},
... {name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80},
... {name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40},
... {name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39},
... {name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2},
... {name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33},
... {name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54},
... {name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'}
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('659eacf8003118441033cb6e'),
    '1': ObjectId('659eacf8003118441033cb6f'),
    '2': ObjectId('659eacf8003118441033cb70'),
    '3': ObjectId('659eacf8003118441033cb71'),
    '4': ObjectId('659eacf8003118441033cb72'),
    '5': ObjectId('659eacf8003118441033cb73'),
    '6': ObjectId('659eacf8003118441033cb74'),
    '7': ObjectId('659eacf8003118441033cb75'),
    '8': ObjectId('659eacf8003118441033cb76'),
    '9': ObjectId('659eacf8003118441033cb77'),
    '10': ObjectId('659eacf8003118441033cb78')
  }
}
learn>
```

3. Используя второй способ, вставьте в коллекцию единорогов документ

```
learn> document=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 16
5})
{
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
```

```
learn> db.unicorns.insertOne(document)
{
  acknowledged: true,
  insertedId: ObjectId('659ead9a003118441033cb79')
}
learn>
```

4. Проверьте содержимое коллекции с помощью метода find

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('659eacf8003118441033cb6e'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('659eacf8003118441033cb6f'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('659eacf8003118441033cb70'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
```

Практическое задание 2.2.1

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

```
learn> db.unicorns.find({gender: "m"}).sort({name: 1})
[
  {
    _id: ObjectId('659ead9a003118441033cb79'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('659eacf8003118441033cb6e'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('659eacf8003118441033cb74'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
```

```
learn> db.unicorns.find({gender: "f"}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('659eacf8003118441033cb6f'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('659eacf8003118441033cb73'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('659eacf8003118441033cb76'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

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

Практическое задание 2.2.2

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

Практическое задание 2.2.3

1. Вывести список единорогов в обратном порядке добавления.

Практическое задание 2.1.4

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

```
learn> db.unicorns.find({}, {loves: {$slice: 1}, _id: 0})
[
  {
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Aurora',
    loves: [ 'carrot' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    name: 'Unicrom',
```

Практическое задание 2.3.1:

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

```
learn> db.unicorns.find({weight: {$gt: 500, $lt: 700}}, {_id: 0})
[
  {
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
```

Практическое задание 2.3.2:

1. Вывести список самцов единорогов весом от полтонны и предпочитающих grape и lemon, исключив вывод идентификатора.

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

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: 0}})
[
  {
    _id: ObjectId('659eacf8003118441033cb78'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

Практическое задание 2.3.4:

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

```
learn> db.unicorns.find({gender: "m"}, {name: 1, loves: {$slice: 1}}).sort({name: 1})
[
  {
    _id: ObjectId('659ead9a003118441033cb79'),
    name: 'Dunx',
    loves: [ 'grape' ]
  },
  {
    _id: ObjectId('659eacf8003118441033cb6e'),
    name: 'Horny',
    loves: [ 'carrot' ]
  },
  {
    _id: ObjectId('659eacf8003118441033cb74'),
    name: 'Kenny',
```

Практическое задание 3.1.1:

1. Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.insertMany([
... {name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...     name: "Jim Wehrle"
... }}
... ,
... {name: "New York",
... populatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...     name: "Michael Bloomberg",
... party: "I"}}
... ,
... {name: "Portland",
... populatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...     name: "Sam Adams",
... party: "D"}}
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('659ec65b003118441033cb7a'),
    '1': ObjectId('659ec65b003118441033cb7b'),
    '2': ObjectId('659ec65b003118441033cb7c')
  }
}
learn>
```

2. Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре

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

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

```
learn> db.towns.find({"mayor.party" : {$exists: 0}}, {name: 1, mayor: 1, _id: 0})
[ { name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } } ]
```

Практическое задание 3.1.2:

1. Сформировать функцию для вывода списка самцов единорогов.

```
learn> let get_unicorns_male = function() {return db.unicorns.find({gender: 'm'})}
```

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

```
learn> let cursor = get_unicorns_male().limit(2).sort({name: 1})
```

3. Вывести результат, используя forEach.

```
learn> cursor.forEach( (obj) => print(obj) )
{
  _id: ObjectId('659ead9a003118441033cb79'),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId('659eacf8003118441033cb6e'),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}

learn>
```

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
learn> db.unicorns.find({gender: 'f', weight: {$gt: 500, $lt: 600}}).count()
2
```

Практическое задание 3.2.2:

Вывести список предпочтений.

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

Практическое задание 3.2.3:

Посчитать количество особей единорогов обоих полов.

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

Практическое задание 3.3.1:

1. Выполнить вставку:

```
learn> db.unicorns.insertOne( {name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'} )
{
  acknowledged: true,
  insertedId: ObjectId('659ed5b99608a51fbb700f3d')
}
learn>
```

2. Проверить содержимое коллекции `unicorns`.

```
learn> db.unicorns.find().sort("name")
[
  {
    _id: ObjectId('659eacf8003118441033cb6f'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('659eacf8003118441033cb73'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('659ed5b99608a51fbb700f3d'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  },
```

Практическое задание 3.3.2:

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

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

2. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Ayna'})
[
  {
    _id: ObjectId('659eacf80031184411033cb73'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
learn>
```

Практическое задание 3.3.3:

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

Практическое задание 3.3.4:

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

```
learn> db.unicorns.updateMany( { gender: 'm' }, { $inc: { vampires: 5 } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn>
```

2. Проверить содержимое коллекции `unicorns`.

```
learn> db.unicorns.find({gender: 'm'})
[
  {
    _id: ObjectId('659eacf8003118441033cb6e'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('659eacf8003118441033cb70'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId('659eacf8003118441033cb71'),
```

Практическое задание 3.3.5:

1. Изменить информацию о городе Портленд: мэр этого города теперь беспартийный.

```
learn> db.towns.updateOne({name: "Portland"}, {$unset: {'mayor.party': 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: 'Portland'})
[
  {
    _id: ObjectId('659ec65b003118441033cb7c'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
learn>
```

Практическое задание 3.3.6:

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

```
learn> db.unicorns.updateOne({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId('659eacf8003118441033cb77'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn>
```

Практическое задание 3.3.7:

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

```
learn> db.unicorns.updateOne(
... {name: 'Aurora'},
... {$addToSet: {loves: {$each: ['sugar', 'lemon']}}}
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId('659eacf8003118441033cb6f'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

Практическое задание 3.4.1:

1. Удалите документы с беспартийными мэрами.

```
learn> db.towns.deleteMany({'mayor.party': {$exists: 0}})
{ acknowledged: true, deletedCount: 1 }
learn>
```

2. Проверьте содержание коллекции.

```
learn> db.towns.find()
[
  {
    _id: ObjectId('659ede99003118441033cb7e'),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('659ede99003118441033cb7f'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

3. Очистите коллекцию.

```
learn> db.towns.deleteMany({})
{ acknowledged: true, deletedCount: 2 }
learn>
```

4. Просмотрите список доступных коллекций.

```
learn> show collections
towns
unicorns
learn>
```

Практическое задание 4.1.1:

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

```
learn> db.areas.insertMany([
... {
...   "_id": "forest",
...   "name": "Enchanted Forest",
...   "description": "Habitat zone for unicorns in dense forests, filled with magic and mystery."
... }
... ,
... {
...   "_id": "mountain",
...   "name": "Mountain Sanctuary",
...   "description": "High-altitude areas where unicorns inhabit crystal-clear streams and majestic mo
untain peaks."
... }
... ,
... {
...   "_id": "underwater",
...   "name": "Sea Kingdom",
...   "description": "Habitat zone for unicorns underwater, where they swim among the sea waters and m
ating enchanted bubbles."
... }
... ])
{
  acknowledged: true,
  insertedIds: { '0': 'forest', '1': 'mountain', '2': 'underwater' }
}
```

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

```
learn> db.unicorns.updateMany( { gender: 'm' }, { $set: { area: { $ref: 'areas', $id: 'forest' } } } )

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn>
```

3. Проверьте содержание коллекции единорогов.

Практическое задание 4.2.1:

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

Практическое задание 4.3.1:

1. Получите информацию о всех индексах коллекции `unicorns` .

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

2. Удалите все индексы, кроме индекса для идентификатора.

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

3. Попытайтесь удалить индекс для идентификатора.

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

Практическое задание 4.4.1:

1. Создайте объемную коллекцию `numbers`, задействовав курсор:

```
learn> for(i = 0; i < 100000; i++){db.numbers.insertOne({value: i})}
{
  acknowledged: true,
  insertedId: ObjectId('659ef7987cf81bf2cc0ddd88')
}
learn>
```

2. Выберите последних четыре документа.

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId('659ef7987cf81bf2cc0ddd88'), value: 99999 },
  { _id: ObjectId('659ef7987cf81bf2cc0ddd87'), value: 99998 },
  { _id: ObjectId('659ef7987cf81bf2cc0ddd86'), value: 99997 },
  { _id: ObjectId('659ef7987cf81bf2cc0ddd85'), value: 99996 }
]
learn>
```

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

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

```
executionTimeMillis: 127,
```

4. Создайте индекс для ключа `value`.

```
learn> db.numbers.createIndex({value: 1})
value_1
learn>
```

5. Получите информацию обо всех индексах коллекции `numbres`.

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

6. Выполните запрос 2.

```
executionTimeMillis: 13,
```

Более эффективным оказался запрос с использованием индекса

Вывод

В данной лабораторной работе были получены основные навыки работы с NoSQL базой данных MongoDB, в том числе выполнены упражнения на работу с: CRUD операциями, агрегатными функциями, ссылками и индексами.