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

Отчет

по лабораторной работе № 5
«РАБОТА С БД В СУБД MONGODB»»
по дисциплине «**Базы данных**»

Автор: Чаптыков Николай

Факультет: ИКТ

Группа: К32422

Преподаватель: Говорова М.М.

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).

Ход работы:

Задание 8.1.1

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

```
test> use learn
switched to db learn
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm',
... vampires: 63});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b1d8527eeb791260a2b0") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f',
... vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b1d9527eeb791260a2b1") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender:
... 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b1d9527eeb791260a2b2") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm',
... vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b1d9527eeb791260a2b3") }
... 'm', vampires: 54});
```

3. Используя второй способ вставьте в коллекцию единорог документ:

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

```
learn> db.unicorns.insert(document)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b341527eeb791260a2bb") }
}
learn>
```

4. Проверьте содержимое коллекции с помощью метода find

Задание 8.1.2

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

```
learn> db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
[
  {
    _id: ObjectId("6473b1d9527eeb791260a2b1"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("6473b1da527eeb791260a2b5"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId("6473b1da527eeb791260a2b8"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

```
learn> db.unicorns.find({gender: 'm'}).limit(3).sort({name: 1})
[
  {
    _id: ObjectId("6473b341527eeb791260a2bb"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6473b1d8527eeb791260a2b0"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("6473b1da527eeb791260a2b6"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
learn>
```

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

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId("6473b1d9527eeb791260a2b1"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

Задание 8.1.3

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

```
learn> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0})
[
  {
    _id: ObjectId("6473b1d8527eeb791260a2b0"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("6473b1d9527eeb791260a2b2"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("6473b1d9527eeb791260a2b3"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("6473b1da527eeb791260a2b6"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
```

Задание 8.1.4

1. Вывести список единорогов в обратном порядке добавления

```
learn> db.unicorns.find({}).sort({$natural: -1})
[   name: 'Raleigh',
  { weight: 421,
    _id: ObjectId("6473b341527eeb791260a2bb"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,d("6473b200527eeb791260a2b9"),
    gender: 'm',',
    vampires: 165
  },vampires: 54
  {,
    _id: ObjectId("6473b20d527eeb791260a2ba"),
    name: 'Nimue',"6473b341527eeb791260a2bb"),
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'65
  },
  {
    _id: ObjectId("6473b200527eeb791260a2b9"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId("6473b1da527eeb791260a2b8"),
```

Задание 8.1.5

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',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    name: 'Roooooodles',
```

Задание 8.1.6

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

```
learn> db.unicorns.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'
  }
]
learn>
```

Задание 8.1.7

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

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

Задание 8.1.8

1. Найти всех единорогов, не имеющих ключ vampires

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId("6473b20d527eeb791260a2ba"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

Задание 8.1.9

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

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

Задание 8.2.1

1. Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.insert({name: "Punxsutawney ", population: 6200, last_sensus: ISODate("2008-01-31"), famous_for: [""], mayout: {name: "Jim Wehrle"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b85e527eeb791260a2bc") }
}
learn> db.towns.insert({name: "New York", population: 22200000, last_sensus: ISODate("2009-07-31"), famous_for: ["status of liberty", "food"], mayout: {name: "Mic
hale Bloomberg", party: "I"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b8ae527eeb791260a2bd") }
}
learn> db.towns.insert({name: "Portland", population: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"], mayout: {name: "Sam Adams", party:
 "D"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6473b8f2527eeb791260a2be") }
}
learn>
```

2. Сформировать запрос, который возвращает список городов с независимыми мэрами, вывести только название города и информацию о мэре

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

3. Сформировать запрос, который возвращает список беспартийных мэров, вывести только название города и информацию о мэре

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

Задание 8.2.2

1. Сформировать функцию для вывода списка самцов единорогов

```
learn> fetch_male = function() {return this.gender == 'm'}
[Function: fetch_male]
learn>
```

2. Создать курсор для этого списка из первых двухх особей с сортировкой в лексикографическом порядке
3. Вывести результат, используя forEach

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

Задание 8.2.3

1. Вывести количество самок единорогов весом от полтонны до 600 кг

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

Задание 8.2.4

1. Вывести список предпочтений

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

Задание 8.2.5

1. Посчитать количество особей единорогов обоих полов

```
learn> db.unicorns.aggregate({'$group': {_id: '$gender', count: {$sum: 1}}})
[ { _id: 'm', count: 6 }, { _id: 'f', count: 5 } ]
learn>
```

Задание 8.2.6

1. Выполнить команду save (insertOne для новых версий)

```
learn> db.unicorns.insertOne({ name: 'Barny', loves: ['grape'], weight: 340, gender: 'm' })
{
  acknowledged: true,
  insertedId: ObjectId("6473c8271ceeb66153a8870f")
}
learn>
```

2. Проверить содержимое коллекции unicorns

```
  {
    _id: ObjectId("6473c8271ceeb66153a8870f"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
learn>
```

Задание 8.2.7

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

```
learn> db.unicorns.updateOne({name: 'Ayna'}, { $set: {"loves": ["strawberry", "lemon"], "weight": 800, "gender": 'f', "v
ampires": 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("6473c65e1ceeb66153a88708"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
learn>
```

Задание 8.2.8

1. Для самца единорогоа Raleigh внести изменения в БД: теперь он любит редбул
2. Проверить содержимое коллекции unicorns

```
learn> db.unicorns.updateOne({name: 'Raleigh'}, {$set: {loves: ['redbull']}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Raleigh'})
[
  {
    _id: ObjectId("6473c65e1ceeb66153a8870a"),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

Задание 8.2.9

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

```
learn> db.unicorns.update({gender: 'm'}, {$inc: {vampires: 5}}, {multi: true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 7,
  modifiedCount: 7,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: 'm'})
[
  {
    _id: ObjectId("6473c65d1ceeb66153a88704"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
```

Задание 8.2.10

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

```
learn> db.towns.update({name: 'Portland'}, {$unset: {'mayout.party': 1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. Проверить содержимое коллекции towns

```
  {
    _id: ObjectId("6473b8f2527eeb791260a2be"),
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayout: { name: 'Sam Adams' }
  }
]
```

Задание 8.2.11

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

```
learn> db.unicorns.update({name: 'Pilot'}, {$push: {loves: 'chocolate'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId("6473c65f1ceeb66153a8870c"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn>
```

Задание 8.2.12

1. Изменить иформацию о самке единорогоа Aurora: теперь она любит еще и сахар, и лимоны

2. Проверить содержимое коллекции unicorns

```
learn> db.unicorns.update({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("6473c65e1ceeb66153a88705"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

Задание 8.2.13

1. Создайте коллекцию towns, включающую следующие документы
2. Удалите документы с беспартийными мэрами
3. Проверьте содержание коллекции

```
learn> db.towns.remove({'mayor.party': {$exists: 0}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find({})
[
  {
    _id: ObjectId("6473cd5b1ceeb66153a88711"),
    name: 'New York',
    population: 22200000,
    last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId("6473cd711ceeb66153a88712"),
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
learn>
  }
]
learn>
```

4. Очистите коллекцию
5. Просмотрите список доступных коллекций

```
learn> db.towns.drop()
true
learn> show collections
unicorns
learn>
```

Задание 8.3.1

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

```
learn> db.areas.find({})
[
  {
    _id: 'rus',
    name: 'Saratov',
    description: 'Place you can not escape'
  },
  { _id: 'arm', name: 'Yerevan', description: 'V nardi igraet' },
  { _id: 'uk', name: 'Cliffs of dover', description: 'nice song' },
  {
    _id: 'mars',
    name: 'Big Chungus',
    description: 'Dfk u doing there'
  }
]
learn>
```

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

```
learn> db.unicorns.update({name: 'Barny'}, {$set: {area: {$ref: 'areas', $id: 'arm'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Aurora'}, {$set: {area: {$ref: 'areas', $id: 'arm'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Pilot'}, {$set: {area: {$ref: 'areas', $id: 'rus'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name: 'Leia'}, {$set: {area: {$ref: 'areas', $id: 'mars'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn>
```

3. Проверьте содержание коллекции единорогов

```
  {
    _id: ObjectId("6473c8271ceeb66153a8870f"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 5,
    area: DBRef("areas", 'arm')
  }
]
learn>
```

Задание 8.3.2

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

```
learn> db.unicorns.createIndex({'name': 1}, {'unique': true})
name_1
learn>
```

Задание 8.3.3

1. Получите информацию обо 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 для идентификатора

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

Задание 8.1.7

1. Создайте объемную коллекцию number, задействовав курсор:
   for (i = 0; i < 10000; i++) {db.numbers.insert({value: i})}
2. Выберите последних четыре документа

```
learn> db.numbers.find({}).skip(9995)
[
  { _id: ObjectId("6473d4531ceeb66153a8ae1e"), value: 9996 },
  { _id: ObjectId("6473d4531ceeb66153a8ae1f"), value: 9997 },
  { _id: ObjectId("6473d4531ceeb66153a8ae20"), value: 9998 },
  { _id: ObjectId("6473d4531ceeb66153a8ae21"), value: 9999 }
]
learn>
```

3. Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса?

```
learn> db.numbers.explain('executionStats').find({}).skip(9996)
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: '17830885',
    planCacheKey: '17830885',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'SKIP',
      skipAmount: 0,
      inputStage: { stage: 'COLLSCAN', direction: 'forward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 3,
    executionTimeMillis: 21,
    totalKeysExamined: 0,
    totalDocsExamined: 9999,
    executionStages: {
```

Запрос выполнялся 21 миллисекунду, гораздо быстрее чем создание самой, довольно простой базы данных.

4. Создайте индекс для ключа Value

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

5. Выведите индексы в numbers

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

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

Запрос выполнялся 13 миллисекунду, гораздо быстрее чем без индекса.

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

Запрос с индексами выполнялся значительно быстрее. Разница в скорости была бы нагляднее на большем количестве данных в коллекции. Стоит также помнить, что индексы используют RAM и Storage, их количество для БД ограничено (64), также они усложняют процесс вписывания новых данных.

Выводы:

В ходе выполнения работы я глубже изучил NoSQL СУБД MongoDB. Были использованы CRUD методы, агрегирующие функции, сортировка, ссылки для связи полей, курсоры, индексы и статистика выполнения запросов.

В процессе выполнения также пришлось столкнуться и с трудностями развертывания MongoDB на Windows 10. Потребовалась дополнительная установка Mongo Shell для выполнения задач. Mongosh имеет неудобную систему для вставки текста из буфера обмена, также некоторые функции из текста лабораторной работы не могли быть исполнены на новой версии MongoDB (save, update)