Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет Инфокоммуникационных технологий

Лабораторная работа №5 по теме

«Работа с БД в СУБД MongoDB»

по дисциплине «Проектирование и реализация баз данных»

Выполнил:

студент 2 курса K32421 группы

Гафаров Данил Альбертович

Преподаватель:

Говорова Марина Михайловна

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).

Практическое задание 8.1.1:

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

```
> use learn
< switched to db learn
> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
  db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
  db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
  db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
  db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
  db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
  db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
  db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
  db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
  db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
  db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
< DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("64dbe54fbf3849f3ef2006a0")
    }
  }
```

3) Используя второй способ, вставьте в коллекцию единорогов документ:

4) Проверьте содержимое коллекции с помощью метода find.

```
> document={name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
< {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
> db.unicorns.insert(document)
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("64dbe6a2bf3849f3ef2006a1")
    }
  }
> db.unicorns.find()
< {
    _id: ObjectId("64dbe54fbf3849f3ef200696"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
```

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.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
< {
    _id: ObjectId("64dbe54fbf3849f3ef200697"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("64dbe54fbf3849f3ef20069b"),
    name: 'Ayna',
```

```
> db.unicorns.find({gender: 'm'}).sort({name: 1})
< {
    _id: ObjectId("64dbe6a2bf3849f3ef2006a1"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("64dbe54fbf3849f3ef200696"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
```

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

```
> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
< {
    _id: ObjectId("64dbe54fbf3849f3ef200697"),
    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 предпотениях и поле.

```
> db.unicorns.find({gender:'m'}, {loves: 0, gender: 0}).sort({name: 1})
< {
    _id: ObjectId("64dbe6a2bf3849f3ef2006a1"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
  {
    _id: ObjectId("64dbe54fbf3849f3ef200696"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
```

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find({}, {name: 1, _id: 0}).sort({ $natural: -1}).toArray()
< [
    { name: 'Dunx' },
    { name: 'Nimue' },
    { name: 'Pilot' },
    { name: 'Leia' },
    { name: 'Raleigh' },
    { name: 'Kenny' },
    { name: 'Ayna' },
    { name: 'Solnara' },
    { name: 'Roooooodles' },
    { name: 'Unicrom' },
    { name: 'Aurora' },
    { name: 'Horny' }
  ]
```

Практическое задание 8.1.5:

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

```
> 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
  }
```

Практическое задание 8.1.6:

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

```
> db.unicorns.find({gender: 'f', weight: {$gt: 500, $lt: 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
  }
```

Практическое задание 8.1.7:

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: ['lemon', 'grape']}}, {_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.unicorns.find({vampires: {$exists: 0}})
< {
    _id: ObjectId("64dbe54fbf3849f3ef2006a0"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
```

Практическое задание 8.1.9:

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})
< {
    name: 'Dunx',
    loves: [
      'grape'
    ]
  }
  {
    name: 'Horny',
    loves: [
      'carrot'
    ]
  }
  {
    name: 'Kenny',
    loves: [
      'grape'
    ]
  }
```

Практическое задание 8.2.1:

1) *Создайте коллекцию towns:*

```
> 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,
```

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

```
> db.towns.find({"mayor.party": 'I'}, {_id: 0, name: 1, mayor: 1})
< {
    name: 'New York',
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
> db.towns.find({"mayor.party": {$exists: 0}}, {_id: 0, name: 1, mayor: 1})
< {
    name: 'Punxsutawney ',
    mayor: {
      name: 'Jim Wehrle'
    }
  }
```

Практическое задание 8.2.2:

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*

```
> let cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2); null;
< null
> cursor.forEach(uni => print(uni))
< {
    _id: ObjectId("64dbe6a2bf3849f3ef2006a1"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
< {
    _id: ObjectId("64dbe54fbf3849f3ef200696"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
```

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 8.2.4:

Вывести список предпочтений.

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

Практическое задание 8.2.6:

1. *Выполнить команду:*

```
> db.unicorns.save({name: 'Barny', loves: ['grape'],
weight: 340, gender: 'm'})
```

2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
< {
    acknowledged: true,
    insertedId: ObjectId("64dc0fc6bf3849f3ef2006a5")
  }
> db.unicorns.findOne({name:'Barny'})
< {
    _id: ObjectId("64dc0fc6bf3849f3ef2006a5"),
    name: 'Barny',
    loves: [
      'grape'
    ],
    weight: 340,
    gender: 'm'
  }
```

Практическое задание 8.2.7:

1. *Для самки единорога* `Ayna` *внести изменения в БД: теперь ее вес 800, она убила 51 вапмира.*
2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.update({gender: 'f', name: 'Ayna'}, {$set: {weight: 800, vampires: 51}})
< DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Ayna'})
< {
    _id: ObjectId("64dbe54fbf3849f3ef20069b"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
```

Практическое задание 8.2.8:

1. *Для самца единорога* `Raleigh` *внести изменения в БД: теперь он любит рэдбул.*
2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.update({gender: 'm', name: 'Raleigh'}, {$addToSet: {loves: "redbull"}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Raleigh'})
< {
    _id: ObjectId("64dbe54fbf3849f3ef20069d"),
    name: 'Raleigh',
    loves: [
      'apple',
      'sugar',
      'redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```

Практическое задание 8.2.9:

3. *Всем самцам единорогов увеличить количество убитых вапмиров на 5.*
4. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 8,
    modifiedCount: 8,
    upsertedCount: 0
  }
> db.unicorns.find()
< {
    _id: ObjectId("64dbe54fbf3849f3ef200696"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
```

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'}, {$unset: {"mayor.party": 1}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 0,
    upsertedCount: 0
  }
> db.towns.find({name: 'Portland'})
< {
    _id: ObjectId("64dc065cbf3849f3ef2006a4"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams'
    }
```

Практическое задание 8.2.11:

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

```
> db.unicorns.update({name: 'Pilot'}, {$addToSet: {loves: "chocolate"}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Pilot'})
< {
    _id: ObjectId("64dbe54fbf3849f3ef20069f"),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon',
      'chocolate'
```

Практическое задание 8.2.12:

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

```
> db.unicorns.update({name: 'Aurora'}, {$addToSet: {loves: {$each: ["sugar", "lemons"]}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Aurora'})
< {
    _id: ObjectId("64dbe54fbf3849f3ef200697"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemons'
    ],
```

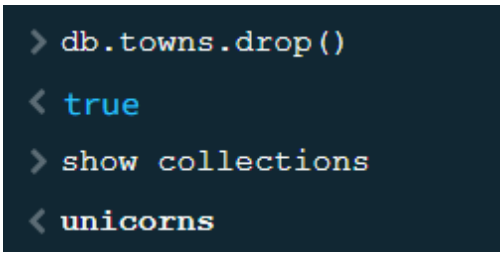
Практическое задание 8.2.13:

1) *Создайте коллекцию towns:*
2) *Удалите документы с беспартийными мэрами.*

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

3) *Проверьте содержание коллекции.*

4) *Очистите коллекцию.*
5) *Просмотрите список доступных коллекций.*



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.find()
< {
    _id: 'atl',
    name: 'Atlantis',
    info: 'The lost underwater city'
  }
  {
    _id: 'zve',
    name: 'Zveropolis',
    info: 'The home for all animals'
  }
  {
    _id: 'sha',
    name: 'Shambala',
    info: 'Magical Kingdom of spirit'
  }
```

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

```
> db.unicorns.update({name: 'Roooooodles'}, {$set: {home: {$ref: "areas", $id: "sha"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.update({name: 'Kenny'}, {$set: {home: {$ref: "areas", $id: "atl"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.update({name: 'Solnara'}, {$set: {home: {$ref: "areas", $id: "zve"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

3) *Проверьте содержание коллекции единорогов*

```
> db.unicorns.find({home: {$exists: 1}}, {name: 1, home:1})
< {
    _id: ObjectId("64dc1907bf3849f3ef2006ae"),
    name: 'Roooooodles',
    home: DBRef("areas", 'sha')
  }
  {
    _id: ObjectId("64dc1907bf3849f3ef2006af"),
    name: 'Solnara',
    home: DBRef("areas", 'zve')
  }
  {
    _id: ObjectId("64dc1907bf3849f3ef2006b1"),
    name: 'Kenny',
    home: DBRef("areas", 'atl')
  }
```

Практическое задание 8.3.2:

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

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

Возможно, но нелогично.

Практическое задание 8.3.3:

1) *Получите информацию о всех индексах коллекции* `unicorns`.
2) *Удалите все индексы, кроме индекса для идентификатора.*
3) *Попытайтесь 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 }
> db.unicorns.dropIndex("_id_")
✖ ▸ MongoServerError: cannot drop _id index
learn>
```

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.*

```
> for(i = 0; i < 100000; i++){db.number.insert({value: i})}
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("64dc2254bf3849f3ef218d55")
    }
  }
> db.number.find().skip(99996)
< {
    _id: ObjectId("64dc2254bf3849f3ef218d52"),
    value: 99996
  }
  {
    _id: ObjectId("64dc2254bf3849f3ef218d53"),
    value: 99997
  }
  {
    _id: ObjectId("64dc2254bf3849f3ef218d54"),
    value: 99998
  }
  {
    _id: ObjectId("64dc2254bf3849f3ef218d55"),
    value: 99999
  }
```

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 100000,
  executionTimeMillis: 50,
```

4) *Создайте индекс для ключа* `value`*.*

5) *Получите информацию о всех индексах коллекции* `numbres`*.*

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

6) *Выполните запрос 2.*

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

```
executionStats: {
  executionSuccess: true,
  nReturned: 100000,
  executionTimeMillis: 35,
```

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

   Запрос при наличии индексации был значительно эффективнее.

Выводы:

В ходе выполнения лабораторной работы мы познакомились с NoSQL СУБД MongoDB, получили практические навыки работы с CRUD-операциями, научились создавать коллекции, создавать индексы.