Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет прикладной информатики
Образовательная программа Мобильные и сетевые технологии
Направление подготовки 09.03.03 Мобильные и сетевые технологии

ОТЧЕТ по Лабораторной работе № 6
по дисциплине «Проектирование и реализация баз данных»

Обучающийся: Шукалов Андрей Денисович
Преподаватель: Говорова Марина Михайловна

Санкт-Петербург, 2025

Описание задания

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.

Ход работы

Практическое задание 2.1.1

1. Создать базу данных learn.
2. Заполнить коллекцию единорогов unicorns.
3. Вставить в коллекцию предопределенный документ.
4. Проверить содержимое коллекции.

```
> 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('68dccaf07ff2b8e13e6a9116')
    }
  }
```

Рис. 1: Заполнение коллекции unicorns

```
> insert_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(insert_document)
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('68dccb4e7ff2b8e13e6a9119')
    }
  }
```

Рис. 2: Вставка документа

```
> db.unicorns.find()
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910c'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a910d'),
```

Рис. 3: Содержимое коллекции

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.

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

```
> db.unicorns.find({gender: 'f'}).limit(3).sort({name: 1})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910d'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a9111'),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a9114'),
```

Рис. 4: Список самок, ограниченный первыми тремя особями и отсортированный по имени

Рис. 5: Список самцов, ограниченный первыми тремя особями и отсортированный по имени

```
> db.unicorns.findOne({gender: 'f', loves: 'carrot'})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910d'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910d'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Рис. 6: Список самок, которые любят carrot с помощью findOne и find().limit(1)

Практическое задание 2.2.2

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

```
> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910c'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a910e'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a910f'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a9112'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  }
```

Рис. 7: Список самцов единорогов без информации о предпочтениях и поле.

Практическое задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural: -1})
< {
    _id: ObjectId('68dccb4e7ff2b8e13e6a9119'),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a9116'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a9115'),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon'
    ],
```

Рис. 8: Спсиок единорогов в обратном порядке добавления.

Практическое задание 2.1.4

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

```
> db.unicorns.find({}, {name: 1, loves: {$slice: 1}, _id: 0})
< {
    name: 'Horny',
    loves: [
      'carrot'
    ]
  }
  {
    name: 'Aurora',
    loves: [
      'carrot'
    ]
  }
  {
    name: 'Unicrom',
    loves: [
      'energon'
    ]
  }
  {
    name: 'Roooooodles',
    loves: [
      'apple'
    ]
  }
```

Рис. 9: Спсиок единорогов с названием любимого предпочтения

Практическое задание 2.3.1

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

```
> 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'
    ],
```

Рис. 10: Список самок единорогов с весом от 500 до 700 кг без вывода идентификатора

Практическое задание 2.3.2

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

```
> 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
  }
```

Рис. 11: Список самцов единорогов весом от полутонны и предпочитающих grape и lemon без вывода идентификатора

Практическое задание 2.3.3

Найти всех единорогов, не имеющих ключ vampires.

```
> db.unicorns.find({vampires: {$exists: false}})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a9116'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
```

Рис. 12: Единороги без ключа vampires

Практическое задание 2.3.4

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

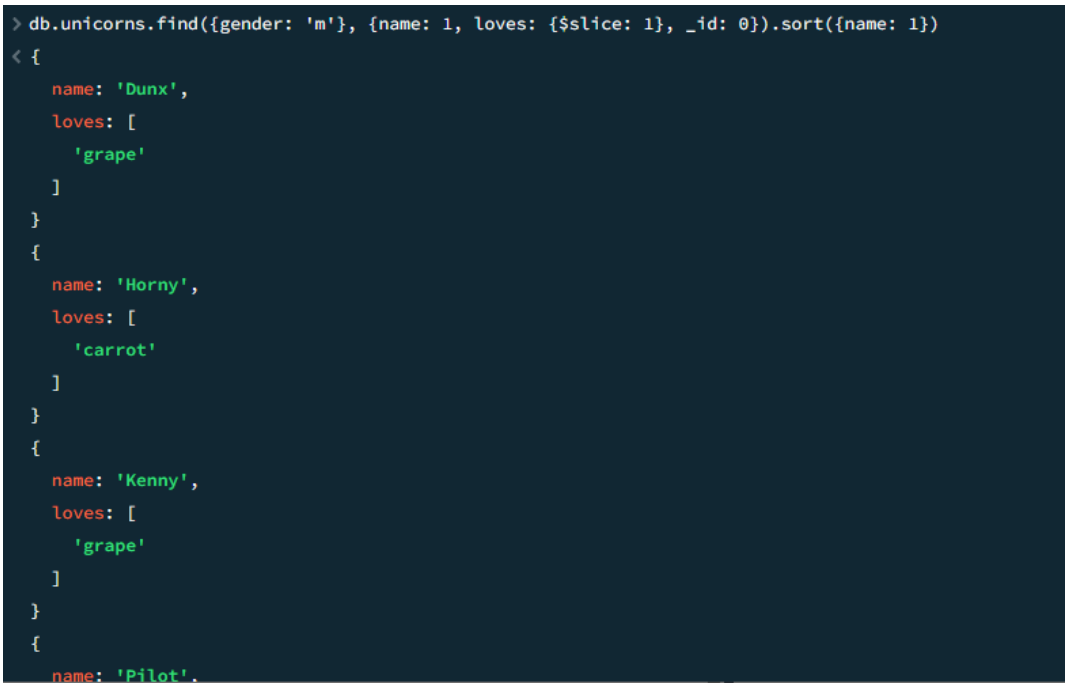

Рис. 13: Упорядоченный список имён самцов единорогов

Практическое задание 3.1.1

1. Создать коллекцию towns, включающую выписанные документы.
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.

Рис. 14: Вставка данных

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

Рис. 15: Независимые меры

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

Рис. 16: Беспартийные меры

Практическое задание 3.1.2

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.

```
> 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'});
  db.unicorns.insert ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('68dcd1547ff2b8e13e6a9135')
    }
  }
> var cursor = db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(2)
> cursor.forEach(function(obj) {print(obj.name); })
< Dunx
< Dunx
```

Рис. 17: Результат выполнения

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 600}}).count()
< 6
```

Рис. 18: Количество самок единорогов весом от 500 до 600 кг

Практическое задание 3.2.2

Вывести список предпочтений.

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

Рис. 19: Список предпочтений

Практическое задание 3.2.3

Посчитать количество особей единорогов обоих полов.

```
> db.unicorns.aggregate([{$group: {_id: "$gender", count: {$sum: 1}}}])
< {
    _id: 'f',
    count: 16
  }
  {
    _id: 'm',
    count: 21
  }
```

Рис. 20: Количество особей обоих полов

Практическое задание 3.3.1

1. Выполнить команду: *> db.unicorns.save(name: 'Barny', loves: ['grape'], weight: 340, gender: 'm')*
2. Проверить содержимое коллекции unicorns

Команды save не нашёл в текущей версии, заменил на команду insert.

```
> db.unicorns.insert({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('68dd20817ff2b8e13e6a9136')
    }
  }
> db.unicorns.find()
< {
```

Рис. 21: Результат выполнения

Практическое задание 3.3.2

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

```
> db.unicorns.updateOne({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: "Ayna"})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a9111'),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
```

Рис. 22: Запрос изменения и вывод содержимого коллекции

Практическое задание 3.3.3

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

```
> db.unicorns.updateMany({name: "Raleigh"}, {$set: {loves: ['redbull']}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 3,
    modifiedCount: 3,
    upsertedCount: 0
  }
> db.unicorns.find({name: "Raleigh"})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a9113'),
    name: 'Raleigh',
    loves: [
      'redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```

Рис. 23: Запрос изменения и вывод содержимого коллекции

Практическое задание 3.3.4

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

```
> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 22,
    modifiedCount: 22,
    upsertedCount: 0
  }
> db.unicorns.find({gender: 'm'})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910c'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
  {
    _id: ObjectId('68dccaf07ff2b8e13e6a910e'),
    name: 'Unicrom',
    loves: [
      'energon'
```

Рис. 24: Запрос изменения и вывод содержимого коллекции

Практическое задание 3.3.5

1. Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.
2. Проверить содержимое коллекции towns.

```
> db.towns.updateOne({name: 'Portland'}, {$unset: {'mayor.party': ''}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 0,
    upsertedCount: 0
  }
> db.towns.find({name: 'Portland'})
< {
    _id: ObjectId('68dccf7b7ff2b8e13e6a911c'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams'
    }
  }
```

Рис. 25: Запрос изменения и вывод содержимого коллекции

Практическое задание 3.3.6

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

```
> db.unicorns.updateMany({name: "Pilot"}, {$addToSet: {loves: "chocolate"}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 3,
    modifiedCount: 3,
    upsertedCount: 0
  }
> db.unicorns.find({name: "Pilot"})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a9115'),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon',
      'chocolate'
    ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
```

Рис. 26: Запрос изменения и вывод содержимого коллекции

Практическое задание 3.3.7

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

```
> db.unicorns.updateMany({name: "Aurora"}, {$addToSet: {loves: ['sugar', 'lemon']}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 4,
    modifiedCount: 4,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Aurora'})
< {
    _id: ObjectId('68dccaf07ff2b8e13e6a910d'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      [
        'sugar',
        'lemon'
      ]
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Рис. 27: Запрос изменения и вывод содержимого коллекции

Практическое задание 3.4.1

1. Создать коллекцию towns, содержащую предоставленые документы.
2. Удалить документы с беспартийными мэрами.
3. Проверить содержание коллекции.
4. Очистить коллекцию.
5. Посмотреть список доступных коллекций,

```
> db.towns.insertMany([{name: "Punxsutawney ",
  popujatiuon: 6200,
  last_sensus: ISODate("2008-01-31"),
  famous_for: ["phil the groundhog"],
  mayor: {
     name: "Jim Wehrle"
  }},
  {name: "New York",
  popujatiuon: 22200000,
  last_sensus: ISODate("2009-07-31"),
  famous_for: ["status of liberty", "food"],
  mayor: {
     name: "Michael Bloomberg",
  party: "I"}},
  {name: "Portland",
  popujatiuon: 528000,
  last_sensus: ISODate("2009-07-20"),
  famous_for: ["beer", "food"],
  mayor: {
     name: "Sam Adams",
  party: "D"}}
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('68dd23a67ff2b8e13e6a9137'),
      '1': ObjectId('68dd23a67ff2b8e13e6a9138'),
      '2': ObjectId('68dd23a67ff2b8e13e6a9139')
    }
  }
```

Рис. 28: Создание коллекции towns

Рис. 29: Удаление беспартийных мэров и проверка содержания коллекции

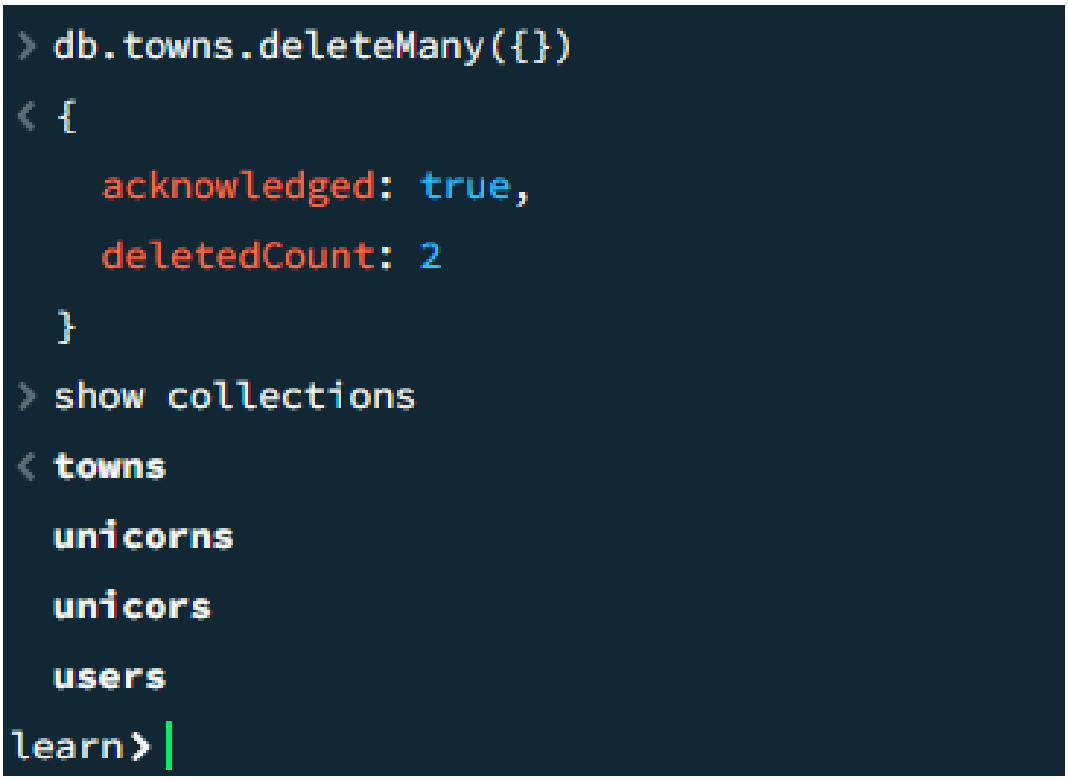

Рис. 30: Список доступных коллекций

Практическое задание 4.1.1

1. Создать коллекцию зон обидания еднирогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.

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

3. Проверьте содержание коллекции едиорогов.

```
> db.habitat.insertMany([
    {
      _id: "forest",
      full_name: "enchanted forest",
      description: "Dense woodlands with abundant magical energy are common habitats."
    },
    {
      _id: "peaks",
      full_name: "mountain valleys and peaks",
      description: "Remote, high-altitude regions offer a unique sanctuary."
    },
    {
      _id: "meadows",
      full_name: "high-altitude meadows",
      description: "Rich expanses of green, offering abundant grasses, herbs, and flowers for grazing."
    }
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': 'forest',
      '1': 'peaks',
      '2': 'meadows'
    }
  }
```

Рис. 31: Создание коллекции зон обитания

```
> db.unicorns.update({name: "Nimue"}, {$set: {habitat: {$ref: "habitat", $id: "peaks"}}})
< DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.update({name: "Dunx"}, {$set: {habitat: {$ref: "habitat", $id: "meadows"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Рис. 32: Добавление нескольким единорогам ссылку на зону обитания

```
> db.unicorns.find({name: "Nimue"})
< {
    _id: ObjectId('68dd24db7ff2b8e13e6a9143'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f',
    habitat: DBRef('habitat', 'peaks')
  }
```

Рис. 33: Содержимое коллекции единорогов

Практическое задание 4.2.1

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

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

Рис. 34: Создание индекса

Практическое задание 4.3.1

1. Получить информацию о всех индексах коллекции unicorns.
2. Удалить все индексы, кроме индекса для идентификатора.
3. Попробовать удалить индекс для идентификатора.

```
> db.unicorns.createIndex({name: 1}, {unique: true})
< name_1
> db.unicorns.getIndexes()
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
  ]
> db.unicorns.dropIndexes()
< {
    nIndexesWas: 2,
    msg: 'non-_id indexes dropped for collection',
    ok: 1
  }
> db.unicorns.dropIndex("_id_")
✖ ▸ MongoServerError[InvalidOptions]: cannot drop _id index
```

Рис. 35: Выполнение задания

Практическое задание 4.4.1

1. Создать объемную коллекцию numbers, задействовав курсор: *for(i = 0; i < 100000; i++)db.numbers.insert(value: i)*
2. Выбрать последних четыре документа.
3. Проанализировать план выполнения запроса 2.
4. Создать индекс для ключа value.
5. Получить информацию о всех индексах коллекции numbres.
6. Выполнить запрос 2.
7. Проанилизоровать план выполнения запроса с установленным индексом.
8. Сравнить время выполнения запросов с индексом и без.

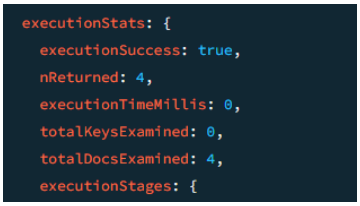

Рис. 36: Время выполнения без индекса

Рис. 37: Время выполнения с индексом

Как видно на такой выборке недостаточно увидеть разницу, так как оба запроса отрабатывают 0 мс, но, для большего размера данных, версия с индексом будет оотрабатывать быстрее.

Контрольные вопросы

- **Как используется оператор точка?** Оператор точка используются для перехода по вложенным документам, их полям.
- **Как можно использовать курсос?** Курсос является своего рода итератором по результату выборки, с ним можно обрабатывать все эти документы, не выводя изначальный список. (можно применять методы limit, map, forEach и тд).
- **Какие возможности агрегирования данных существуют в MongoDB?** Есть конвееры (pipelines), которые позволяют обрабатывать документ по этапно, передавая результат от этапа к этапу. Также есть простые функции для выполнения аргегирования: count, distinct.
- **Какая из функций save или update более детально позволит настроить редактирование документов коллекции?** update подходить лучше для редактирования документов, ведь он имеет специальные операторы обновления: $set, $inc, ..., позволяя гибко изменять поля документов. Метод save просто сохраняет новый документ, если документ такой уже есть, он его перезаписывает, но не позволяет детально редактировать поля документа.
- **Как происходит удаление документов из коллекции по умолчанию?** С помощью методов коллекции deleteOne и deleteMany, они позволяют передавать в себя фильтр, по которому будет удалён один/все подходящие под этот фильтр документы.
- **Назовите способы связывания коллекций в MongoDB.** Можно встраивать документы, т.е. хранить один документ внутри другого. Также есть ссылки, документ содержит поле, которые ссылается на документ в другой коллекции. И есть оператор $lookup, являющимся аналогом LEFT JOIN.
- **Сколько индексов можно установить на одну коллекцию в БД MongoDB?** 64 индекса на одну коллекцию.
- **Как получить информацию о всех индексах базы данных MongoDB?** С помощью метода коллекции getIndexes().

Выводы

При выполнения работы были получены базовые навыки работы с MongoDB, изучены её возможности и поведения, а также отличия от реляционных баз данных.