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: Гордеев А. А.

Факультет: ПИН

Группа:К3239

Преподаватель: Говорова М. М.

Санкт-Петербург 2025

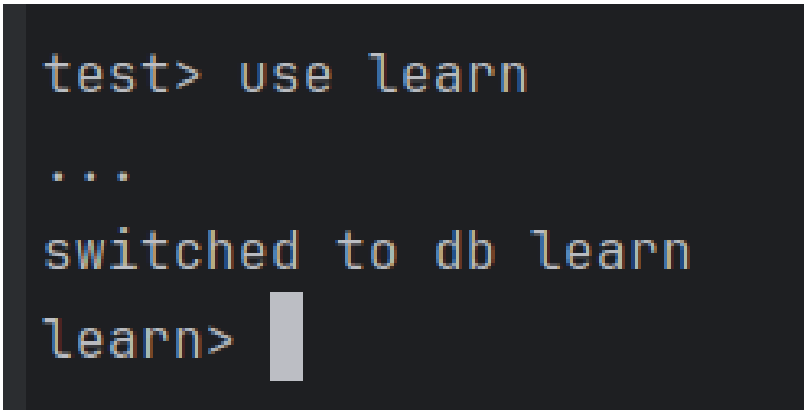
ОГЛАВЛЕНИЕ

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:

- Создайте базу данных learn.



- Заполните коллекцию единорогов unicorns.

```
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'}
... ]);
```

- Используя второй способ, вставьте в коллекцию единорогов документ.

```
learn> let unicornDocument = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165};
... db.unicorns.insertOne(unicornDocument);
{
  acknowledged: true,
  insertedId: ObjectId('682b17e06c43f18d16d861eb')
}
```

- Проверьте содержимое коллекции с помощью метода find.

```
learn> db.unicorns.find();
[
  {
    _id: ObjectId('682b13186c43f18d16d861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
```

Практическое задание 2.2.1:

- Сформируйте запросы для вывода списков самцов и самок единорогов.

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1});
[
  {
    _id: ObjectId('682b17e06c43f18d16d861eb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
```

- Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3);
[
  {
    _id: ObjectId('682b13186c43f18d16d861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e8'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

```
learn> db.unicorns.findOne({gender: 'f', loves: 'carrot'});
{
  _id: ObjectId('682b13186c43f18d16d861e1'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1);
[
  {
    _id: ObjectId('682b13186c43f18d16d861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

- Модифицируйте запрос для вывода списков самцов единорогов,

исключив из результата информацию о предпотениях и поле.

```
learn> db.unicorns.find({gender: 'm'}, {loves: 0, _id: 0});
[
  { name: 'Horny', weight: 600, gender: 'm', vampires: 63 },
  { name: 'Unicrom', weight: 984, gender: 'm', vampires: 182 },
  { name: 'Roooooodles', weight: 575, gender: 'm', vampires: 99 },
  { name: 'Kenny', weight: 690, gender: 'm', vampires: 39 },
  { name: 'Raleigh', weight: 421, gender: 'm', vampires: 2 },
  { name: 'Pilot', weight: 650, gender: 'm', vampires: 54 },
  { name: 'Dunx', weight: 704, gender: 'm', vampires: 165 }
]
```

- Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1});
[
  {
    _id: ObjectId('682b17e06c43f18d16d861eb'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('682b13186c43f18d16d861ea'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
```

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

```
learn> 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' ] },
  { name: 'Solnara', loves: [ 'apple' ] },
  { name: 'Ayna', loves: [ 'strawberry' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Leia', loves: [ 'apple' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Nimue', loves: [ 'grape' ] },
  { name: 'Dunx', loves: [ 'grape' ] }
]
```

Практическое задание 2.3.1:

- Вывести список самок единорогов весом от полутонны 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'
  }
]
```

- Вывести список самцов единорогов весом от полутонны 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
  }
]
```

- Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}});
[
  {
    _id: ObjectId('682b13186c43f18d16d861ea'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

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

```
learn> db.unicorns.find(
...   {gender: 'm'},
...   {name: 1, loves: {$slice: 1}, _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' ] }
]
```

Практическое задание 3.1.1:

- Создайте коллекцию towns, включающую следующие документы.

```
learn> db.towns.insertMany([
...   {
...     name: "Punxsutawney",
...     population: 6200,
...     last_census: new Date("2008-01-31"),
...     famous_for: [""],
...     mayor: {
...       name: "Jim Wehrle"
...     }
...   },
...   {
...     name: "New York",
...     population: 22200000,
...     last_census: new Date("2009-07-31"),
...     famous_for: ["statue of liberty", "food"],
...     mayor: {
...       name: "Michael Bloomberg",
...       party: "I"
...     }
...   },
...   {
```

- 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' }
  }
]
```

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

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

- Сформировать функцию для вывода списка самцов единорогов. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке. Вывести результат, используя forEach.

```
learn> fn = function() { return this.gender == "m"; }
[Function: fn]
learn> var cursor = db.unicorns.find({$where: fn}).sort({name: 1}).limit(2)

learn> cursor.forEach(function(obj){ print(obj.name); });
Dunx
Horny
```

Практическое задание 3.2.1:

- Вывести количество самок единорогов весом от полутонны до 600 кг.

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

- Вывести список предпочтений.

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

- Посчитать количество особей единорогов обоих полов.

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

Практическое задание 3.3.1:

- Выполнить команду. Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.insertOne({
...   name: 'Barny',
...   loves: ['grape'],
...   weight: 340,
...   gender: 'm'
... });
{
  acknowledged: true,
  insertedId: ObjectId('682b20dc6c43f18d16d861ef')
}
```

```
learn> db.unicorns.find({name: 'Barny'});
[
  {
    _id: ObjectId('682b20dc6c43f18d16d861ef'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
```

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

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

- Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Ayna'});
[
  {
    _id: ObjectId('682b13186c43f18d16d861e5'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

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

```
learn> db.unicorns.updateOne(
...   {name: 'Raleigh', gender: 'm'},
...   {$addToSet: {loves: 'redbull'}}
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

- Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Raleigh'});
[
  {
    _id: ObjectId('682b13186c43f18d16d861e7'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

- 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
}
```

- Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({gender: 'm'});
[
  {
    _id: ObjectId('682b13186c43f18d16d861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e2'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
```

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

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

- Проверить содержимое коллекции towns.

```
learn> db.towns.find({name: 'Portland'});
[
  {
    _id: ObjectId('682b1ce66c43f18d16d861ee'),
    name: 'Portland',
    population: 528000,
    last_census: ISODate('2009-07-20T00:00:00.000Z
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
```

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

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

- Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Pilot'});
[
  {
    _id: ObjectId('682b13186c43f18d16d861e9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

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

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

- Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.find({name: 'Aurora'});
[
  {
    _id: ObjectId('682b13186c43f18d16d861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Практическое задание 3.4.1:

- Создайте коллекцию towns, включающую следующие документы.

```
learn> db.towns.drop();
... db.towns.insertMany([
...   {
...     name: "Punxsutawney ",
...     popujatiuon: 6200,
...     last_sensus: new Date("2008-01-31"),
...     famous_for: ["phil the groundhog"],
...     mayor: {
...       name: "Jim Wehrle"
...     }
...   },
...   {
...     name: "New York",
...     popujatiuon: 22200000,
...     last_sensus: new Date("2009-07-31"),
...     famous_for: ["status of liberty", "food"],
...     mayor: {
...       name: "Michael Bloomberg",
...       party: "I"
...     }
...   },
...   {
...     name: "Portland",
...     popujatiuon: 528000,
...     last_sensus: new Date("2009-07-20"),
...     famous_for: ["beer", "food"],
...     mayor: {
...       name: "Sam Adams",
```

- Удалите документы с беспартийными мэрами.

```
learn> db.towns.deleteMany({"mayor.party": "I"});
{ acknowledged: true, deletedCount: 1 }
```

- Проверьте содержание коллекции.

```
learn> db.towns.find();
[
  {
    _id: ObjectId('682b22cb6c43f18d16d861f0'),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('682b22cb6c43f18d16d861f2'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

- Очистите коллекцию.

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

- Просмотрите список доступных коллекций.

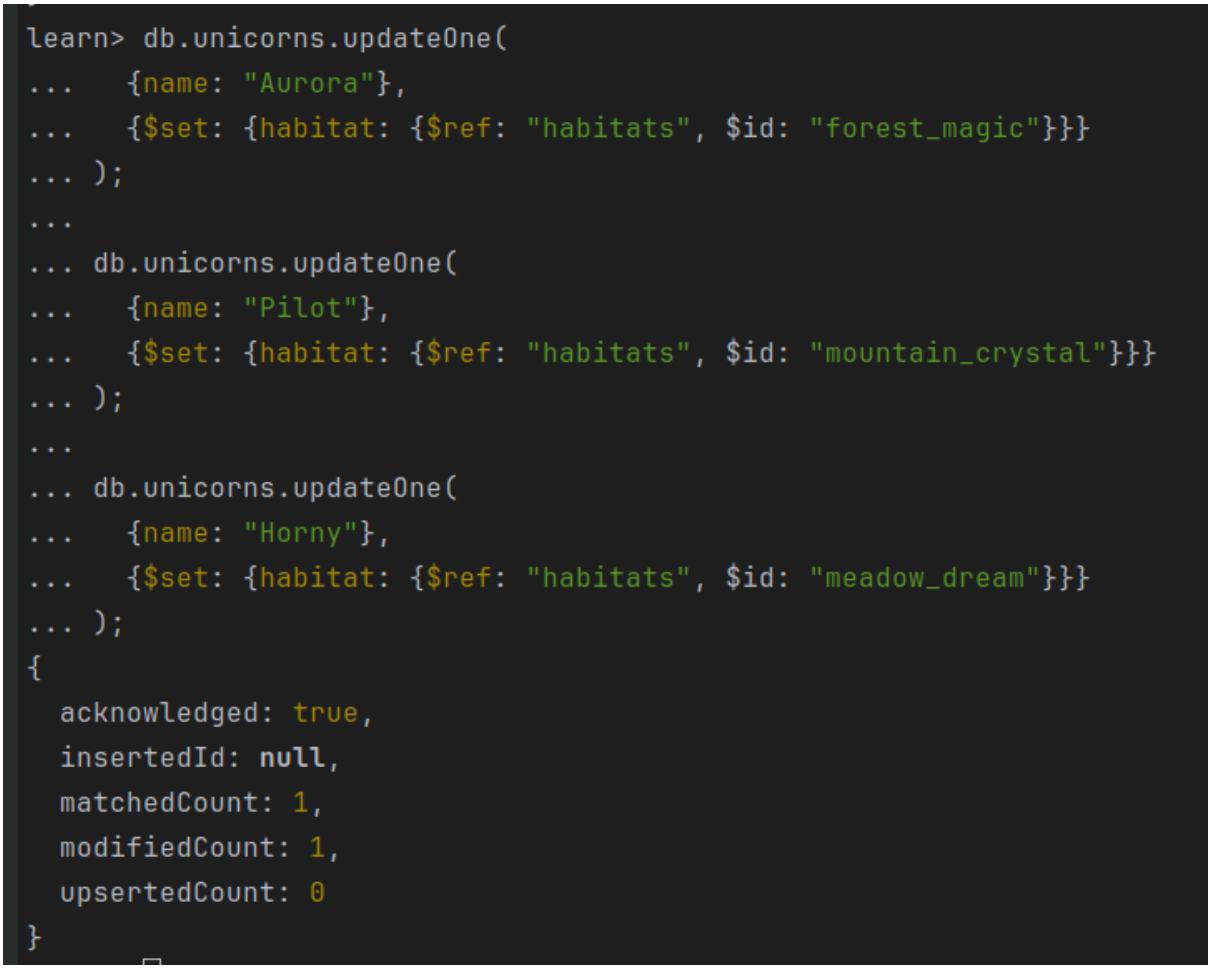
```
learn> show collections;
towns
unicorns
```

Практическое задание 4.1.1:

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

```
learn> db.habitats.insertMany([
...   {_id: "forest_magic", name: "Magic Forest", description: "A forest full of glitter and ancient trees."},
...   {_id: "mountain_crystal", name: "Crystal Mountain", description: "High peaks made of pure crystal, reflecting rainbows.
"},
...   {_id: "meadow_dream", name: "Dreamy Meadow", description: "Endless fields of soft grass and sweet flowers."}
... ]);
{
  acknowledged: true,
  insertedIds: { '0': 'forest_magic', '1': 'mountain_crystal', '2': 'meadow_dream' }
}
```

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



```
learn> db.unicorns.updateOne(
...   {name: "Aurora"},
...   {$set: {habitat: {$ref: "habitats", $id: "forest_magic"}}}
... );
...
... db.unicorns.updateOne(
...   {name: "Pilot"},
...   {$set: {habitat: {$ref: "habitats", $id: "mountain_crystal"}}}
... );
...
... db.unicorns.updateOne(
...   {name: "Horny"},
...   {$set: {habitat: {$ref: "habitats", $id: "meadow_dream"}}}
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

- Проверьте содержание коллекции едиорогов.

```
learn> db.unicorns.find({name: {$in: ["Aurora", "Pilot", "Horny"]}});
[
  {
    _id: ObjectId('682b13186c43f18d16d861e0'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habitat: DBRef('habitats', 'meadow_dream')
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e1'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat: DBRef('habitats', 'forest_magic')
  },
  {
    _id: ObjectId('682b13186c43f18d16d861e9'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
```

Практическое задание 4.2.1:

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

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

- 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 }
]
```

- Удалите все индексы, кроме индекса для идентификатора.

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

- Попытайтесь удалить индекс для идентификатора.

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

Практическое задание 4.4.1:

- Создайте объемную коллекцию numbers, задействовав курсор.

```
learn> for(let i=0;i<10000;i++) {db.numbers.insertOne({value:i});}
{
  acknowledged: true,
  insertedId: ObjectId('682b26bb6c43f18d16d9785a')
}
```

- Выберите последних четыре документа.

```
learn> db.numbers.find().sort({value: -1}).limit(4);
[
  { _id: ObjectId('682b25fd6c43f18d16d8e442'), value: 33359 },
  { _id: ObjectId('682b25fd6c43f18d16d8e441'), value: 33358 },
  { _id: ObjectId('682b25fd6c43f18d16d8e440'), value: 33357 },
  { _id: ObjectId('682b25fd6c43f18d16d8e43f'), value: 33356 }
]
```

- Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра executionTimeMillis) 107

```
learn> db.numbers.explain("executionStats").find().sort({value: -1}).limit(4);
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: '0B9816AB',
    planCacheKey: '0B9816AB',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'SORT',
      sortPattern: { value: -1 },
      memLimit: 104857600,
      limitAmount: 4,
      type: 'simple',
      inputStage: { stage: 'COLLSCAN', direction: 'forward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 107,
    totalKeysExamined: 0,
    totalDocsExamined: 71272,
```

- Создайте индекс для ключа value.

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

- Получите информацию о всех индексах коллекции numbres.

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

- Выполните запрос 2. Значение 2. Данные закешировались и ответ стал мгновенным.

Вывод

В ходе выполнение лабораторной работы я научился работать с MongoDB, выполнять CRUD-операции, узнать про добавление, поиск, удаление, сортировки, операторы, как связывать таблицы и индексацию.