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: Даньшин С. А.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

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.

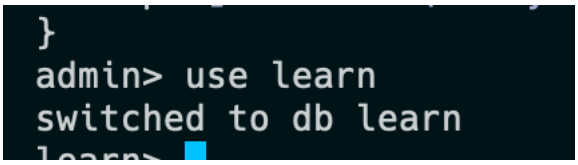
Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4+, 6.0.6 (текущая).

Выполнение работы

2.1.1

1) Создайте базу данных learn.



2) *Заполните коллекцию единорогов unicorns:*

3) *Используя второй способ, вставьте в коллекцию единорогов 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> db.unicorns.insert(document)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657b5d2a2d381c7dc591cd0a') }
}
```

4) *Проверьте содержимое коллекции с помощью метода find:*

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfe'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccff'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd00'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd01'),
```

2.2.1

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

Самки - db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfe'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd02'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd05'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

Самцы – db.unicorns.find({gender: 'm'}).sort({name: 1})

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    _id: ObjectId('657b5d2a2d381c7dc591cd0a'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd03'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('657b5c792d381c7dc591cd06'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657b5c792d381c7dc591cd08'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
```

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"})

```
learn> db.unicorns.findOne({gender: 'f', loves: "carrot"})
{
  _id: ObjectId('657b5c6b2d381c7dc591ccfe'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
  weight: 450,
  gender: 'f',
  vampires: 43,
  location: DBRef('locations', 'STL')
}
```

db.unicorns.find({gender: 'f', loves: "carrot"}).limit(1)

```
learn> db.unicorns.find({gender: 'f', loves: "carrot"}).limit(1)
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfe'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    location: DBRef('locations', 'STL')
  }
]
```

2.2.2

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

Самцы - db.unicorns.find({gender: 'm'}, {loves: 0, vampires: 0}).sort({name: 1})

```
learn> db.unicorns.find({gender: 'm'}, {loves: 0, vampires: 0}).sort({name: 1})
[
  {
    _id: ObjectId('657b5d2a2d381c7dc591cd0a'),
    name: 'Dunx',
    weight: 704,
    gender: 'm'
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfd'),
    name: 'Horny',
    weight: 600,
    gender: 'm'
  },
```

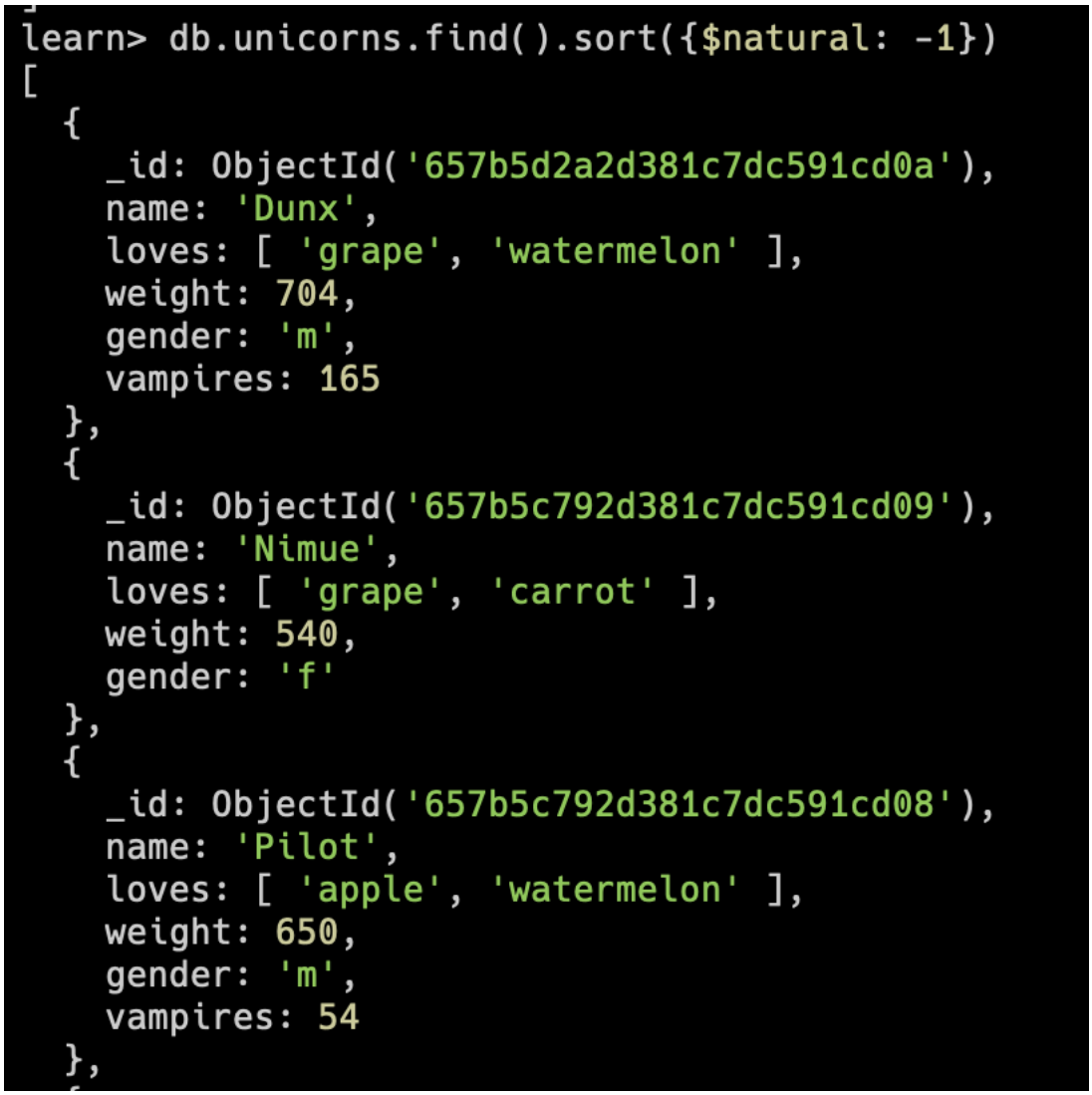
Самки - db.unicorns.find({gender: 'f'}, {loves: 0, vampires: 0}).sort({name: 1}).limit(3)

```
learn> db.unicorns.find({gender: 'f'}, {loves: 0, vampires: 0}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfe'),
    name: 'Aurora',
    weight: 450,
    gender: 'f'
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd02'),
    name: 'Ayna',
    weight: 733,
    gender: 'f'
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd05'),
    name: 'Leia',
    weight: 601,
    gender: 'f'
  }
```

2.2.3

Вывести список единорогов в обратном порядке добавления.

db.unicorns.find().sort({$natural: -1})



2.2.4

Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор

db.unicorns.find({}, {_id: 0, loves: {$slice: [0, 1]}})

```
learn> db.unicorns.find({}, {_id: 0, loves: {$slice: [0, 1]}})
[
  {
    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',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
```

2.3.1

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

db.unicorns.find({weight: {$lt: 700, $gt: 500}, gender: 'f'}, {_id: 0})

```
learn> db.unicorns.find({weight: {$lt: 700, $gt: 500}, gender: 'f'}, {_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'
  },
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

2.3.2

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

db.unicorns.find({loves: {$all: ['lemon', 'grape']}, weight: {$gt: 500}}, {_id: 0})

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

2.3.3

Найти всех единорогов, не имеющих ключ `vampires`

db.unicorns.find({vampires: {$exists: false}})

2.3.4

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

db.unicorns.find({gender: 'm'}, {name: 1, loves: {$slice: [0, 1]}, _id: 0}).sort({name: 1})

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

3.1.1

1) **Создайте коллекцию towns, включающую следующие документы:**

   db.towns.insert(…)

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

   db.towns.find({'mayor.party': 'I'}, {'name': 1, 'mayor.name': 1})

```
learn> db.towns.find({'mayor.party': 'I'}, {'name': 1, 'mayor.name': 1})
[
  {
    _id: ObjectId('657b6aa22d381c7dc591cd0c'),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg' }
  }
]
```

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

   db.towns.find({'mayor.party': {$exists: false}}, {'name': 1, mayor: 1})

```
learn> db.towns.find({'mayor.party': {$exists: false}}, {'name': 1, mayor: 1})
[
  {
    _id: ObjectId('657b6a922d381c7dc591cd0b'),
    name: 'Punxsutawney ',
    mayor: { name: 'Jim Wehrle' }
  }
]
```

3.1.2.

1) *Сформировать функцию для вывода списка самцов единорогов.*

function get_male_unicorns() { return db.unicorns.find({gender: "m"}) }

```
learn> get_male_unicorns()
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfd'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccff'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd00'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
```

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

- var cursor = get_male_unicorns();null;
- cursor.sort({name: 1}).limit(2);null;

4) *Вывести результат, используя forEach.*

- cursor.forEach(function(obj){

  print(obj)

  })

```
learn> cursor = get_male_unicorns();null;
null
learn> var cursor = get_male_unicorns();null;
null
learn> cursor.sort({name: 1}).limit(2);null;
null
learn> cursor.forEach(function(obj){
... print(obj)
... })
{
  _id: ObjectId('657b5d2a2d381c7dc591cd0a'),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId('657b5c6b2d381c7dc591ccfd'),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

```
learn> db.unicorns.find({weight: {$lt: 600, $gt: 500}, gender: 'f'}).count()
3
```

3.2.2

Вывести список предпочтений.

db.unicorns.distinct("loves")

3.2.3

Посчитать количество особей единорогов обоих полов.

db.unicorns.aggregate({$group: {_id: "$gender", count: {$sum: 1}}})

```
learn> db.unicorns.aggregate({$group: {_id: "$gender", count: {$sum: 1}}})
[ { _id: 'f', count: 6 }, { _id: 'm', count: 8 } ]
```

3.3.1

1. *Выполнить команду*

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

```
learn> db.unicorns.insertOne({ name: 'Barny', loves: ['grape'], weight: 340, gender: 'm' })
{
  acknowledged: true,
  insertedId: ObjectId('657e1d77acd03de552f73568')
}
```

2. *Проверить содержимое коллекции* `unicorns`

db.unicorns.find(ObjectId('657e1d77acd03de552f73568'))

```
learn> db.unicorns.find(ObjectId('657e1d77acd03de552f73568'))
[
  {
    _id: ObjectId('657e1d77acd03de552f73568'),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
```

3.3.2

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

db.unicorns.updateOne({ name: "Ayna" }, [{ $set: { weight: 800, vampires: 51}}]

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

2. *Проверить содержимое коллекции* `unicorns`.

db.unicorns.find({ name: "Ayna" })

```
learn> db.unicorns.find({ name: "Ayna" })
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd02'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

3.3.3

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

db.unicorns.updateOne({name: "Raleigh"}, [{$set: {loves: ["redbull"]}}]

```
learn> db.unicorns.updateOne({name: "Raleigh"}, [{$set: {loves: ["redbull"]}}])
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. *Проверить содержимое коллекции* `unicorns`.

db.unicorns.find({ name: "Raleigh" })

```
learn> db.unicorns.find({ name: "Raleigh" })
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd04'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

3.3.4

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

db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})

```
learn> db.unicorns.updateMany({gender: 'm'}, {$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 9,
  modifiedCount: 9,
  upsertedCount: 0
}
```

2. *Проверить содержимое коллекции* `unicorns`.

db.unicorns.find({gender: 'm'})

```
learn> db.unicorns.find({ name: "Raleigh" })
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd04'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
```

```
learn> db.unicorns.find({ name: "Raleigh" })
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd04'),
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  }
]
```

3.3.5

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

db.towns.updateOne({name: "Portland"}, [{$unset: "mayor.party"}])

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

2. *Проверить содержимое коллекции* `towns`

db.towns.find({name: "Portland"})

```
learn> db.towns.find({name: "Portland"})
[
  {
    _id: ObjectId('657b6aab2d381c7dc591cd0d'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

3.3.6

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

db.unicorns.updateOne({name: "Pilot"}, {$push: {loves: "chocolate"}})

```
learn> db.unicorns.updateOne({name: "Pilot"}, {$push: {loves: "chocolate"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. *Проверить содержимое коллекции* `unicorns`.

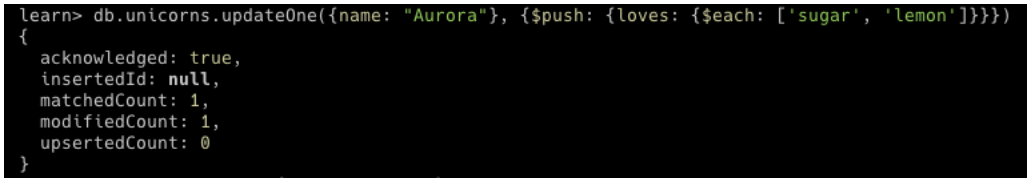
db.unicorns.find({name: "Pilot"})

```
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('657b5c792d381c7dc591cd06'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  },
```

3.3.7

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

db.unicorns.updateOne({name: "Aurora"}, {$push: {loves: {$each: ['sugar', 'lemon']}}})



2. *Проверить содержимое коллекции* `unicorns`.

db.unicorns.find({name: "Aurora"})

3.4.1

1) *Создайте коллекцию towns, включающую следующие документы:*

```
learn> db.towns.find()
[
  {
    _id: ObjectId('657f2b8cacd03de552f73569'),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('657f2b96acd03de552f7356a'),
    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('657f2ba6acd03de552f7356b'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

2) *Удалите документы с беспартийными мэрами.*

db.towns.deleteMany({$not: {$exists: "mayor.party"}})

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

3) *Проверьте содержание коллекции.*

db.towns.find()

```
learn> db.towns.find()
[
  {
    _id: ObjectId('657f2b96acd03de552f7356a'),
    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('657f2ba6acd03de552f7356b'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

4) *Очистите коллекцию.*

db.towns.drop()

```
learn> db.towns.drop()
true
```

5) *Просмотрите список доступных коллекций.*

show collections

```
learn> show collections
unicorns
```

4.1.1

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

db.locations.insertMany([

{"_id": "MGL", "name": "Mystic Glade", "description": "A hidden glade surrounded by ancient forests, where magic permeates the air."},

{"_id": "CRY", "name": "Crystal Haven", "description": "A serene valley adorned with crystalline formations, reflecting the colors of the unicorns' enchantments."},

{"_id": "STL", "name": "Starlight Meadow", "description": "A meadow bathed in perpetual starlight, where unicorns graze under the celestial glow."},

])

```
learn> db.locations.insertMany([
...     {"_id": "MGL", "name": "Mystic Glade", "description": "A hidden glade surrounded by ancient forests, where magic p
ermeates the air."},
...     {"_id": "CRY", "name": "Crystal Haven", "description": "A serene valley adorned with crystalline formations, refle
cting the colors of the unicorns' enchantments."},
...     {"_id": "STL", "name": "Starlight Meadow", "description": "A meadow bathed in perpetual starlight, where unicorns
graze under the celestial glow."},
... ])
{
  acknowledged: true,
  insertedIds: { '0': 'MGL', '1': 'CRY', '2': 'STL' }
}
```

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

db.unicorns.updateMany({loves: {$in: ['lemon']}}, {$set: {location: {$ref: "locations", $id: 'STL'}}})

```
learn> db.unicorns.updateMany({loves: {$in: ['lemon']}}, {$set: {location: {$ref: "locations", $id: 'STL'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
```

3. *Проверьте содержание коллекции едиорогов.*

db.unicorns.find({loves: {$in: ['lemon']}})

```
learn> db.unicorns.find({loves: {$in: ['lemon']}})
[
  {
    _id: ObjectId('657b5c6b2d381c7dc591ccfe'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    location: DBRef('locations', 'STL')
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd02'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51,
    location: DBRef('locations', 'STL')
  },
  {
    _id: ObjectId('657b5c6b2d381c7dc591cd03'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 44,
    location: DBRef('locations', 'STL')
  }
]
```

4.2.1

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

db.numbers.find({value: {$in: [9999, 9998, 9997]}})

```
MongoServerError: Index build failed: 5f66acf9-e1f9-4abb-9dd7-5dbdda9eab98: Collection learn.unicorns ( 7ce59a92-f657-434d
-8de7-a32ea9319176 ) :: caused by :: E11000 duplicate key error collection: learn.unicorns index: name_1 dup key: { name:
"Nimue" }
```

В коллекции было 2 единорога с именем ‘Nimue’

db.unicorns.deleteOne({name: "Nimue"})

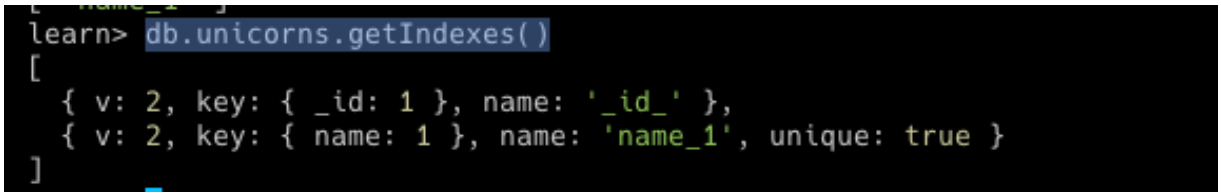
Попробуем еще раз навесить индекс

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

4.3.1

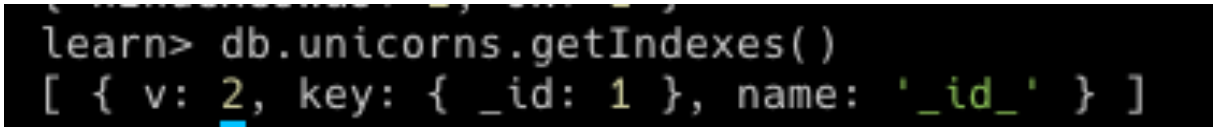
1. *Получите информацию обо всех индексах коллекции* `unicorns`.

db.unicorns.getIndexes()



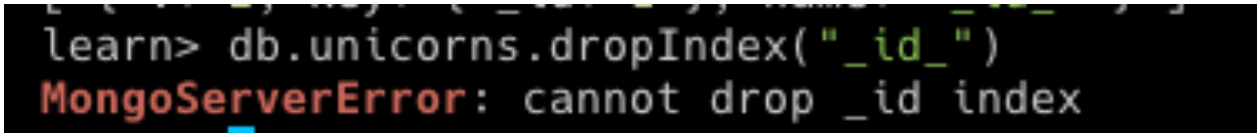
2. *Удалите все индексы, кроме индекса для идентификатора.*

db.unicorns.dropIndex("name_1")



3. *Попытайтесь удалить индекс для идентификатора.*

db.unicorns.dropIndex("_id_")



4.4.1

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

```
for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
```

2. *Выберите последних четыре документа.*

db.numbers.find({value: {$in: [9999, 9998, 9997]}})

```
learn> db.numbers.find({value: {$in: [9999, 9998, 9997]}})
[
  { _id: ObjectId('657f3a7aacd03de552f75c79'), value: 9997 },
  { _id: ObjectId('657f3a7aacd03de552f75c7a'), value: 9998 },
  { _id: ObjectId('657f3a7aacd03de552f75c7b'), value: 9999 }
]
```

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

db.numbers.explain("executionStats").find({value: {$in: [9999, 9998, 9997]}})

```
executionStats: {
  executionSuccess: true,
  nReturned: 3,
  executionTimeMillis: 18,
```

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

db.numbers.ensureIndex({value: 1})

```
learn> db.numbers.ensureIndex({value: 1})
[ 'value_1' ]
```

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

db.numbers.getIndexes()

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

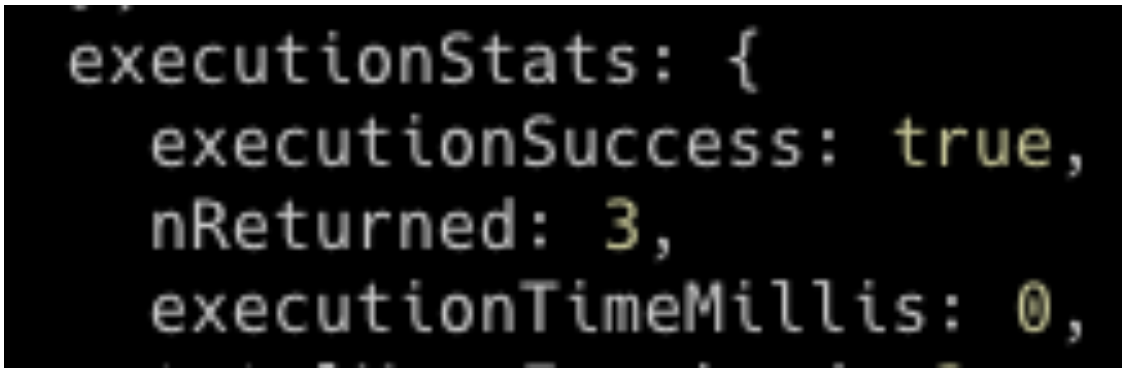
6. Выполните запрос 2.

db.numbers.find({value: {$in: [9999, 9998, 9997]}})

```
learn> db.numbers.find({value: {$in: [9999, 9998, 9997]}})
[
  { _id: ObjectId('657f3a7aacd03de552f75c79'), value: 9997 },
  { _id: ObjectId('657f3a7aacd03de552f75c7a'), value: 9998 },
  { _id: ObjectId('657f3a7aacd03de552f75c7b'), value: 9999 }
]
```

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

db.numbers.explain("executionStats").find({value: {$in: [9999, 9998, 9997]}})



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

Супермега ускорение для запросов с фильтрацией по конкретным значениям. Если бы мы выбирали 4 последних документа с помощью сортировки, то индексы бы никак не ускорили выполнение запроса.

Вывод

В ходе лабораторной работы были освоены практические навыки по созданию, функций, документов и коллекций в СУБД MongoDB. Были созданы функции на выборку данных, а также были созданы запросы для удаления, замены и вставки данных разными способами. Также был создан курсор и различные запросы на выборку данных.