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: Федорин К.В.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

ЛАБОРАТОРНАЯ РАБОТА №6

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.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4+, 7.0.4

2.1 ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

Практическое задание 2.1.1:

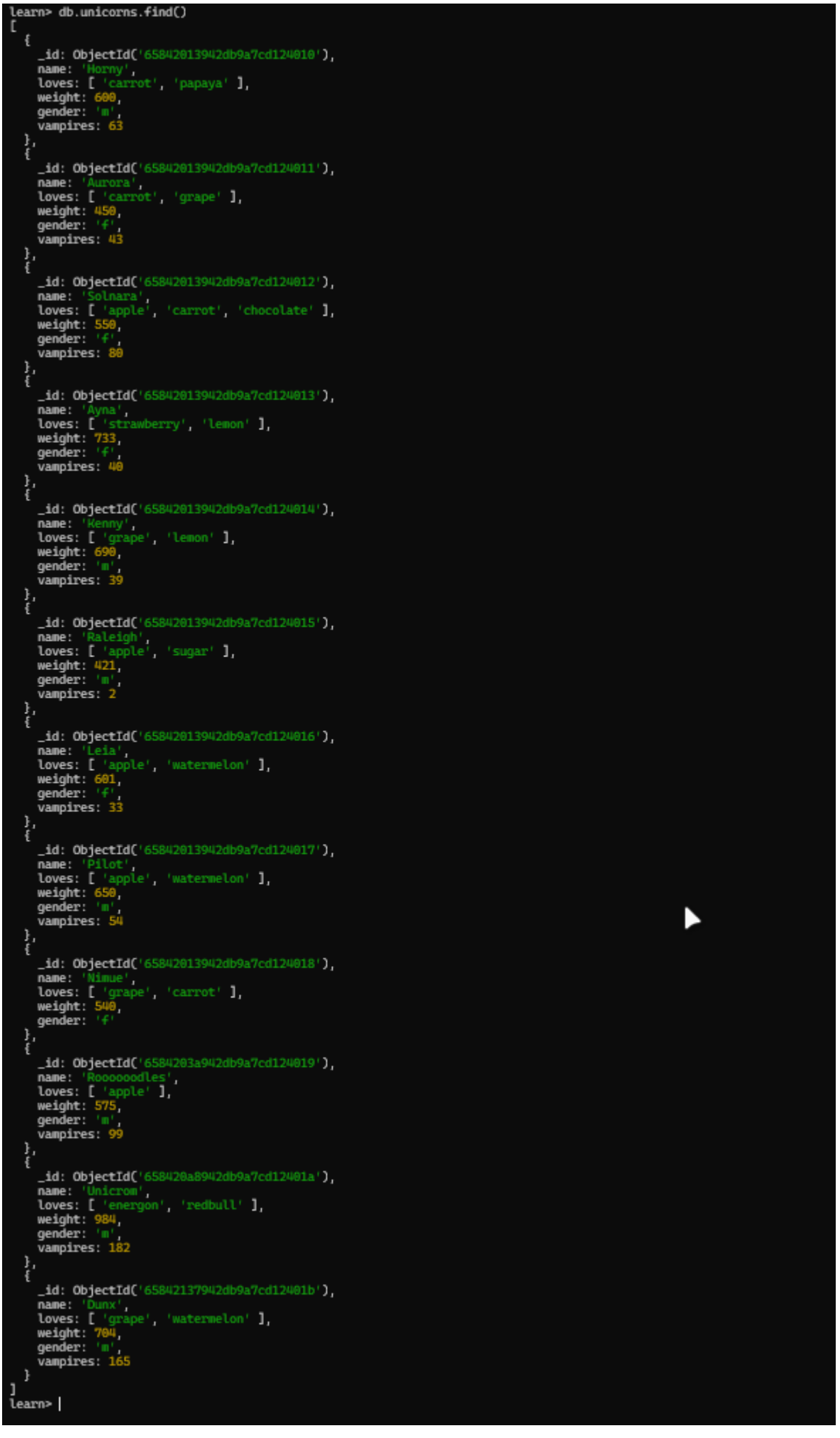
Исполнены все команды по вставке единорогов, пример вызова одной из них:

```
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658420a8942db9a7cd12401a') }
}
```

Второй способ:

```
learn> doc = ({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(doc)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65842137942db9a7cd12401b') }
}
```

Вывод:


learn> db.unicorns.find()
[
{
_id: ObjectId('658420139420db9a7cd124010'),
name: 'Horny',
loves: ['carrot', 'papaya'],
weight: 600,
gender: 'm',
vampires: 63
},
{
_id: ObjectId('658420139420db9a7cd124011'),
name: 'Aurora',
loves: ['carrot', 'grape'],
weight: 450,
gender: 'f',
vampires: 43
},
{
_id: ObjectId('658420139420db9a7cd124012'),
name: 'Solnara',
loves: ['apple', 'carrot', 'chocolate'],
weight: 550,
gender: 'f',
vampires: 80
},
{
_id: ObjectId('658420139420db9a7cd124013'),
name: 'Ayna',
loves: ['strawberry', 'lemon'],
weight: 733,
gender: 'f',
vampires: 40
},
{
_id: ObjectId('658420139420db9a7cd124014'),
name: 'Kenny',
loves: ['grape', 'lemon'],
weight: 690,
gender: 'm',
vampires: 39
},
{
_id: ObjectId('658420139420db9a7cd124015'),
name: 'Raleigh',
loves: ['apple', 'sugar'],
weight: 421,
gender: 'm',
vampires: 2
},
{
_id: ObjectId('658420139420db9a7cd124016'),
name: 'Leia',
loves: ['apple', 'watermelon'],
weight: 601,
gender: 'f',
vampires: 33
},
{
_id: ObjectId('658420139420db9a7cd124017'),
name: 'Pilot',
loves: ['apple', 'watermelon'],
weight: 650,
gender: 'm',
vampires: 54
},
{
_id: ObjectId('658420139420db9a7cd124018'),
name: 'Nimue',
loves: ['grape', 'carrot'],
weight: 540,
gender: 'f'
},
{
_id: ObjectId('6584203a9420db9a7cd124019'),
name: 'Roooooodles',
loves: ['apple'],
weight: 575,
gender: 'm',
vampires: 99
},
{
_id: ObjectId('658420a8942db9a7cd12401a'),
name: 'Unicrom',
loves: ['energon', 'redbull'],
weight: 984,
gender: 'm',
vampires: 182
},
{
_id: ObjectId('658421379420db9a7cd12401b'),
name: 'Dunx',
loves: ['grape', 'watermelon'],
weight: 704,
gender: 'm',
vampires: 165
}
]
learn>

2.2 ВЫБОРКА ДАННЫХ ИЗ БД

Практическое задание 2.2.1:

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

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

```
learn> db.unicorns.find({gender: 'm'}).sort({name:1})
[
  {
    _id: ObjectId('65842137942db9a7cd12401b'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('65842013942db9a7cd124010'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65842013942db9a7cd124014'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('65842013942db9a7cd124017'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('65842013942db9a7cd124015'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('6584203a942db9a7cd124019'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('658420a8942db9a7cd12401a'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
]
learn>
```

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

```
learn> db.unicorns.find({gender: 'f'}).sort({name:1}).limit(3)
[
  {
    _id: ObjectId('65842013942db9a7cd124011'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('65842013942db9a7cd124013'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('65842013942db9a7cd124016'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

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

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

db.unicorns.findOne({gender: 'f', loves:'apple'})

```
learn> db.unicorns.find({gender: 'f', loves:'apple'}).limit(1)
[
  {
    _id: ObjectId('658420139442db9a7cd124012'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
]
learn> db.unicorns.findOne({gender: 'f', loves:'apple'})
{
  _id: ObjectId('658420139442db9a7cd124012'),
  name: 'Solnara',
  loves: [ 'apple', 'carrot', 'chocolate' ],
  weight: 550,
  gender: 'f',
  vampires: 80
}
learn>
```

Практическое задание 2.2.2:

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

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

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

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

Практическое задание 2.1.4:

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

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

2.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

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})

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})

```
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>
```

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ vampires.

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

```
learn> db.unicorns.find({vampires:{$exists: false}}, {})
[
  {
    _id: ObjectId('658420139 42db9a7cd124018'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

Практическое задание 2.3.4:

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

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

3.1 ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 3.1.1:

Создайте коллекцию towns, включающую следующие документы:

```
learn> db.towns.insert({name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...     name: "Jim Wehrle"
... }}
... )
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65842c97942db9a7cd12401c') }
}
learn> db.towns.insert({name: "New York",
... populatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...     name: "Michael Bloomberg",
... party: "I"}}
... )
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65842cb7942db9a7cd12401d') }
}
learn> db.towns.insert({name: "Portland",
... populatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...     name: "Sam Adams",
... party: "D"}}
... )
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('65842cc7942db9a7cd12401e') }
}
learn>
```

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:1})

```
learn> db.towns.find({"mayor.party":"I"}, {name:1, mayor:1})
[
  {
    _id: ObjectId('65842cb7942db9a7cd12401d'),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn>
```

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, _id:0})

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

Практическое задание 3.1.2:

1) Создать курсор для списка самцов из первых двух особей с сортировкой в лексикографическом порядке.

   var cur = db.unicorns.find();null;

   cur.sort({name:1}).limit(2);null;

2) Вывести результат, используя forEach.

   cur.forEach(function(obj){print(obj.name);})

```
learn> cur.forEach(function(obj){print(obj.name);})
Aurora
Ayna

learn>
```

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()

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

Практическое задание 3.2.2:

Вывести список предпочтений.

db.unicorns.distinct("loves")

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

Практическое задание 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: 5 }, { _id: 'm', count: 7 } ]
learn>
```

Практическое задание 3.3.1:

Выполнить команду:

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

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

Практическое задание 3.3.3:

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

db.unicorns.update({name:'Raleigh', gender:'m'}, {$set:{loves:['redbool']}})

Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.update({name:'Raleigh', gender:'m'}, {$set:{loves:['redbool']}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Raleigh'})
[
  {
    _id: ObjectId('65842013942db9a7cd124015'),
    name: 'Raleigh',
    loves: [ 'redbool' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

Практическое задание 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: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Raleigh'})
[
  {
    _id: ObjectId('65842013942db9a7cd124015'),
    name: 'Raleigh',
    loves: [ 'redbool' ],
    weight: 421,
    gender: 'm',
    vampires: 7
  }
]
learn>
```

2. *Проверить содержимое коллекции* `unicorns`.

Практическое задание 3.3.5:

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

db.towns.updateOne({name: 'Portland'}, {$unset:{"mayor.party":1}})

```
learn> db.towns.updateOne({name: 'Portland'}, {$unset:{"mayor.party":1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find()
[
  {
    _id: ObjectId('65842c97942db9a7cd12401c'),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('65842cb7942db9a7cd12401d'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('65842cc7942db9a7cd12401e'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
learn>
```

2. *Проверить содержимое коллекции* `towns`.

Практическое задание 3.3.6:

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

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

Проверить содержимое коллекции unicorns.

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

learn> db.unicorns.find({name:'Pilot'})
[
  {
    _id: ObjectId('658420139 42db9a7cd124017'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn>
```

Практическое задание 3.3.7:

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

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

Проверить содержимое коллекции unicorns.

```
learn> db.unicorns.updateOne({name:'Aurora'}, {$addToSet:{loves:{$each:['lemon', 'sugar']}}}
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name:'Aurora'})
[
  {
    _id: ObjectId('658420139 42db9a7cd124011'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'lemon', 'sugar' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

3.4 УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

1) Удалите документы с беспартийными мэрами из towns:

   db.towns.deleteMany({'mayor.party':null})

```
learn> db.towns.find()
[
  {
    _id: ObjectId('658447dd942db9a7cd124021'),
    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('658447eb942db9a7cd124022'),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

2) Очистите коллекцию.

   db.towns.deleteMany({})

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

learn>
```

4 ССЫЛКИ И РАБОТА С ИНДЕКСАМИ В БАЗЕ ДАННЫХ MONGODB

Практическое задание 4.1.1:

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

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

Проверьте содержание коллекции единорогов.

db.unicorns.update({name:'Dunx'}, {$set:{town:{$ref: 'towns', $id: ObjectId('65845094942db9a7cd124037')}}})

db.unicorns.update({name:'Nimue'}, {$set:{town:{$ref: 'towns', $id: ObjectId('65845094942db9a7cd124037')}}})

db.unicorns.update({name:'Pilot'}, {$set:{town:{$ref: 'towns', $id: ObjectId('6584507e942db9a7cd124036')}}})

```
learn> db.unicorns.update({name:'Dunx'}, {$set:{town:{$ref: 'towns', $id:  ObjectId('65845094942db9a7cd124037')}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name:'Nimue'}, {$set:{town:{$ref: 'towns', $id:  ObjectId('65845094942db9a7cd124037')}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.update({name:'Pilot'}, {$set:{town:{$ref: 'towns', $id:   ObjectId('6584507e942db9a7cd124036')}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```


_id: ObjectId('658452bd942db9a7cd124041'),
name: 'Leia',
loves: ['apple', 'watermelon'],
weight: 601,
gender: 'f',
vampires: 33
},
{
_id: ObjectId('658452c1942db9a7cd124042'),
name: 'Pilot',
loves: ['apple', 'watermelon'],
weight: 650,
gender: 'm',
vampires: 54,
town: DBRef('towns', ObjectId('6584507e942db9a7cd124036'))
},
{
_id: ObjectId('658452c4942db9a7cd124043'),
name: 'Nimue',
loves: ['grape', 'carrot'],
weight: 540,
gender: 'f',
town: DBRef('towns', ObjectId('65845094942db9a7cd124037'))
},
{
_id: ObjectId('658452ce942db9a7cd124044'),
name: 'Dunx',
loves: ['grape', 'watermelon'],
weight: 704,
gender: 'm',
vampires: 165,
town: DBRef('towns', ObjectId('65845094942db9a7cd124037'))
}
]

4.2 НАСТРОЙКА ИНДЕКСОВ

Практическое задание 4.2.1:

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

db.unicorns.ensureIndex({'name':1}, {'unique':true})

```
learn> db.unicorns.ensureIndex({'name':1}, {'unique':true})
[ 'name_1' ]
learn> db.unicorns.find()
[
  {
    _id: ObjectId('6584528f942db9a7cd124039'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('65845297942db9a7cd12403a'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6584529c942db9a7cd12403b'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
```

4.3 УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 4.3.1:

1) Получите информацию о всех индексах коллекции unicorns

   db.unicorns.getIndexes()

```
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.

db.unicorns.dropIndex('name_1')

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

3) Попытайтесь удалить индекс для идентификатора.

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

4.4 ПЛАН ЗАПРОСА

Практическое задание 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.unicorns.explain('executionStats').find().skip(100000-4).limit(4)

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

db.unicorns.explain('executionStats').find().skip(100000-4).limit(4)

executionTimeMillisEstimate: 27

4) Создайте индекс для ключа value.

db.numbers.ensureIndex({'value':1}, {'unique':true})

5) Получите информацию о всех индексах коллекции numbers.

db.numbers.getIndexes()

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

6) Выполните запрос 2.

   db.unicorns.explain('executionStats').find().skip(100000-4).limit(4)

   executionTimeMillisEstimate: 26

7) Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?
8)
9) Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

   С индексом немного эффективнее.

Вывод:

Была изучена СУБД MongoDB выполнено множество запросов и обращений к созданной в ней базе данных. В сравнении с SQL намного более щадящая структуризация БД и запросов к ним, однако есть явная проблема с внешними ключами и их нормативанием.