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: Петухов С.А.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.

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.

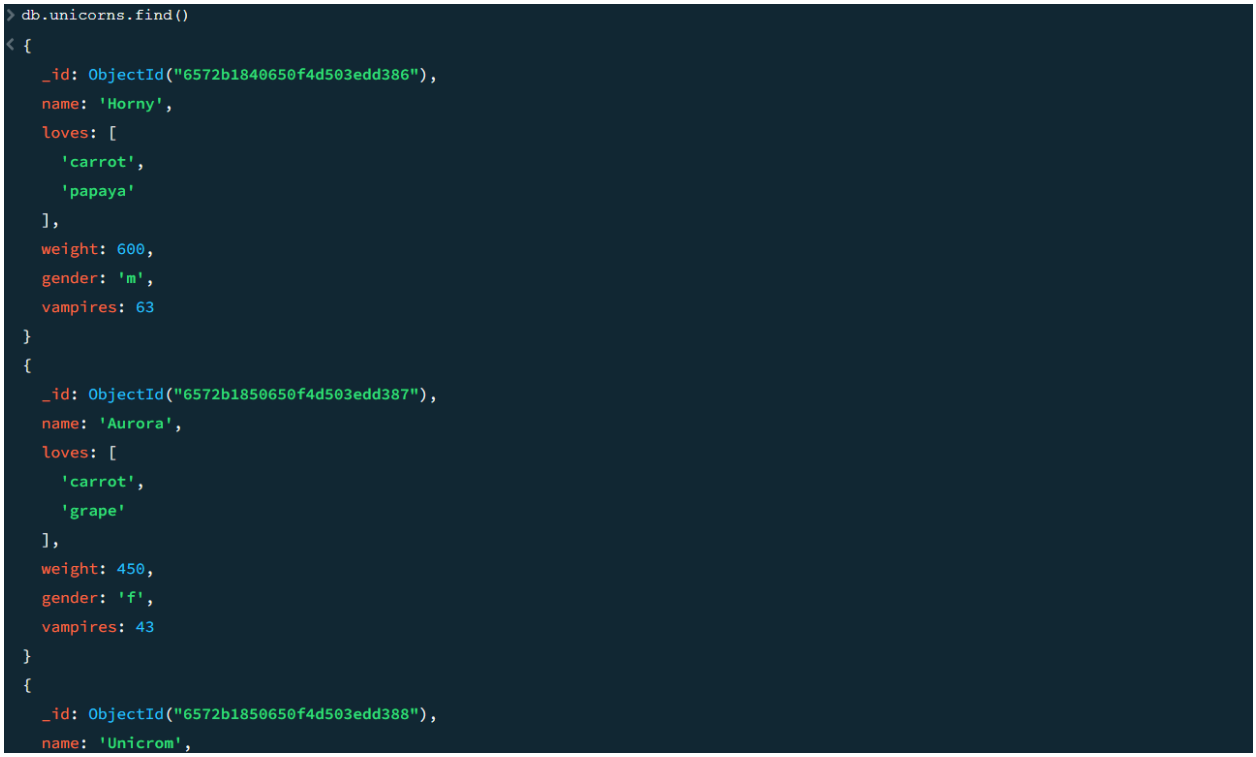
Выполнение:

Практическое задание 2.1.1:

1. *Создайте базу данных learn.*
2. *Заполните коллекцию единорогов unicorns:*
3. *Используя второй способ, вставьте в коллекцию единорогов документ:*

```
> 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("6572b1850650f4d503edd390")
    }
  }
```

```
> doc = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
< {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
> db.unicorns.insertOne(doc)
< {
    acknowledged: true,
    insertedId: ObjectId("6572b2370650f4d503edd391")
  }
```



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.*

Код

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

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



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

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

```
> db.unicorns.find({ gender: 'm' }).sort({ name: 1 });
< {
    _id: ObjectId("6572b2370650f4d503edd391"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("6572b1840650f4d503edd386"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
```

```
> db.unicorns.find({ gender: 'f' }).limit(3).sort({ name: 1 });
< {
    _id: ObjectId("6572b1850650f4d503edd387"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("6572b1850650f4d503edd38b"),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
    _id: ObjectId("6572b1850650f4d503edd38e"),
    name: 'Leia',
    loves: [
```

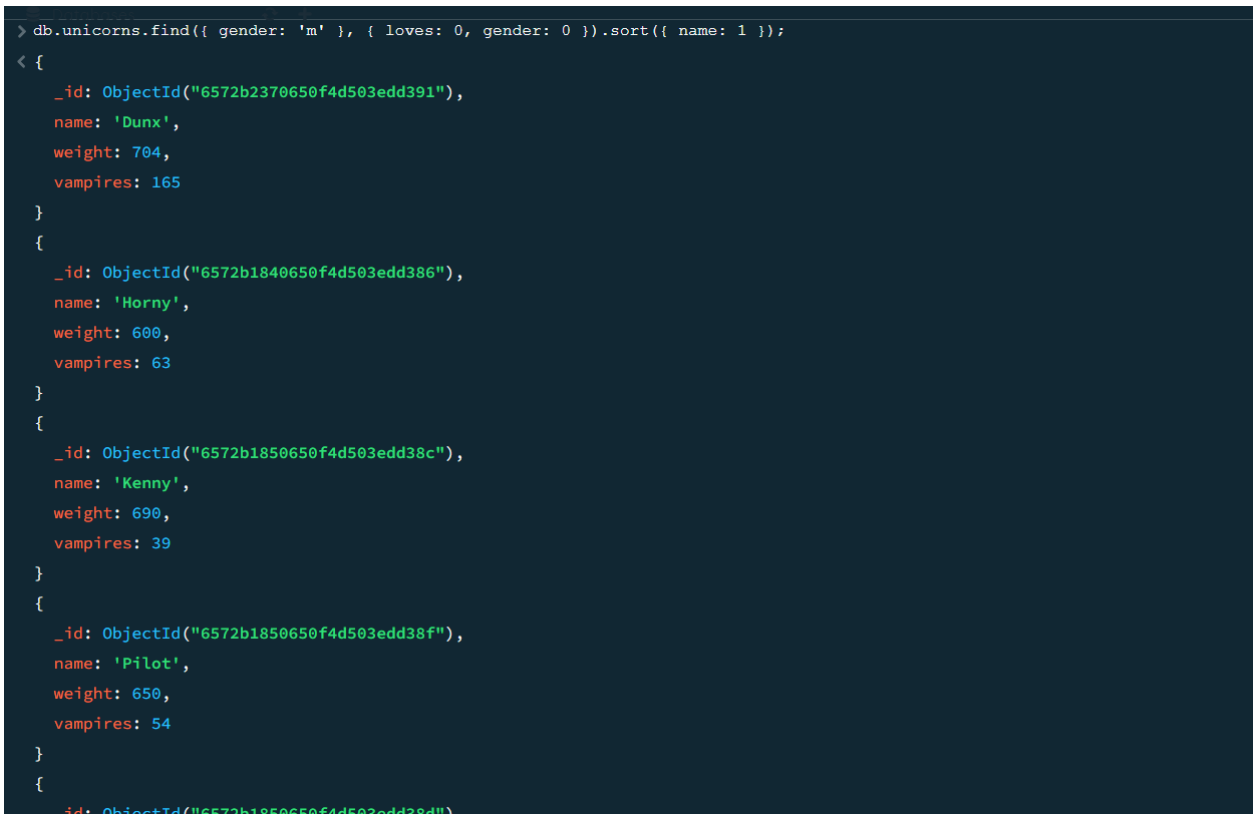
```
> db.unicorns.findOne({ loves: 'carrot', gender: 'f' });
< {
    _id: ObjectId("6572b1850650f4d503edd387"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
> db.unicorns.find({ loves: 'carrot', gender: 'f' }).limit(1);
< {
    _id: ObjectId("6572b1850650f4d503edd387"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Практическое задание 2.2.2:

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

Код

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



Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

Код

```
db.unicorns.find().sort({ _id: -1 });
```

```
> db.unicorns.find().sort({ _id: -1 });
< {
    _id: ObjectId("6572b2370650f4d503edd391"),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId("6572b1850650f4d503edd390"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
```

Практическое задание 2.1.4:

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

Код

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

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

Практическое задание 2.3.1:

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

Код

```
db.unicorns.find({ gender: 'f', weight: { $gte: 500, $lte: 700 } }, { _id: 0
}).sort({ name: 1 });
```

```
> db.unicorns.find({ gender: 'f', weight: { $gte: 500, $lte: 700 } }, { _id: 0 }).sort({ name: 1 });
< {
    name: 'Leia',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
  {
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    name: 'Solnara',
    loves: [
      'apple',
```

Практическое задание 2.3.2:

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

Код

```
db.unicorns.find({gender: 'm', weight: { $gte: 500 }, loves: { $all: ['grape',
'lemon'] }}, { _id: 0 });
```

```
> 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
  }
```

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.

Код

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

```
> db.unicorns.find({ vampires: { $exists: false } });
< {
    _id: ObjectId("6572b1850650f4d503edd390"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
```

Практическое задание 2.3.4:

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

Код

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

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

Практическое задание 3.1.1:

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

```
{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"}}
```

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.*

Код

```
db.towns.find({ "mayor.party": "I" }, { _id: 0, name: 1});

db.towns.find({ "mayor.party": { $exists: false } }, { _id: 0, name: 1,
"mayor.name": 1 });
```

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

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

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.*
4. *Содержание коллекции единорогов* `unicorns`*:*

Код

```
var MaleUnicorns = function() {
    var maleUnicorns = db.unicorns.find({ gender: 'm' });
    return maleUnicorns;
  };

var cursor = MaleUnicorns().sort({ name: 1 }).limit(2);


cursor.forEach(function(obj) {
    printjson(obj);
  });
```

```
> MaleUnicorns();
< Horny
< Unicrom
< Kenny
< Raleigh
< Pilot
< Dunx
```

```
> var cursor = MaleUnicorns().sort({ name: 1 }).limit(2);
> cursor.forEach(function(obj) {
      printjson(obj);
    });
< {
    _id: ObjectId("6573474fa649c816b644a532"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
< {
    _id: ObjectId("657346c5a649c816b644a528"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
```

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

Код

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

Практическое задание 3.2.2:

Вывести список предпочтений.

Код

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

Практическое задание 3.2.3:

Посчитать количество особей единорогов обоих полов.

Код

```
db.unicorns.aggregate([{$group: {_id: "$gender", count: { $sum: 1 }}}])
> db.unicorns.aggregate([{$group: {_id: "$gender", count: { $sum: 1 }}}])
< {
    _id: 'f',
    count: 5
  }
  {
    _id: 'm',
    count: 6
  }
Lab6>
```

Практическое задание 3.3.1:

1. *Выполнить команду:*

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

Проверить содержимое коллекции `unicorns`.

Код

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

```
> db.unicorns.save({name: 'Barny', loves: ['grape'],
  weight: 340, gender: 'm'})
✖ ▸ TypeError: db.unicorns.save is not a function
```

В интернете пишут, что данный метод устарел и больше не поддерживается

Практическое задание 3.3.2:

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

Код

```
db.unicorns.update({ name: 'Ayna', gender: 'f' },{$set: {weight: 800,vampires: 51}})
```

```
> db.unicorns.update({ name: 'Ayna', gender: 'f' },{$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
  }
Lab6>
```

```
{
  _id: ObjectId("6573474fa649c816b644a52c"),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 800,
  gender: 'f',
  vampires: 51
}
```

Практическое задание 3.3.3:

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

Код

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

```
> db.unicorns.update({ name: 'Raleigh', gender: 'm' },{$set: {loves: ['redbull']}});
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

```
{
  _id: ObjectId("6573474fa649c816b644a52e"),
  name: 'Raleigh',
  loves: [
    'redbull'
  ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
```

Практическое задание 3.3.4:

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

Код

```
db.unicorns.update({ gender: 'm' },{$inc: {vampires: 5}},{ multi: true })
> db.unicorns.update({ gender: 'm' },{$inc: {vampires: 5}},{ multi: true });
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 6,
    modifiedCount: 6,
    upsertedCount: 0
  }
```

```
> db.unicorns.find()
< {
    _id: ObjectId("657346c5a649c816b644a528"),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
  {
    _id: ObjectId("657346f6a649c816b644a529"),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId("657346f6a649c816b644a52a"),
    name: 'Unicrom',
```

Практическое задание 3.3.5:

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

Код

```
db.towns.update({ name: "Portland" },{$set: {"mayor.party": null}})
> db.towns.update({ name: "Portland" },{$set: {"mayor.party": null}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

```
  _id: ObjectId("65733ef2a649c816b644a527"),
  name: 'Portland',
  population: 528000,
  last_sensus: 2009-07-20T00:00:00.000Z,
  famous_for: [
    'beer',
    'food'
  ],
  mayor: {
    name: 'Sam Adams',
    party: null
  }
}
```

Практическое задание 3.3.6:

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

Код

```
db.unicorns.updateOne({ name: "Pilot", gender: "m" },{$push: {loves: "шоколад"}})
```

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

```
{
  _id: ObjectId("6573474fa649c816b644a530"),
  name: 'Pilot',
  loves: [
    'apple',
    'watermelon',
    'шоколад'
  ],
  weight: 650,
  gender: 'm',
  vampires: 59
}
```

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`.

Код

```
db.unicorns.update({ name: "Aurora", gender: "f" },{$push: {loves: { $each: ["sugar", "lemons"] }}})
```

```
> db.unicorns.update({ name: "Aurora", gender: "f" },{$push: {loves: { $each: ["sugar", "lemons"] }}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

```
{
  _id: ObjectId("657346f6a649c816b644a5
  name: 'Aurora',
  loves: [
    'carrot',
    'grape',
    'sugar',
    'lemons'
  ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

Практическое задание 3.4.1:

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

```
{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"}}
```

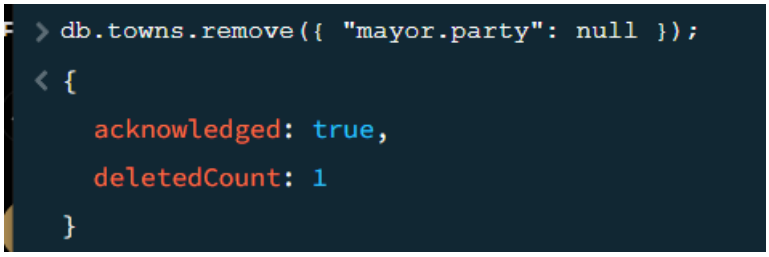
```
{name: "Portland",
popujatiuon: 528000,
last_sensus: ISODate("2009-07-20"),
famous_for: ["beer", "food"],
mayor: {
   name: "Sam Adams",
   party: "D"}}
```

2. *Удалите документы с беспартийными мэрами.*
3. *Проверьте содержание коллекции.*
4. *Очистите коллекцию.*
5. *Просмотрите список доступных коллекций.*

Код

```
db.towns.remove({ "mayor.party": null })

db.towns.remove({})
```



```
> db.towns.find()
< {
    _id: ObjectId("657420eea649c816b644a534"),
    name: 'New York',
    population: 22200000,
    last_sensus: 2009-07-31T00:00:00.000Z,
    famous_for: [
      'statue of liberty',
      'food'
    ],
    mayor: {
      name: 'Michael Bloomberg',
      party: 'I'
    }
  }
  {
    _id: ObjectId("657420eea649c816b644a535"),
    name: 'Portland',
    population: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [
      'beer',
      'food'
    ],
    mayor: {
      name: 'Sam Adams',
```

```
> db.getCollectionNames()
< [ 'towns', 'unicorns', 'Unicorn' ]
```

Практическое задание 4.1.1:

1. *Создайте коллекцию зон обитания единорогов, 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.*
4. *Содержание коллекции единорогов* `unicorns`*:*

Код

```
db.HabitatZone.insert({_id : "us", name: "United States"})
db.HabitatZone.insert({_id : "ru", name: "Russia"})
db.HabitatZone.insert({_id : "eu", name: "Europa"})

db.unicorns.update({name: "Horny"}, {$set: {habitatZone: {$ref: "HabitatZone",
$id: "us"}}})
```

```
db.unicorns.update({name: "Aurora"}, {$set: {habitatZone: {$ref: "HabitatZone", $id: "ru"}}})

db.unicorns.update({name: "Unicrom"}, {$set: {habitatZone: {$ref: "HabitatZone", $id: "eu"}}})
```

```
> db.HabitatZone.insert({_id : "us", name: "United States"})
  db.HabitatZone.insert({_id : "ru", name: "Russia"})
  db.HabitatZone.insert({_id : "eu", name: "Europa"})
< {
    acknowledged: true,
    insertedIds: {
      '0': 'eu'
    }
  }
> db.HabitatZone.find()
< {
    _id: 'us',
    name: 'United States'
  }
  {
    _id: 'ru',
    name: 'Russia'
  }
  {
    _id: 'eu',
    name: 'Europa'
  }
```

```
> db.unicorns.update({name: "Horny"}, {$set: {habitatZone: {$ref: "HabitatZone", $id: "us"}}})

  db.unicorns.update({name: "Aurora"}, {$set: {habitatZone: {$ref: "HabitatZone", $id: "ru"}}})

  db.unicorns.update({name: "Unicrom"}, {$set: {habitatZone: {$ref: "HabitatZone", $id: "eu"}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Практическое задание 4.2.1:

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

```
db.unicorns.insert({name:
'Horny',        dob:        new
Date(1992,2,13,7,47),   loves:
['carrot','papaya'],   weight:
600,  gender:  'm',  vampires:
63});

db.unicorns.insert({name:
'Aurora', dob: new Date(1991,
0,   24,   13,   0),   loves:
['carrot',  'grape'],  weight:
```

```
450, gender: 'f', vampires:
43});

db.unicorns.insert({name:
'Unicrom', dob: new Date(1973,
1, 9, 22, 10), loves:
['energon', 'redbull'],
weight: 984, gender: 'm',
vampires: 182});

db.unicorns.insert({name:
'Roooooodles', dob: new
Date(1979, 7, 18, 18, 44),
loves: ['apple'], weight: 575,
gender: 'm', vampires: 99});

db.unicorns.insert({name:
'Solnara', dob: new Date(1985,
6, 4, 2, 1), loves:['apple',
'carrot', 'chocolate'],
weight:550, gender:'f',
vampires:80});

db.unicorns.insert({name:'Ayn
a', dob: new Date(1998, 2, 7,
8, 30), loves: ['strawberry',
'lemon'], weight: 733, gender:
'f', vampires: 40});

db.unicorns.insert({name:'Ken
ny', dob: new Date(1997, 6, 1,
10, 42), loves: ['grape',
'lemon'], weight:
690, gender: 'm', vampires:
39});

db.unicorns.insert({name:
'Raleigh', dob: new Date(2005,
4, 3, 0, 57), loves: ['apple',
'sugar'], weight: 421, gender:
'm', vampires: 2});

db.unicorns.insert({name:
'Leia', dob: new Date(2001, 9,
8, 14, 53), loves: ['apple',
'watermelon'], weight: 601,
gender: 'f', vampires: 33});

db.unicorns.insert({name:
'Pilot', dob: new Date(1997,
2, 1, 5, 3), loves: ['apple',
'watermelon'], weight: 650,
gender: 'm', vampires: 54});

db.unicorns.insert ({name:
'Nimue', dob: new Date(1999,
11, 20, 16, 15), loves:
['grape', 'carrot'], weight:
540, gender: 'f'});

db.unicorns.insert {name:
'Dunx', dob: new Date(1976, 6,
18, 18, 18), loves: ['grape',
```

```
'watermelon'],  weight:  704,
gender: 'm', vampires: 165
```

Код

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

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

Практическое задание 4.3.1:

1. *Получите информацию о всех индексах коллекции* `unicorns`.
2. *Удалите все индексы, кроме индекса для идентификатора.*
3. *Попытайтесь удалить индекс для идентификатора.*

Код

```
db.unicorns.getIndexes()

db.unicorns.dropIndex("name_1")

db.unicorns.dropIndex("_id_")
```

```
> 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
```

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.*
3. *Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра* `executionTimeMillis`*)*
4. *Создайте индекс для ключа* `value`*.*
5. *Получите информацию о всех индексах коллекции* `numbres`*.*
6. *Выполните запрос 2.*
7. *Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?*

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

Код

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

db.numbers.find().sort({ _id: -1 }).limit(4)

db.numbers.find().sort({ _id: -1 }).limit(4).explain("executionStats")

db.numbers.createIndex({ value: 1 })
db.numbers.getIndexes();
```

```
> db.numbers.find().sort({ _id: -1 }).limit(4);
< {
    _id: ObjectId("65742f1b0185f77ec61047d7"),
    value: 99999
  }
```

```
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 30,
    totalKeysExamined: 4,
    totalDocsExamined: 4,
    executionStages: {
```

```
> db.numbers.createIndex({ value: 1 })
  db.numbers.getIndexes();
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { value: 1 }, name: 'value_1' }
  ]
Lab6>
```

```
   },
   executionStats: {
     executionSuccess: true,
     nReturned: 4,
     executionTimeMillis: 1,
     totalKeysExamined: 4,
     totalDocsExamined: 4,
     executionStages: {
       stage: 'limit'
```

Поиск данных с индексом работает гораздо быстрее и эффективнее.