Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №6 «Реализация БД с использованием СУБД MongoDB. Запросы к базе данных»

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Бархатова Н.А.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

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.

ВСТАВКА ДОКУМЕНТОВ В КОЛЛЕКЦИЮ

Практическое задание 2.1.1

1. *Создайте базу данных learn.*
2. *Заполните коллекцию единорогов unicorns:*

```
learn> db.createCollection("unicorns")
{ ok: 1 }
Please enter a MongoDB connection string (Default: mongodb://localhost/): learn> db.unicorns.insert({name: 'Horny', loves: ['car
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642862255ab0de2acaf66') }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6576428c2255ab0de2acaf67') }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642922255ab0de2acaf68') }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642972255ab0de2acaf69') }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6576429b2255ab0de2acaf6a') }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('6576429f2255ab0de2acaf6b') }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642a32255ab0de2acaf6c') }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642a72255ab0de2acaf6d') }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642ac2255ab0de2acaf6e') }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642b02255ab0de2acaf6f') }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642b52255ab0de2acaf70') }
}
```

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.unicors.insert(document)
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657642f42255ab0de2acaf71') }
}
```

4. *Проверьте содержимое коллекции с помощью метода find.*

ВЫБОРКА ДАННЫХ ИЗ БД

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.*

```
learn> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId('6576428c2255ab0de2acaf67'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('6576429f2255ab0de2acaf6b'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('657642ac2255ab0de2acaf6e'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1})
[
  {
    _id: ObjectId('657643172255ab0de2acaf72'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657642862255ab0de2acaf66'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('657642a32255ab0de2acaf6c'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('657642b02255ab0de2acaf6f'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657642a72255ab0de2acaf6d'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('657642972255ab0de2acaf69'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
```

2. *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('6576428c2255ab0de2acaf67'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

```
learn> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId('6576428c2255ab0de2acaf67'),
    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 предпочтениях и поле.

```
learn> db.unicorns.find({gender: 'm'}, {gender: 0, loves: 0}).sort({name: 1})
[
  {
    _id: ObjectId('657643172255ab0de2acaf72'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('657642862255ab0de2acaf66'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('657642a32255ab0de2acaf6c'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('657642b02255ab0de2acaf6f'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId('657642a72255ab0de2acaf6d'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
```

Практическое задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find().sort({$natural: -1})
[
  {
    _id: ObjectId('657643172255ab0de2acaf72'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('657642b52255ab0de2acaf70'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('657642b02255ab0de2acaf6f'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('657642ac2255ab0de2acaf6e'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    _id: ObjectId('657642a72255ab0de2acaf6d'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('657642a32255ab0de2acaf6c'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('6576429f2255ab0de2acaf6b'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('6576429b2255ab0de2acaf6a'),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    _id: ObjectId('657642972255ab0de2acaf69'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
```

Практическое задание 2.2.4

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

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

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Практическое задание 2.3.1

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

```
learn> db.unicorns.find({weight: {$gt:500, $lt: 700}}, {_id: 0})
[
  {
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    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'
  }
]
```

Практическое задание 2.3.2

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

```
learn> db.unicorns.find({weight: {$gt:500}, loves: {$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`.

```
learn> db.unicorns.find({vampires: {$exists: false}})
[
  {
    _id: ObjectId('657642b52255ab0de2acaf70'),
    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.

```
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' ] }
]
```

ЗАПРОС К ВЛОЖЕННЫМ ОБЪЕКТАМ

Практическое задание 3.1.1

1. *Создайте коллекцию towns, включающую следующие документы*

```
learn> db.towns.find()
[
  {
    _id: ObjectId('65764e212255ab0de2acaf73'),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('65764e3d2255ab0de2acaf74'),
    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('65764e4d2255ab0de2acaf75'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

2. 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"}, {mayor:1, name:1, _id:0})
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn>
```

3. 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}}, {mayor:1, name:1, _id:0})
[ { name: 'Punxsutawney ', mayor: { name: 'Jim Wehrle' } } ]
learn>
```

КУРСОРЫ

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.*

```
learn> function printUnicornsMale(){
... var cursor = db.unicorns.find({gender: 'm'}); null;
... cursor.sort({name:1}).limit(2); null;
... cursor.forEach(function(unicorn) {
... print(unicorn.name);
... });
... }
[Function: printUnicornsMale]
```

```
[Function: printUnicornsMale]
learn> printUnicornsMale()
Dunx
Horny
```

АГРЕГИРОВАННЫЕ ЗАПРОСЫ

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

Практическое задание 3.2.2

Вывести список предпочтений.

```
learn> 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.

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

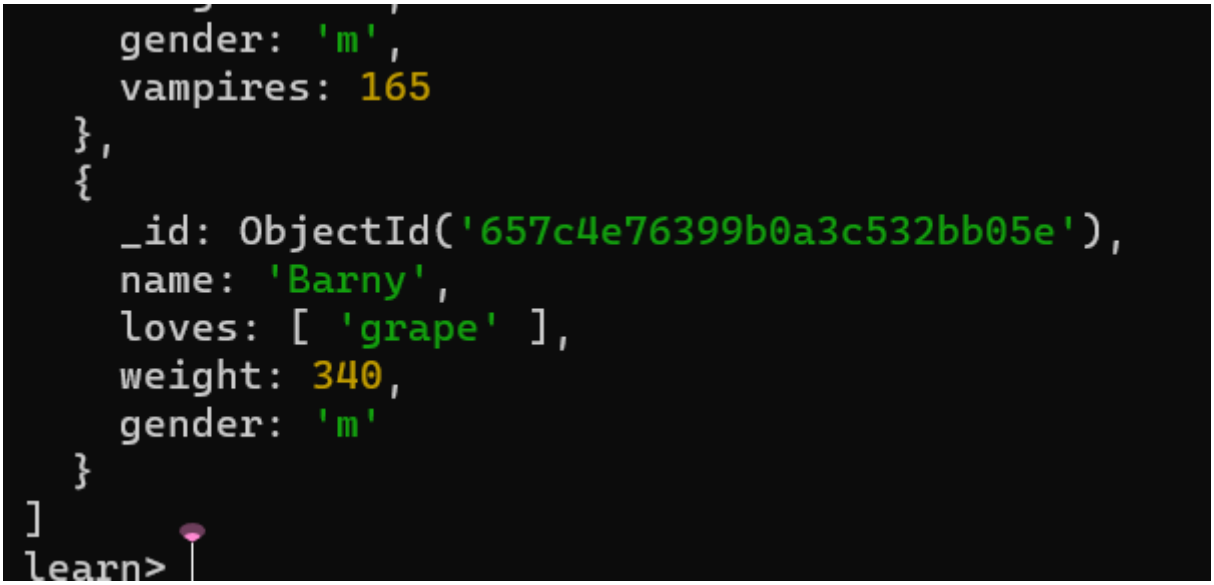
РЕДАКТИРОВАНИЕ ДАННЫХ

Практическое задание 3.3.1

1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции* `unicorns`.



Практическое задание 3.3.2

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

```
learn> db.unicorns.replaceOne({name : "Ayna"}, {name: "Ayna", loves:
... ["strawberry", "lemon"], weight: 800, gender: "f", vampires: 51})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

3. *Проверить содержимое коллекции unicorns*

Практическое задание 3.3.3

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

```
learn> db.unicorns.replaceOne({name : "Raleigh", gender: "m"}, {name:
... "Raleigh", loves: ['apple', 'sugar', 'redbull']}, {upsert: true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Практическое задание 3.3.4

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

```
learn> db.unicorns.update({gender : "m"}, {$inc: {vampire:5}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Практическое задание 3.3.5

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

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

Практическое задание 3.3.6

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

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

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`.

```
learn> db.unicorns.update({name : "Aurora", "gender": "m"}, {$addToSet: {loves: {$each: ["lemon", "sugar"]}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
learn>
```

УДАЛЕНИЕ ДАННЫХ ИЗ КОЛЛЕКЦИИ

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"}}
```

```
learn> db.towns.insert({name: "Punxsutawney ",
... popujatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: ["phil the groundhog"],
... mayor: {
...     name: "Jim Wehrle"
... }}
... )
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWr
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657cd27c8743a0e00b349fc7') }
}
learn> db.towns.insert({name: "New York",
... popujatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...     name: "Michael Bloomberg",
... party: "I"}}
... )
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657cd2908743a0e00b349fc8') }
}
learn> db.towns.insert({name: "Portland",
... popujatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...     name: "Sam Adams",
... party: "D"}}
... )
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657cd2a58743a0e00b349fc9') }
}
```

2. *Удалите документы с беспартийными мэрами.*

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

3. *Проверьте содержание коллекции.*

```
learn> db.towns.find()
[
  {
    _id: ObjectId('657cd2908743a0e00b349fc8'),
    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('657cd2a58743a0e00b349fc9'),
    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.*
5. *Просмотрите список доступных коллекций.*

```
learn> db.towns.remove({})
{ acknowledged: true, deletedCount: 2 }
learn> show collections
towns
unicorns
learn> db.towns.find()

learn>
```

ССЫЛКИ В БД

Практическое задание 4.1.1

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

```
learn> db.habitat.insertMany([
...   {
...     _id: "zone1",
...     name: "Зона 1",
...     description: "Описание зоны 1"
...   },
...   {
...     _id: "zone2",
...     name: "Зона 2",
...     description: "Описание зоны 2"
...   }
... ]);
```

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

```
learn> var zone1Id = db.habitat.findOne({_id: "zone1"})._id

learn> var zone2Id = db.habitat.findOne({_id: "zone2"})._id
```

```
learn> db.unicorns.updateMany({name: "Horny"}, {$set: {habitat:{$ref:"habitat",
$id: zone1Id}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateMany({name: "Nimue"}, {$set: {habitat:{$ref:"habitat",
$id: zone2Id}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

3. *Проверьте содержание коллекции единорогов.*


learn> db.unicorns.find()
[
{
_id: ObjectId('657cd5f88743a0e00b349fd1'),
name: 'Horny',
loves: ['carrot', 'papaya'],
weight: 600,
gender: 'm',
vampires: 63,
habitat: DBRef('habitat', 'zone1')
},
{
_id: ObjectId('657cd5fe8743a0e00b349fd2'),
name: 'Aurora',
loves: ['carrot', 'grape'],
weight: 450,
gender: 'f',
vampires: 43
},
{
_id: ObjectId('657cd6048743a0e00b349fd3'),
name: 'Unicrom',
loves: ['energon', 'redbull'],
weight: 984,
gender: 'm',

НАСТРОЙКА ИНДЕКСОВ

Практическое задание 4.2.1

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

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

УПРАВЛЕНИЕ ИНДЕКСАМИ

Практическое задание 4.3.1

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 }
]
learn>
```

2. *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_' } ]
learn>
```

3. *Попытайтесь удалить индекс для идентификатора.*

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

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})}
```

```
learn> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('657cdcc68743a0e00b3651b5') }
}
```

2. *Выберите последних четыре документа.*

```
learn> db.numbers.find().sort({value: -
... 1}).limit(4).explain("executionStats")
{
```

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

```
  nReturned: 4,
  executionTimeMillis: 142,
  totalKeysExamined: 0,
```

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

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

5. *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' }
]
```

6. *Выполните запрос 2.*

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

```
  nReturned: 4,
  executionTimeMillis: 20,
  totalKeysExamined: 4,
```

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

Более эффективным оказался запрос, сделанный после добавления индекса. Он выполняется в 7 раз быстрее.

Вывод

В ходе выполнения данной лабораторной работы были освоены практические навыки работы с CRUD-операциями в СУБД MongoDB. Были изучены основные команды для создания, чтения, обновления и удаления данных в коллекциях MongoDB. Также было изучено использование вложенных объектов в коллекциях, агрегации данных и изменение данных с помощью ссылок и индексов. В целом, данная лабораторная работа позволила овладеть основными навыками работы с БД в СУБД MongoDB и приобрести практический опыт в выполнении различных операций с данными.