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: Русинов В.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).

Практическое задание 2.1.1:

1. *Создайте базу данных learn.*

```
database> use learn
switched to db learn
```

2. *Заполните коллекцию единорогов unicorns:*

```
      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'});
```

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('658b4914fecf0095ff3f167e') }
}
```

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

```
learn> db.unicorns.find()
[
  {
    _id: ObjectId('658b4837fecf0095ff3f1673'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('658b4837fecf0095ff3f1674'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f1675'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
```

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.*

   Самцы:

   db.unicorns.find({gender: "m"}).limit(3).sort({name: 1})

```
learn> db.unicorns.find({gender: "m"}).limit(3).sort({name: 1})
[
  {
    _id: ObjectId('658b4914fecf0095ff3f167e'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('658b4837fecf0095ff3f1673'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f1679'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

   Самки:

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

```
learn> db.unicorns.find({gender: "f"}).limit(3).sort({name: 1})
[
  {
    _id: ObjectId('658b4837fecf0095ff3f1674'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f1678'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f167b'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
```

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

db.unicorns.findOne({loves: "carrot"})

```
learn> db.unicorns.findOne({loves: "carrot"})
{
  _id: ObjectId('658b4837fecf0095ff3f1673'),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

db.unicorns.find({loves: "carrot"}).limit(1)

```
learn> db.unicorns.find({loves: "carrot"}).limit(1)
[
  {
    _id: ObjectId('658b4837fecf0095ff3f1673'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
```

Практическое задание 2.2.2:

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

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

```
learn> db.unicorns.find({gender : "m"}, {loves : 0, gender : 0}).sort({name: 1})
[
  {
    _id: ObjectId('658b4914fecf0095ff3f167e'),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  },
  {
    _id: ObjectId('658b4837fecf0095ff3f1673'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f1679'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
```

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

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

```
learn> db.unicorns.find().sort({ $natural: -1 })
[
  {
    _id: ObjectId('658b4914fecf0095ff3f167e'),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f167d'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f167c'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
```

Практическое задание 2.2.4:

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

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

```
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
  },
```

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})

```
learn> db.unicorns.find({gender : "f", weight : {$gte : 500, $lte : 700}}, {_id: 0})
[
  {
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

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
  }
]
```

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('658b4838fecf0095ff3f167d'),
    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.

db.unicorns.find({gender : "m"}, {loves: {$slice : 1}})

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"}}
```

```
db.towns.insertMany([{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"}}
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('658b4dd4fecf0095ff3f167f'),
    '1': ObjectId('658b4dd4fecf0095ff3f1680'),
    '2': ObjectId('658b4dd4fecf0095ff3f1681')
  }
}
```

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 : 1})

```
learn> db.towns.find({"mayor.party" : "I"}, {name : 1, mayor : 1})
[
  {
    _id: ObjectId('658b4dd4fecf0095ff3f1680'),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
```

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

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

```
learn> db.towns.find({"mayor.party" : {$exists : 1}}, {name : 1, mayor : 1, _id : false})
[
  {
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  { name: 'Portland', mayor: { name: 'Sam Adams', party: 'D' } }
]
```

Практическое задание 3.1.2:

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

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

```
learn> function getMaleUnicorns() {return db.unicorns.find({gender:'m'})}
[Function: getMaleUnicorns]
learn> getMaleUnicorns()
[
  {
    _id: ObjectId('658b4837fecf0095ff3f1673'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('658b4838fecf0095ff3f1675'),
```

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

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

```
learn> var cursor = getMaleUnicorns().sort({name:1}).limit(2);
```

3. *Вывести результат, используя forEach.*

   cursor.forEach(function(obj){print(obj.name)})

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

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

db.unicorns.find({gender : "f", weight : {$gte : 500, $lte : 600}}).count()

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'
]
```

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: 'm', count: 7 }, { _id: 'f', count: 5 } ]
```

Практическое задание 3.3.1:

1. *Выполнить команду:*

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

save не работает, так что:

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

2. *Проверить содержимое коллекции* `unicorns`.

Практическое задание 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`.

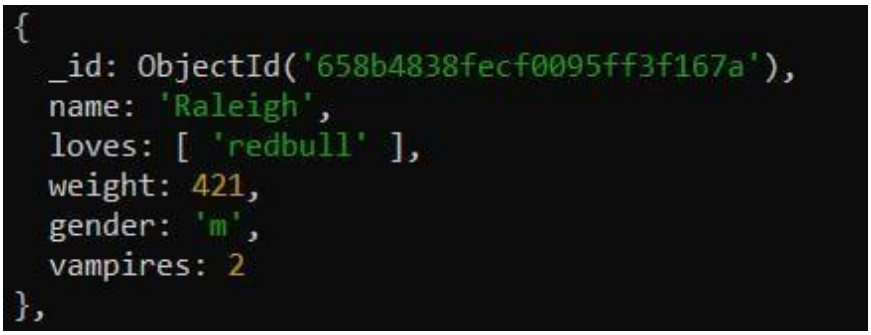
```
{
  _id: ObjectId('658b4838fecf0095ff3f1678'),
  name: 'Ayna',
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51
},
```

Практическое задание 3.3.3:

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

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

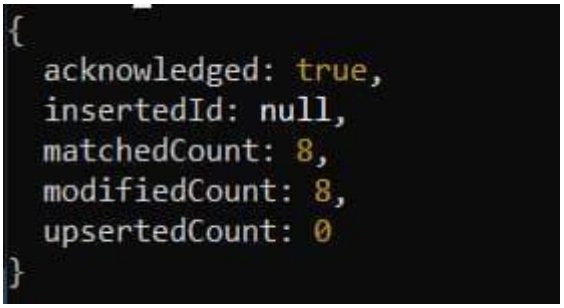
2. *Проверить содержимое коллекции* `unicorns`.



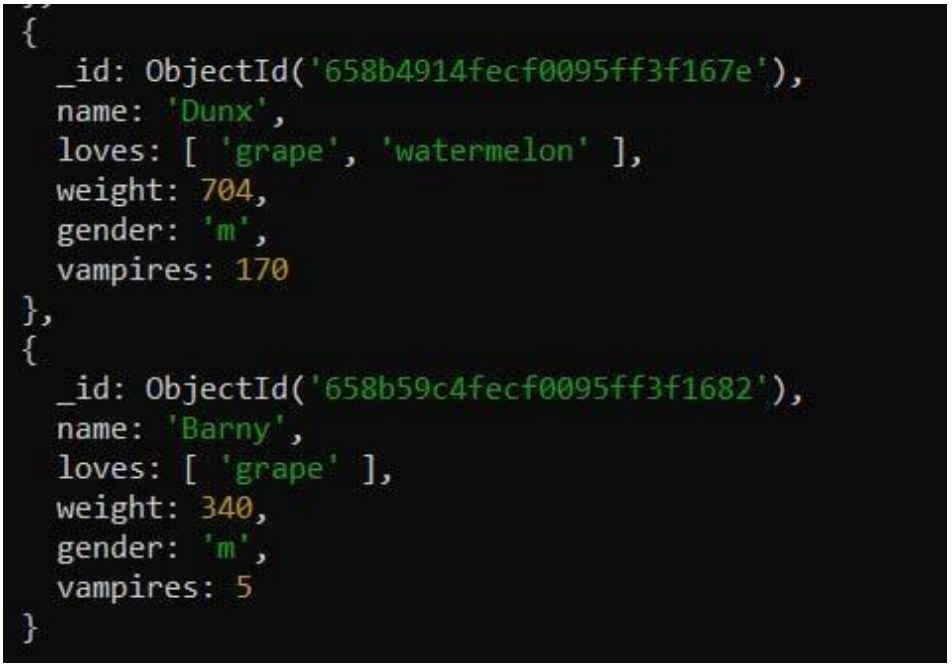
Практическое задание 3.3.4:

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

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



2. *Проверить содержимое коллекции* `unicorns`.



Практическое задание 3.3.5:

1. *Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.*
   db.towns.update({name : "Portland"}, {$unset: {"mayor.party" : 1}})
2. *Проверить содержимое коллекции* `towns`.

Практическое задание 3.3.6:

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

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

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

2. *Проверить содержимое коллекции* `unicorns`.

```
{
  _id: ObjectId('658b4838fecf0095ff3f167c'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  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.*

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

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

2. *Проверить содержимое коллекции* `unicorns`.

```
{
  _id: ObjectId('658b4837fecf0095ff3f1674'),
  name: 'Aurora',
  loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
  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.*

```
db.towns.deleteMany({"mayor.party" : {$exists : false}})
```

3. *Проверьте содержание коллекции.*

```
learn> db.towns.find()
[
  {
    _id: ObjectId('658b4dd4fecf0095ff3f1680'),
    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' }
  }
]
```

Практическое задание 4.1.1:

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

   db.zones.insertMany([{_id : "forest", name: "The forest", description: "trees and trees"}, {_id : "mountain", name: "The mountain", description: "peaks and peaks"}, {_id: "sea", name:"The sea",description: "waves and waves"}])

```
learn> db.zones.insertMany([{_id : "forest", name: "The forest", description: "tr
e: "The mountain", description: "peaks and peaks"}, {_id: "sea", name:"The sea",
... description: "waves and waves"}])
{
  acknowledged: true,
  insertedIds: { '0': 'forest', '1': 'mountain', '2': 'sea' }
}
```

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

   db.unicorns.updateOne({_id: ObjectId("658b59c4fecf0095ff3f1682")}, {$set: {location:{$ref : "zones", $id:"forest"}}})

```
{
  _id: ObjectId('658b59c4fecf0095ff3f1682'),
  name: 'Barny',
  loves: [ 'grape' ],
  weight: 340,
  gender: 'm',
  vampires: 5,
  location: DBRef('zones', 'forest')
}
```

   db.unicorns.updateOne({_id: ObjectId("658b4838fecf0095ff3f167c")}, {$set: {location:{$ref : "zones", $id:"sea"}}})

```
{
  _id: ObjectId('658b4838fecf0095ff3f167c'),
  name: 'Pilot',
  loves: [ 'apple', 'watermelon', 'chocolate' ],
  weight: 650,
  gender: 'm',
  vampires: 59,
  location: DBRef('zones', 'sea')
},
```

Практическое задание 4.2.1:

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

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

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.*

```
learn> db.unicorns.dropIndexes()
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
learn> 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})}
```

```
learn> for(i = 0; i < 1000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('658b640bfecf0095ff405e23') }
}
```

2. *Выберите последних четыре документа.*

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

```
learn>
[
  { _id: ObjectId('658b659dfecf0095ff41e4c3'), value: 99999 },
  { _id: ObjectId('658b659dfecf0095ff41e4c2'), value: 99998 },
  { _id: ObjectId('658b659dfecf0095ff41e4c1'), value: 99997 },
  { _id: ObjectId('658b659dfecf0095ff41e4c0'), value: 99996 }
]
```

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

```
executionStages: {
  stage: 'sort',
  planNodeId: 2,
  nReturned: 4,
  executionTimeMillisEstimate: 113,
  opens: 1,
```

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

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

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?*

```
executionStages: {
  stage: 'limit',
  planNodeId: 3,
  nReturned: 4,
  executionTimeMillisEstimate: 18,
  opens: 1,
```

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

Ускорение при запросе с индексами сильно заметно, следовательно запрос на выборку конкретных значений с индексами намного эффективнее такого же запроса, но без индексов.

Вывод

В ходе лабораторной работы были использованы различные методы и освоены на практике NoSQL БД MongoDB.