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: Пиотуховский А.А.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

Цель работы

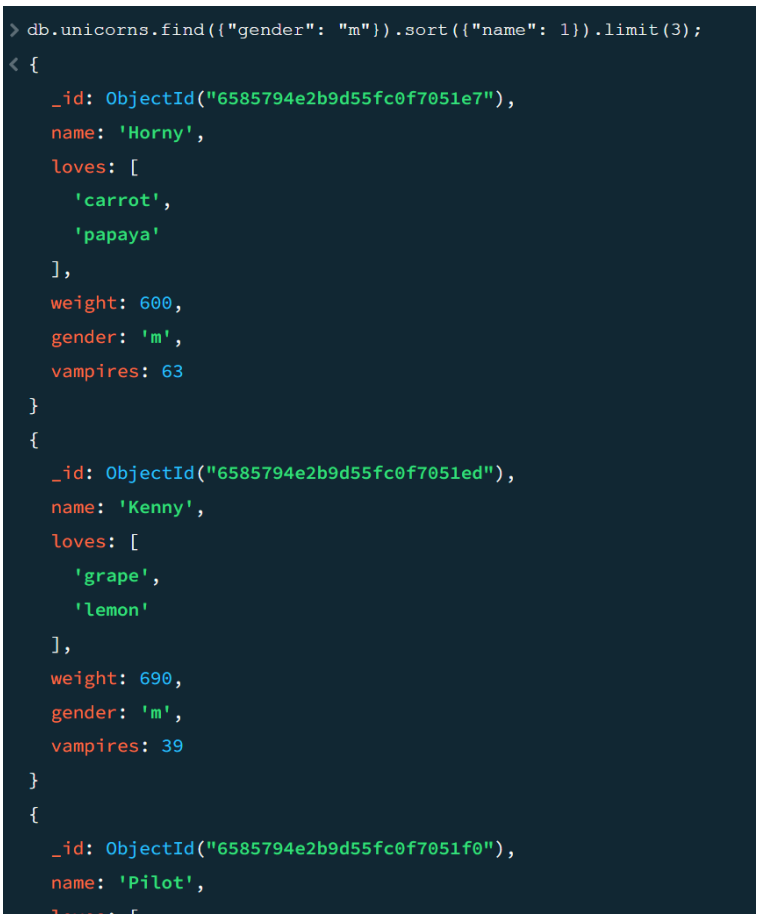
Овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

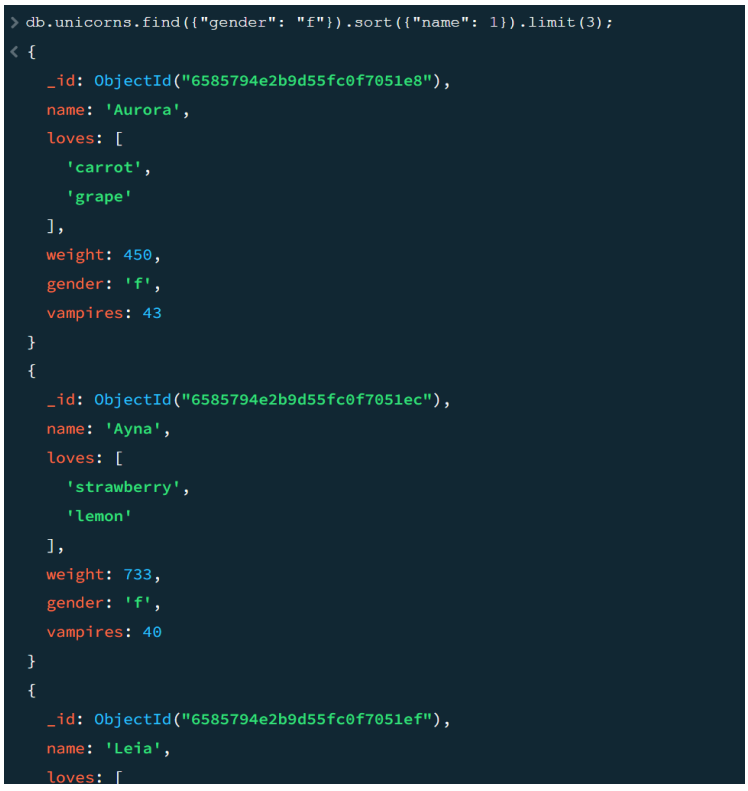
Практическое задание 2.1.1

1. Создайте базу данных learn.
2. Заполните коллекцию единорогов unicorns:
3. Используя второй способ, вставьте в коллекцию единорогов документ:
4. Проверьте содержимое коллекции с помощью метода find.

```
    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'});
```

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



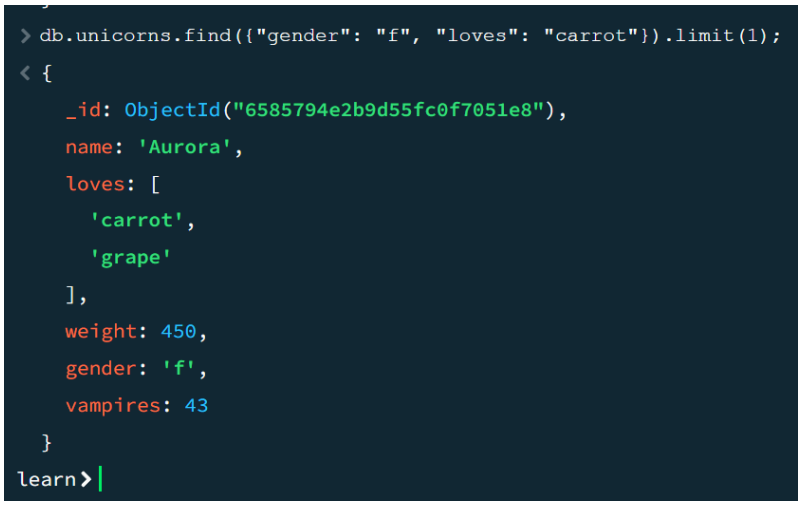


Практическое задание 2.2.1

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.*

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



Практическое задание 2.2.2

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

```
db.unicorns.findOne({"gender": "m", "loves": "carrot"}, {loves:
0, gender: 0})
```

```
> db.unicorns.findOne({"gender": "m", "loves": "carrot"}, {loves: 0, gender: 0})
< {
    _id: ObjectId("6585794e2b9d55fc0f7051e7"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
learn>
```

Практическое задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

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

```
> db.unicorns.find().sort({ $natural: -1 });
< {
    _id: ObjectId("6585794e2b9d55fc0f7051f1"),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId("6585794e2b9d55fc0f7051f0"),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
  {
    _id: ObjectId("6585794e2b9d55fc0f7051ef"),
    name: 'Leia',
    loves: [
      'apple',
      'watermelon'
```

Практическое задание 2.1.4

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

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

Практическое задание 2.3.1

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

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

```
> 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
  }
```

Практическое задание 2.3.2

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

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

```
> db.unicorns.find({weight: { $gt: 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 } }, { _id: 0 })
```

```
> db.unicorns.find({ vampires: { $exists:false } }, { _id: 0 })
< {
    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" }, {name: 1, loves: { $slice: 1 }, _id: 0 }).sort({ name: 1 })
```

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

Практическое задание 3.1.1

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

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.*

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

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

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

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.*

```
const func = (() => db.unicorns.find({"gender" : "m"}).sort({"name":
1}).limit(2))().forEach((document) => print(document))
```

```
> const func = (() => db.unicorns.find({"gender" : "m"}).sort({"name": 1}).limit(2))().forEach((document) => print(document))
< {
    _id: ObjectId("6585794e2b9d55fc0f7051e7"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
< {
    _id: ObjectId("6585794e2b9d55fc0f7051ed"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
learn>
```

Практическое задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
db.unicorns.find({gender: "f", weight: { $gt: 500, $lt: 600 }},
{ _id: 0 }).count()
```

```
> db.unicorns.find({gender: "f", weight: { $gt: 500, $lt: 600 }}, { _id: 0 }).count()
< 2
learn>
```

Практическое задание 3.2.2

Вывести список предпочтений.

```
db.unicorns.distinct("loves")
```

```
> 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}}})
```

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

Практическое задание 3.3.1

1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции unicorns.*

```
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: 7
  }
learn>
```

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.*

```
db.unicorns.updateOne({name: "Ayna"}, {$set: {weight: 800, vampires: 51}});
```

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

Практическое задание 3.3.3

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

```
db.unicorns.updateOne({name: "Raleigh"}, {
    $push: {"loves": "redbull"}
})
```

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

Практическое задание 3.3.4

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

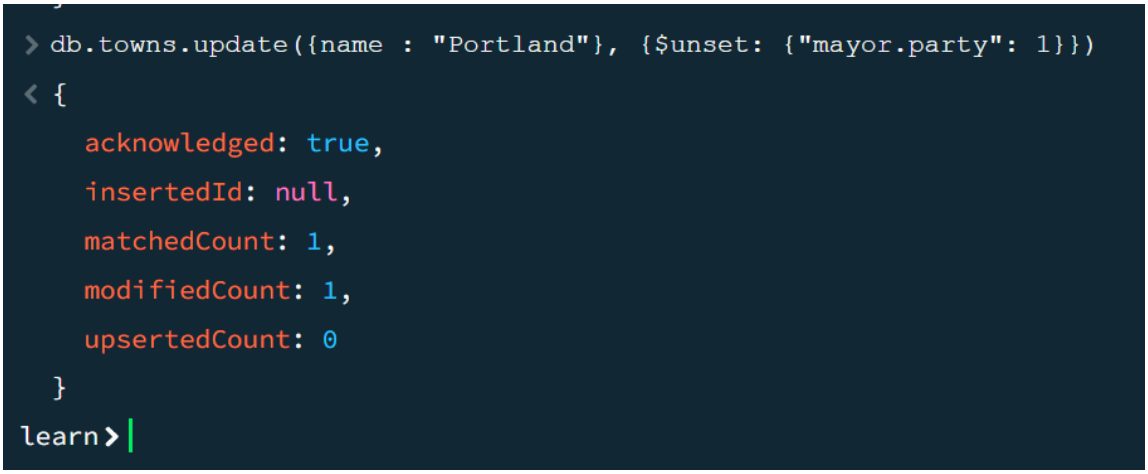
```
db.unicorns.update({gender : "m"}, {$inc: {vampire:5}})
```

```
> db.unicorns.update({gender : "m"}, {$inc: {vampire:5}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
learn>
```

Практическое задание 3.3.5

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

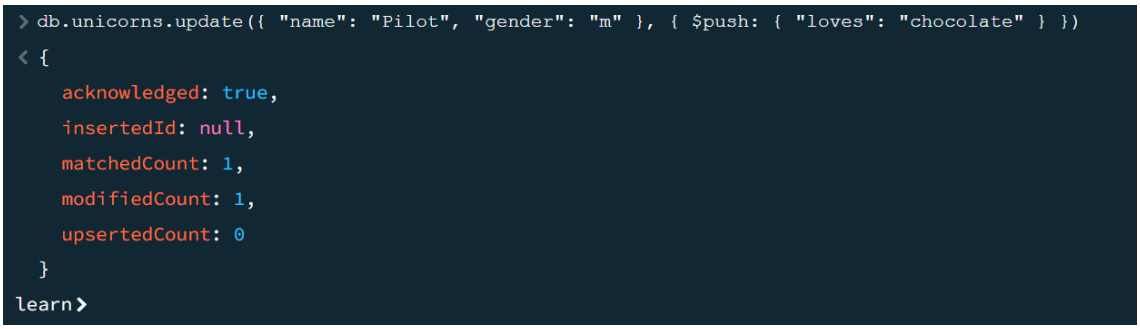
```
db.towns.update({name : "Portland"}, {$unset: {"mayor.party": 1}})
```



Практическое задание 3.3.6

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

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



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.*

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

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

Практическое задание 3.4.1

1. *Создайте коллекцию towns, включающую следующие документы*
2. Удалите документы с беспартийными мэрами.
3. Проверьте содержание коллекции.
4. Очистите коллекцию.
5. Просмотрите список доступных коллекций.

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

```
> db.towns.remove({"mayor.party": { $exists:false }})
< DeprecationWarning: Collection.remove() is deprecated.
< {
    acknowledged: true,
    deletedCount: 2
  }
learn>
```

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.*

```
db.zones.insert([
    {
        _id: "forest",
        name: "Magic Forest",
        description: "A mystical forest where unicorns live
peacefully."
    },
    {
        _id: "meadow",
        name: "Enchanted Meadow",
        description: "A beautiful meadow where unicorns roam and
play."
    },
    {
        _id: "mountain",
        name: "Majestic Mountain",
        description: "A grand mountain where unicorns graze and
rest."
    }
])

var forestZoneId = db.zones.findOne({_id: "forest"})._id;
var meadowZoneId = db.zones.findOne({_id: "meadow"})._id;
var mountainZoneId = db.zones.findOne({_id: "mountain"})._id;

db.unicorns.updateMany({}, [
    {$set: {$ref: "zones", $id: forestZoneId}},
    {$set: {$ref: "zones", $id: meadowZoneId}},
    {$set: {$ref: "zones", $id: mountainZoneId}}
])
```

```
> db.unicorns.updateMany({}, [
      {$set:  {$ref: "zones", $id: forestZoneId}},
      {$set:  {$ref: "zones", $id: meadowZoneId}},
      {$set:  {$ref: "zones", $id: mountainZoneId}}
  ])
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 12,
    modifiedCount: 12,
    upsertedCount: 0
  }
learn>
```

Практическое задание 4.2.1

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

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

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

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.*

```
db.unicorns.getIndexes();
db.unicorns.dropIndexes();
```

```
> db.unicorns.getIndexes();
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
  ]
> db.unicorns.dropIndexes();
< {
    nIndexesWas: 2,
    msg: 'non-_id indexes dropped for collection',
    ok: 1
  }
learn>
```

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?*

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

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

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

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

```
> db.numbers.find().count()
< 100000
> db.numbers.explain("executionStats").find().sort({value: -1}).limit(4)
```

```
executionStats: {
  executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 202,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
  executionStages: {
    stage: 'sort',
    planNodeId: 2,
    nReturned: 4,
    executionTimeMillisEstimate: 202,
    opens: 1,
```

```
> db.numbers.createIndex({value: 1})
< value_1
> db.numbers.getIndexes()
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { value: 1 }, name: 'value_1' }
  ]
>
  db.numbers.explain("executionStats").find().sort({value: -1}).limit(4)
< {
    explainVersion: '2',
    queryPlanner: {
```


executionStats: {
executionSuccess: true,
nReturned: 4,
executionTimeMillis: 15,
totalKeysExamined: 4,
totalDocsExamined: 4,
executionStages: {
stage: 'limit',
planNodeId: 3,
nReturned: 4,
executionTimeMillisEstimate: 10,
opens: 1,
closes: 1,
saveState: 0,

Вывод

В ходе выполнения лабораторной работы я овладел практическими навыками работы с базой данных MongoDB. Я освоил работу с вложенными объектами в коллекциях, осуществлял агрегации и изменения данных, а также работал со ссылками и индексами.