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»

Обучающийся: Гайдук Алина Сергеевна
Факультет прикладной информатики
Группа K3241
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2025

Оглавление

Введение

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.

Программное обеспечение: СУБД MongoDB 4+, 8.0.4 (последняя).

CRUD-операции в СУБД MongoDB. Вставка данных. Выборка данных

Практическое задание 2.1.1:

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

```
> use learn
< switched to db learn
> 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('6825129cc335a9c5472798e8')
    }
  }
learn>
```

3. Используя второй способ, вставьте в коллекцию единорогов документ:

```
{name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
```

```
> db.unicorns.insertOne({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
< {
    acknowledged: true,
    insertedId: ObjectId('6825137ec335a9c5472798e9')
  }
learn>
```

4. Проверьте содержимое коллекции с помощью метода find.

```
> db.unicorns.find()
< {
    _id: ObjectId('6825129cc335a9c5472798de'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    _id: ObjectId('6825129cc335a9c5472798df'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e0'),
    name: 'Unicrom',
    loves: [
      'energon',
      'redbull'
    ],
    weight: 984,
    gender: 'm',
    vampires: 182
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e1'),
    name: 'Roooooodles',
    loves: [
      'apple'
    ],
    weight: 575,
    gender: 'm',
    vampires: 99
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e2'),
    name: 'Solnara',
    loves: [
      'apple',
      'carrot',
      'chocolate'
    ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
```

```
{
  _id: ObjectId('6825129cc335a9c5472798e3'),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 733,
  gender: 'f',
  vampires: 40
}
{
  _id: ObjectId('6825129cc335a9c5472798e4'),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 39
}
{
  _id: ObjectId('6825129cc335a9c5472798e5'),
  name: 'Raleigh',
  loves: [
    'apple',
    'sugar'
  ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
{
  _id: ObjectId('6825129cc335a9c5472798e6'),
  name: 'Leia',
  loves: [
    'apple',
    'watermelon'
  ],
  weight: 601,
  gender: 'f',
  vampires: 33
}
{
  _id: ObjectId('6825129cc335a9c5472798e7'),
  name: 'Pilot',
  loves: [
    'apple',
    'watermelon'
  ],
  weight: 650,
  gender: 'm',
  vampires: 54
}
```

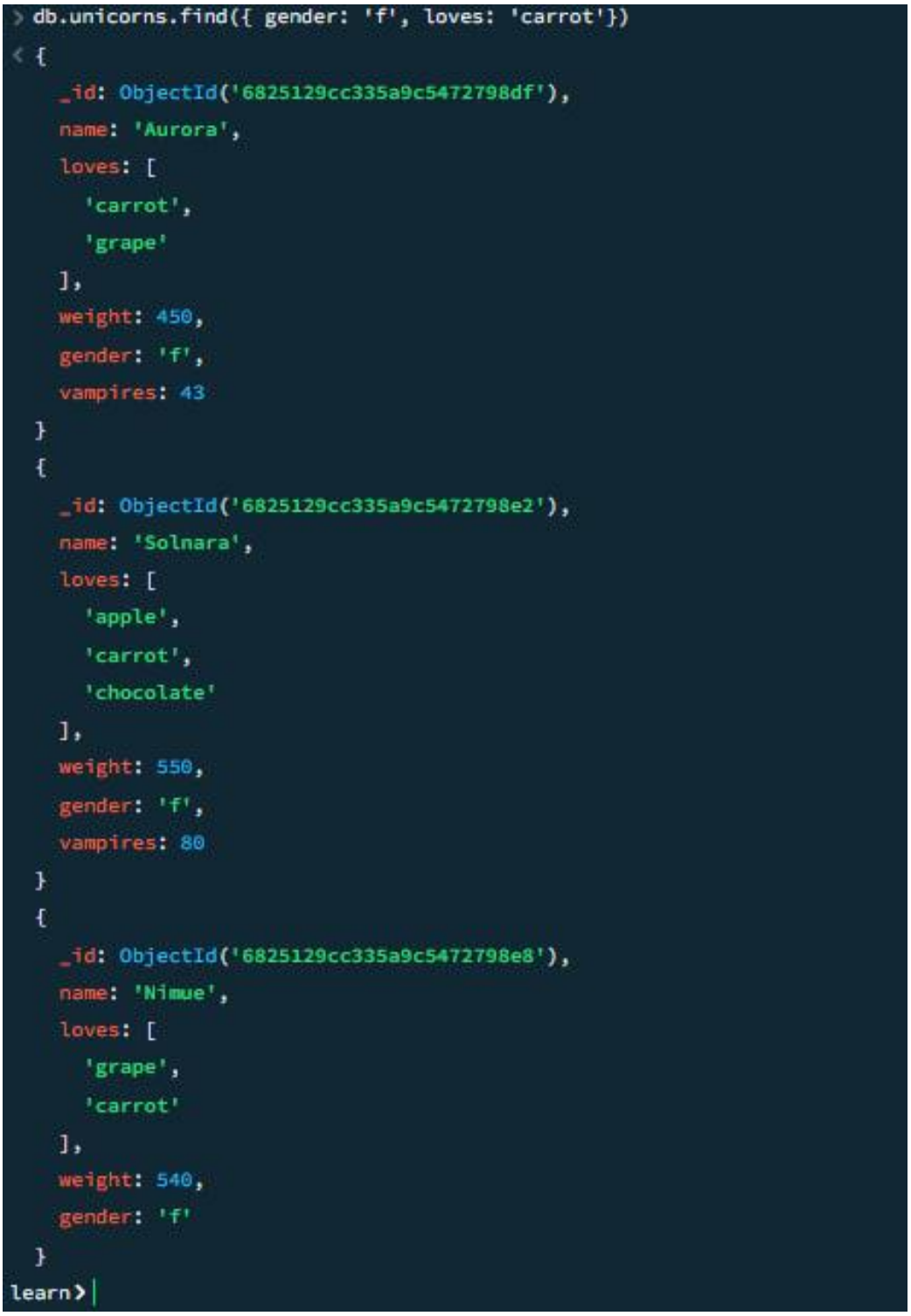
Практическое задание 2.2.1:

1. Сформируйте запросы для вывода списков самцов и самок единорогов. Отсортируйте списки по имени.


db.unicorns.find({ gender: 'm' }).sort({ name: 1 })
{
_id: ObjectId('6825137ec335a9c5472798e9'),
name: 'Dunx',
loves: [
'grape',
'watermelon'
],
weight: 704,
gender: 'm',
vampires: 165
}
{
_id: ObjectId('6825129cc335a9c5472798de'),
name: 'Horny',
loves: [
'carrot',
'papaya'
],
weight: 600,
gender: 'm',
vampires: 63
}
{
_id: ObjectId('6825129cc335a9c5472798e4'),
name: 'Kenny',
loves: [
'grape',
'lemon'
],
weight: 690,
gender: 'm',
vampires: 39
}
{
_id: ObjectId('6825129cc335a9c5472798e7'),
name: 'Pilot',
loves: [
'apple',
'watermelon'
],
weight: 650,
gender: 'm',
vampires: 54
}
{
_id: ObjectId('6825129cc335a9c5472798e5'),
name: 'Raleigh',
loves: [
'apple',
'sugar'
],
weight: 421,
gender: 'm',
vampires: 2
}

2. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.

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

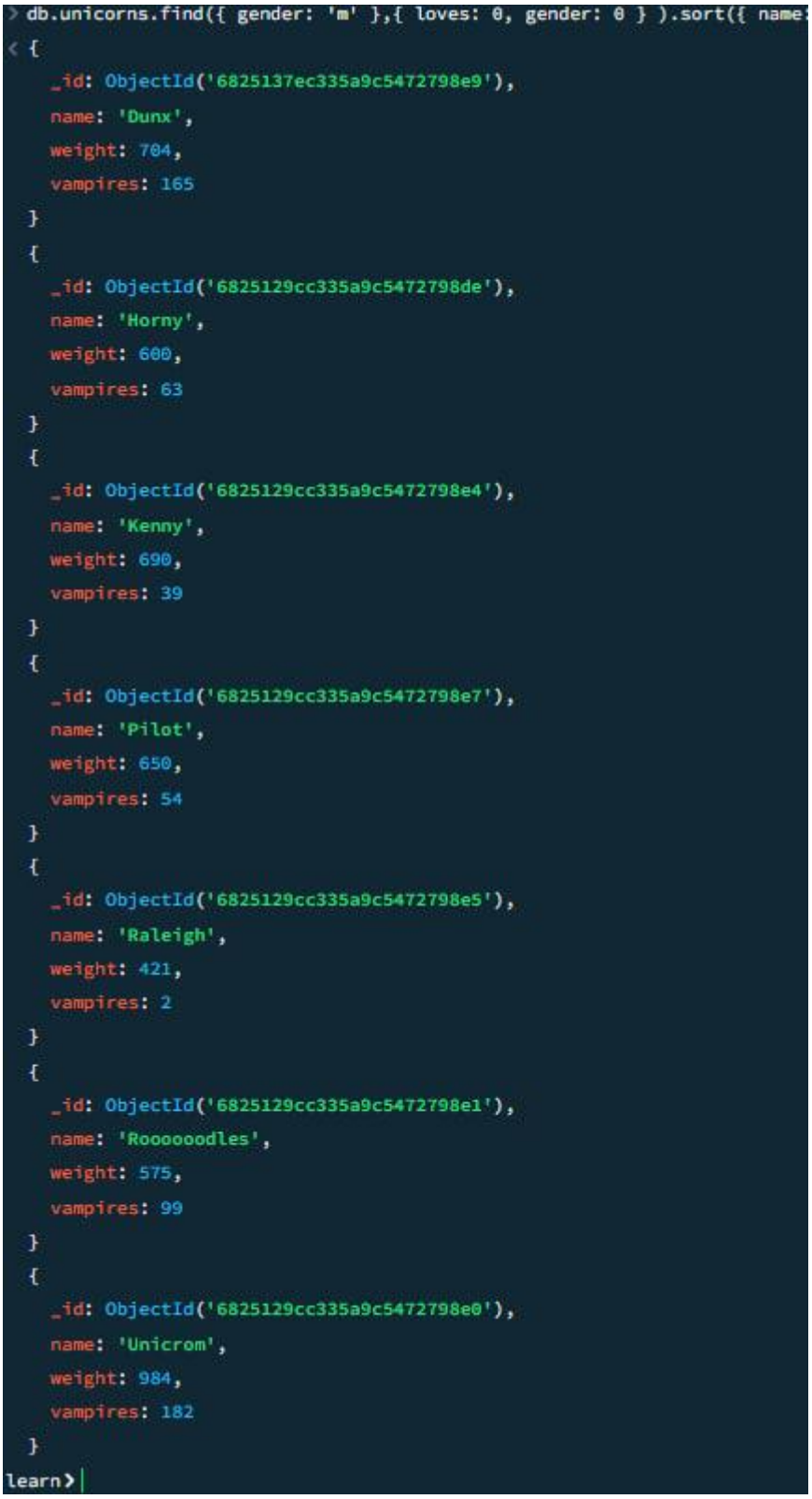


```
> db.unicorns.findOne({ gender: 'f', loves: 'carrot'})
< {
    _id: ObjectId('6825129cc335a9c5472798df'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

```
> db.unicorns.find({ gender: 'f', loves: 'carrot'}).limit(1)
< {
    _id: ObjectId('6825129cc335a9c5472798df'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
learn>
```

Практическое задание 2.2.2:

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



Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({ $natural: -1 })
< {
    _id: ObjectId('6825137ec335a9c5472798e9'),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e8'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e7'),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e6'),
    name: 'Leia',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e5'),
    name: 'Raleigh',
    loves: [
      'apple',
      'sugar'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
```

```
{
  _id: ObjectId('6825129cc335a9c5472798e4'),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 39
}
{
  _id: ObjectId('6825129cc335a9c5472798e3'),
  name: 'Ayna',
  loves: [
    'strawberry',
    'lemon'
  ],
  weight: 733,
  gender: 'f',
  vampires: 40
}
{
  _id: ObjectId('6825129cc335a9c5472798e2'),
  name: 'Solnara',
  loves: [
    'apple',
    'carrot',
    'chocolate'
  ],
  weight: 550,
  gender: 'f',
  vampires: 80
}
{
  _id: ObjectId('6825129cc335a9c5472798e1'),
  name: 'Roooooodles',
  loves: [
    'apple'
  ],
  weight: 575,
  gender: 'm',
  vampires: 99
}
```


{
_id: ObjectId('6825129cc335a9c5472798e0'),
name: 'Unicrom',
loves: [
'energon',
'redbull'
],
weight: 984,
gender: 'm',
vampires: 182
}
{
_id: ObjectId('6825129cc335a9c5472798df'),
name: 'Aurora',
loves: [
'carrot',
'grape'
],
weight: 450,
gender: 'f',
vampires: 43
}
{
_id: ObjectId('6825129cc335a9c5472798de'),
name: 'Horny',
loves: [
'carrot',
'papaya'
],
weight: 600,
gender: 'm',
vampires: 63
}
learn>

Практическое задание 2.2.4:

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

```
> 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
  }
  {
    name: 'Solnara',
    loves: [
      'apple'
    ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
  {
    name: 'Ayna',
    loves: [
      'strawberry'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
```

```
{
  name: 'Kenny',
  loves: [
    'grape'
  ],
  weight: 690,
  gender: 'm',
  vampires: 39
}
{
  name: 'Raleigh',
  loves: [
    'apple'
  ],
  weight: 421,
  gender: 'm',
  vampires: 2
}
{
  name: 'Leia',
  loves: [
    'apple'
  ],
  weight: 601,
  gender: 'f',
  vampires: 33
}
{
  name: 'Pilot',
  loves: [
    'apple'
  ],
  weight: 650,
  gender: 'm',
  vampires: 54
}
{
  name: 'Nimue',
  loves: [
    'grape'
  ],
  weight: 540,
  gender: 'f'
}
{
  name: 'Dunx',
  loves: [
    'grape'
  ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
```

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 })
< {
    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'
  }
learn>
```

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 })
< {
    name: 'Kenny',
    loves: [
      'grape',
      'lemon'
    ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
learn>
```

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ vampires.

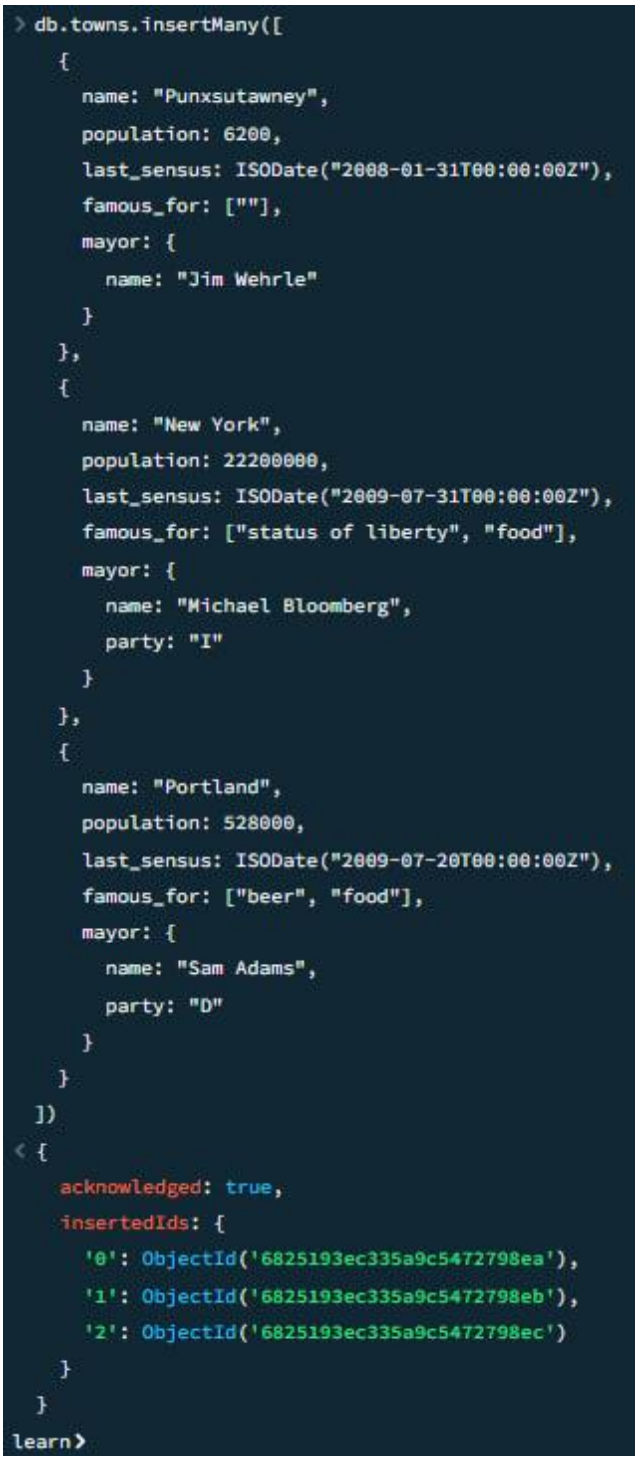
Практическое задание 2.3.4:

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

Запросы к базе данных MongoDB. Выборка данных. Вложенные объекты. Использование курсоров. Агрегированные запросы. Изменение данных

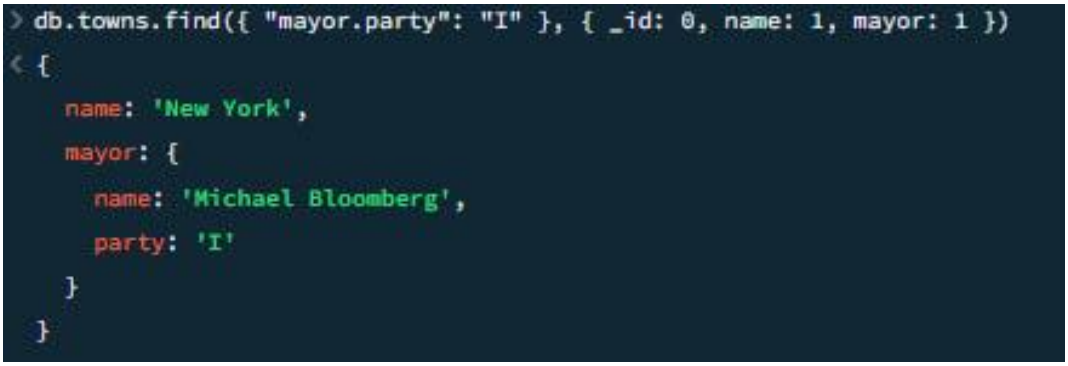
Практическое задание 3.1.1:

1. Создайте коллекцию towns.

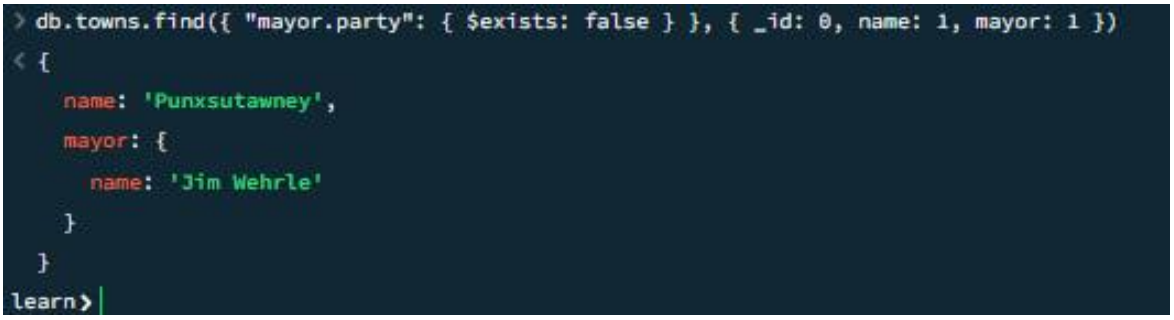


```
> db.towns.insertMany([
    {
      name: "Punxsutawney",
      population: 6200,
      last_sensus: ISODate("2008-01-31T00:00:00Z"),
      famous_for: [""],
      mayor: {
        name: "Jim Wehrle"
      }
    },
    {
      name: "New York",
      population: 22200000,
      last_sensus: ISODate("2009-07-31T00:00:00Z"),
      famous_for: ["status of liberty", "food"],
      mayor: {
        name: "Michael Bloomberg",
        party: "I"
      }
    },
    {
      name: "Portland",
      population: 528000,
      last_sensus: ISODate("2009-07-20T00:00:00Z"),
      famous_for: ["beer", "food"],
      mayor: {
        name: "Sam Adams",
        party: "D"
      }
    }
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('6825193ec335a9c5472798ea'),
      '1': ObjectId('6825193ec335a9c5472798eb'),
      '2': ObjectId('6825193ec335a9c5472798ec')
    }
  }
learn>
```

2. Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре.

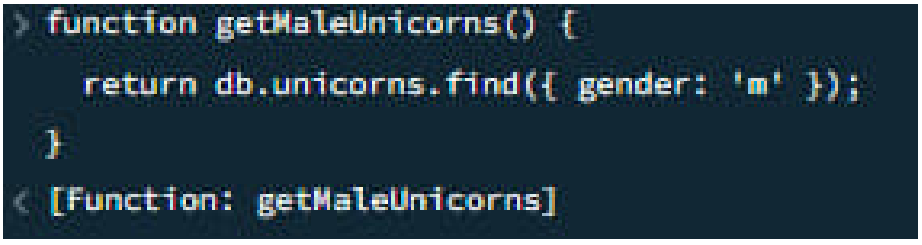


3. Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре.



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.

```
> var cursor = getMaleUnicorns().sort({ name: 1 }).limit(2);
> cursor.forEach(function(obj) {print(obj.name);});
< Dunx
< Horny
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()
< 2
learn>
```

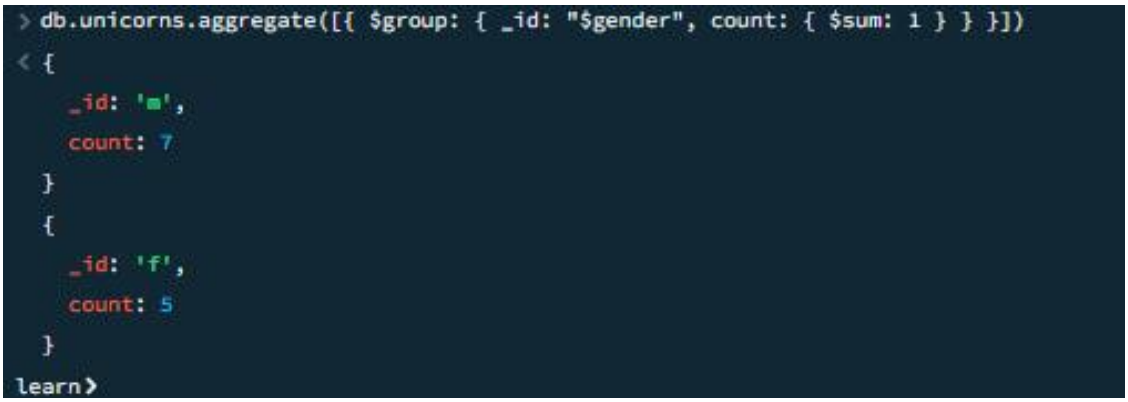
Практическое задание 3.2.2:

Вывести список предпочтений.

```
> 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.

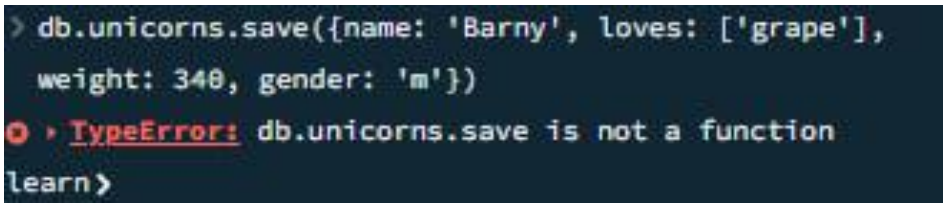


Практическое задание 3.3.1:

1. Выполнить команду:

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

2. Проверить содержимое коллекции unicorns.



Метод save() был удалён из новых MongoDB-драйверов и mongosh. В mongosh его просто нет, попытка вызова вызывает ошибку "save is not a function".

Практическое задание 3.3.2:

1. Для самки единорога Ayna внести изменения в БД: теперь ее вес 800, она убила 51 вампира.
2. 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.

Практическое задание 3.3.4:

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

```
> db.unicorns.updateMany({ gender: "m" }, { $inc: { vampires: 5 } })
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 7,
    modifiedCount: 7,
    upsertedCount: 0
  }
> db.unicorns.find({ gender: "m" })
< {
    _id: ObjectId('6825129cc335a9c5472798de'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e0'),
    name: 'Unicrom',
    loves: [
      'energon',
      'redbull'
    ],
    weight: 984,
    gender: 'm',
    vampires: 187
  }
  {
    _id: ObjectId('6825129cc335a9c5472798e1'),
    name: 'Roooooodles',
    loves: [
      'apple'
    ],
    weight: 575,
    gender: 'm',
    vampires: 104
  }
```

```
{
  _id: ObjectId('6825129cc335a9c5472798e4'),
  name: 'Kenny',
  loves: [
    'grape',
    'lemon'
  ],
  weight: 690,
  gender: 'm',
  vampires: 44
}
{
  _id: ObjectId('6825129cc335a9c5472798e5'),
  name: 'Raleigh',
  loves: [
    'redbull'
  ],
  weight: 421,
  gender: 'm',
  vampires: 7
}
{
  _id: ObjectId('6825129cc335a9c5472798e7'),
  name: 'Pilot',
  loves: [
    'apple',
    'watermelon'
  ],
  weight: 650,
  gender: 'm',
  vampires: 59
}
{
  _id: ObjectId('6825137ec335a9c5472798e9'),
  name: 'Dunx',
  loves: [
    'grape',
    'watermelon'
  ],
  weight: 704,
  gender: 'm',
  vampires: 170
}
```

Практическое задание 3.3.5:

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

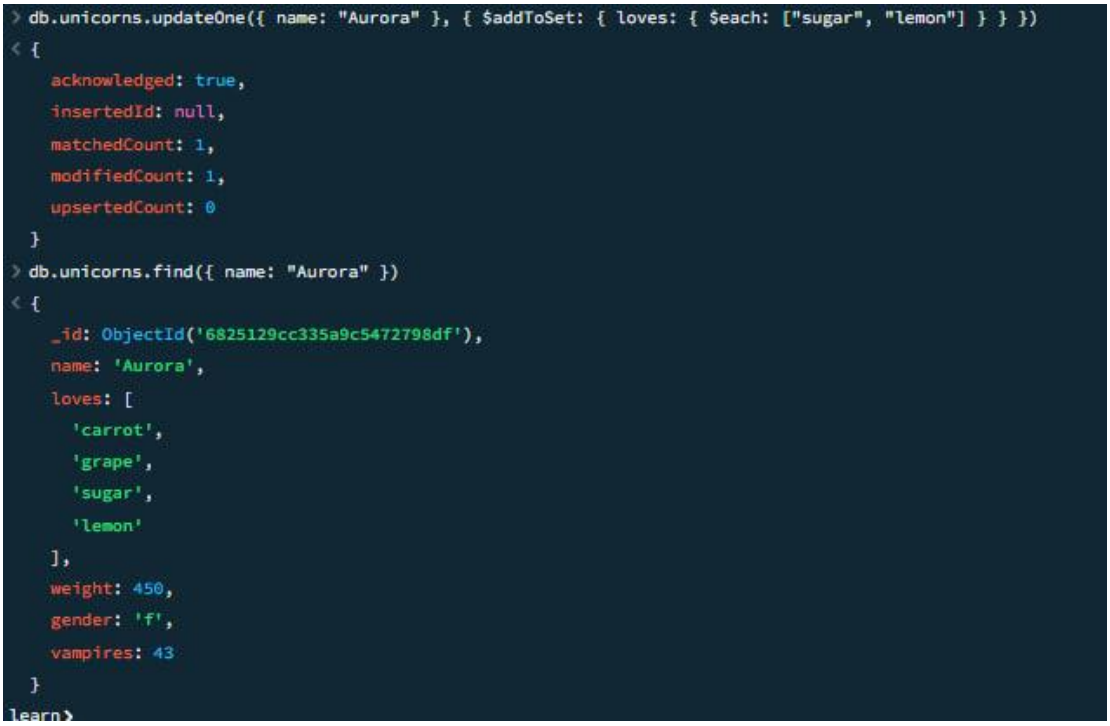
Практическое задание 3.3.6:

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

Проверить содержимое коллекции unicorns.

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.



Практическое задание 3.4.1:

1. Создайте коллекцию towns.
2. Удалите документы с беспартийными мэрами.

```
> db.towns.insertMany([
    {
      name: "Punxsutawney",
      population: 6200,
      last_sensus: ISODate("2008-01-31"),
      famous_for: ["phil the groundhog"],
      mayor: { name: "Jim Wehrle" }
    },
    {
      name: "New York",
      population: 22200000,
      last_sensus: ISODate("2009-07-31"),
      famous_for: ["status of liberty", "food"],
      mayor: { name: "Michael Bloomberg", party: "I" }
    },
    {
      name: "Portland",
      population: 528000,
      last_sensus: ISODate("2009-07-20"),
      famous_for: ["beer", "food"],
      mayor: { name: "Sam Adams", party: "D" }
    }
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('68251f5fc335a9c5472798ed'),
      '1': ObjectId('68251f5fc335a9c5472798ee'),
      '2': ObjectId('68251f5fc335a9c5472798ef')
    }
  }
> db.towns.deleteMany({ "mayor.party": { $exists: false } })
< {
    acknowledged: true,
    deletedCount: 1
  }
```

3. Проверьте содержание коллекции.
4. Очистите коллекцию.
5. Просмотрите список доступных коллекций.

Ссылки и работа с индексами в базе данных MongoDB

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: "field", name: "field with a shining sun", description: "it's so warm and pleasant there." }, { _id: "mountain", name: "snowy mountain", description: "it's ok if you like skiing" }])
< {
    acknowledged: true,
    insertedIds: {
      '0': 'field',
      '1': 'mountain'
    }
  }
```

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

```
> db.unicorns.updateOne({ name: "Aurora" }, { $set: { habitat_zone: { $ref: "zones", $id: "mountain" } } })
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.updateOne({ name: "Horny" }, { $set: { habitat_zone: { $ref: "zones", $id: "field" } } });
  db.unicorns.updateOne({ name: "Unicorn" }, { $set: { habitat_zone: { $ref: "zones", $id: "field" } } });
  db.unicorns.updateOne({ name: "Twinkle" }, { $set: { habitat_zone: { $ref: "zones", $id: "mountain" } } });
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 0,
    modifiedCount: 0,
    upsertedCount: 0
  }
```

3. Проверьте содержание коллекции единорогов.

```
> db.unicorns.find()
< {
    _id: ObjectId('6825129cc335a9c5472798de'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habitat_zone: DBRef('zones', 'field')
  }
  {
    _id: ObjectId('6825129cc335a9c5472798df'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    habitat_zone: DBRef('zones', 'mountain')
  }
```

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 })
< [ 'name_1' ]
learn>
```

Практическое задание 4.3.1:

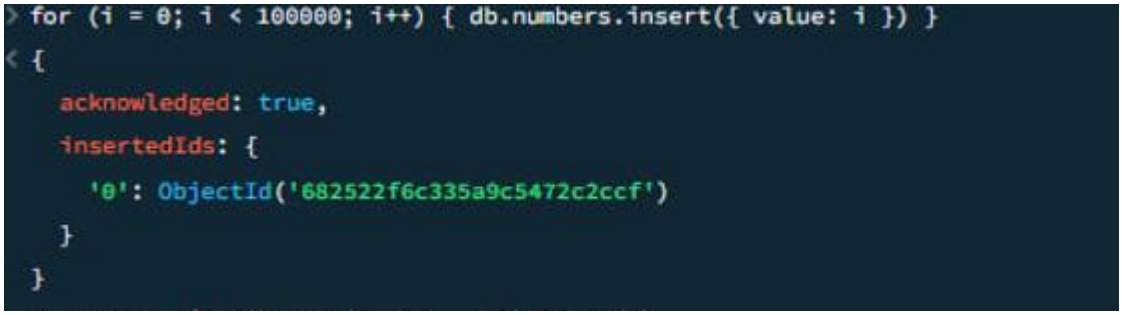
1. Получите информацию о всех индексах коллекции unicorns .
2. Удалите все индексы, кроме индекса для идентификатора.
3. Попытайтесь удалить индекс для идентификатора.

```
> db.unicorns.getIndexes()
< [
    { 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
  }
> db.unicorns.dropIndex("_id_")
✖ ▸ MongoServerError[InvalidOptions]: cannot drop _id index
learn>
```

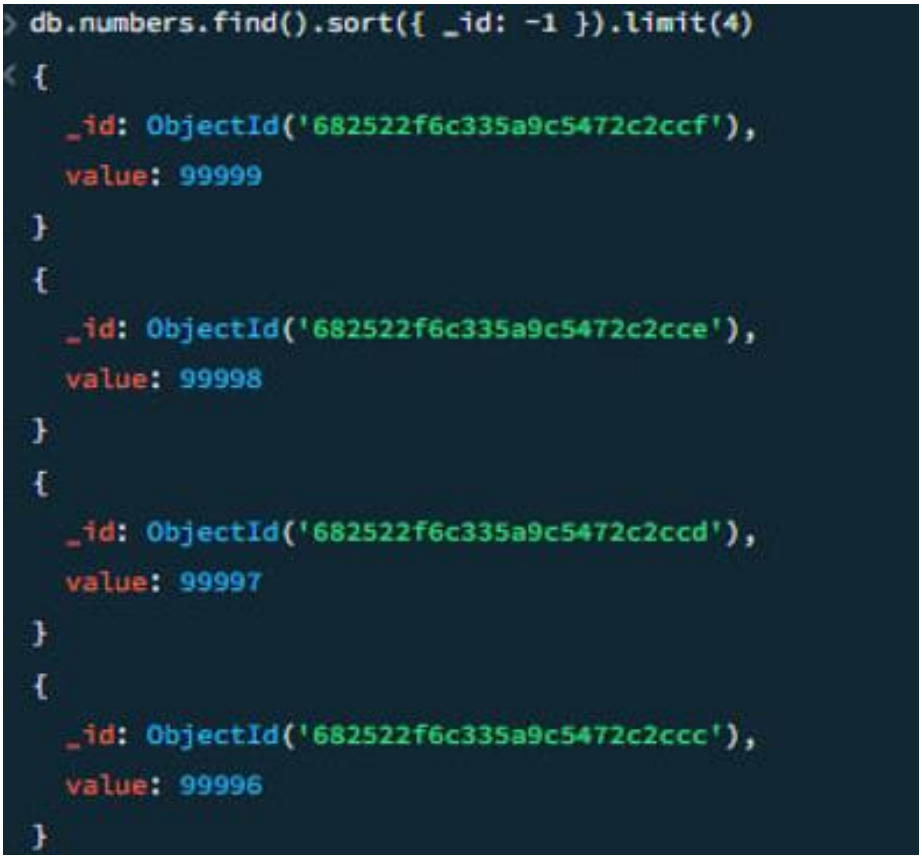
Практическое задание 4.4.1:

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)

```
> db.numbers.explain("executionStats").find({ value: 99999 })
< {
    explainVersion: '1',
    queryPlanner: {
      namespace: 'learn.numbers',
      parsedQuery: {
        value: {
          '$eq': 99999
        }
      },
      indexFilterSet: false,
      queryHash: 'FBBD8DD0',
      planCacheShapeHash: 'FBBD8DD0',
      planCacheKey: 'CCC7FD70',
      optimizationTimeMillis: 0,
      maxIndexedOrSolutionsReached: false,
      maxIndexedAndSolutionsReached: false,
      maxScansToExplodeReached: false,
      prunedSimilarIndexes: false,
      winningPlan: {
        isCached: false,
        stage: 'COLLSCAN',
        filter: {
          value: {
            '$eq': 99999
          }
        },
        direction: 'forward'
      },
      rejectedPlans: []
    },
    executionStats: {
      executionSuccess: true,
      nReturned: 3,
      executionTimeMillis: 245,
```

4. Создайте индекс для ключа value.
5. Получите информацию обо 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?

Запрос с индексом выполняется быстрее, а значит он более эффективен, чем первый запрос.

Вывод

В ходе лабораторной работы были изучены основные операции в MongoDB: добавление, обновление, удаление документов, работа с массивами и вложенными структурами, создание индексов и анализ производительности запросов. Получены навыки связывания коллекций, использования операторов $set, $unset, $push, $addToSet, $inc, а также анализа выполнения запросов с помощью explain. Работа позволила освоить базовые приёмы эффективной работы с документно-ориентированной базой данных MongoDB.