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

Отчет

по лабораторной работе №5

«Работа с БД в СУБД MongoDB»
по дисциплине «Проектирование и реализация баз данных»

Авторы: Антонова М. А.
Факультет: ФИКТ
Группа: К32422
Преподаватель: Говорова М.М.

Санкт-Петербург, 2023

1 Описание работы

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)

2 Практическое задание

Практическое задание 8.1.1:

1) Создайте базу данных 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: {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
4) Проверьте содержимое коллекции с помощью метода find.

Практическое задание 8.1.2:

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.

Практическое задание 8.1.3:

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

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

Практическое задание 8.1.5:

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

Практическое задание 8.1.6:

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

Практическое задание 8.1.7:

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

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

Практическое задание 8.1.9:

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

Практическое задание 8.2.1:

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.

Практическое задание 8.2.2:

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:

   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', 44), 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.insert ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

Практическое задание 8.2.4:

Вывести список предпочтений.

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

Практическое задание 8.2.6:

1) Выполнить команду:

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

2) Проверить содержимое коллекции unicorns.

Практическое задание 8.2.7:

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

2) Проверить содержимое коллекции unicorns.

Практическое задание 8.2.8:

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

Практическое задание 8.2.9:

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

Практическое задание 8.2.10:

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

Практическое задание 8.2.11:

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

Практическое задание 8.2.12:

1) Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.
2) Проверить содержимое коллекции unicorns.

Практическое задание 8.2.13:

1) Создайте коллекцию towns, включающую следующие документы:

{name: "Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for: ["phil the groundhog"], 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) Удалите документы с беспартийными мэрами.
3) Проверьте содержание коллекции.
4) Очистите коллекцию.

5) Просмотрите список доступных коллекций.

Практическое задание 8.3.1:

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:

   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', 44), 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.insert {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}

Практическое задание 8.3.2:

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

Практическое задание 8.3.3:

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

Практическое задание 8.3.4:

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?

3 Выполнение

Практическое задание 8.1.1:

1) Создайте базу данных 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'});

```
test> use learn
switched to db learn
learn> db.createCollection("unicorns")
{ ok: 1 }
learn> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600,
... gender: 'm', vampires: 63});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677ad73b3f78129efe51a6") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450,
... gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677aed3b3f78129efe51a7") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight:
... 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677aff3b3f78129efe51a8") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575,
... gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677b0d3b3f78129efe51a9") }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'],
... weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677b3a3b3f78129efe51aa") }
}
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733,
... gender: 'f', vampires: 40});
```

Рисунок 1 – Практическое задание 8.1.1

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
learn> db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733,
... gender: 'f', vampires: 40});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677b4d3b3f78129efe51ab") }
}
learn> db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight:
... 690, gender: 'm', vampires: 39});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677b593b3f78129efe51ac") }
}
learn> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421,
... gender: 'm', vampires: 2});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677b633b3f78129efe51ad") }
}
learn> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601,
... gender: 'f', vampires: 33});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677c2d3b3f78129efe51ae") }
}
learn> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650,
... gender: 'm', vampires: 54});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677c383b3f78129efe51af") }
}
learn> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540,
... gender: 'f'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64677c4a3b3f78129efe51b0") }
}
```

Рисунок 2 – Практическое задание 8.1.1

3) Используя второй способ, вставьте в коллекцию единорогов документ: {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}

```
  insertedIds: { '0': ObjectId("64677c4a3b3f78129efe51b0") }
}
learn> doc={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.insertOne(doc)
{
  acknowledged: true,
  insertedId: ObjectId("64677cb43b3f78129efe51b1")
}
```

Рисунок 3 – Практическое задание 8.1.1, пункт2

4) Проверьте содержимое коллекции с помощью метода find.

```
}
learn> db.unicorns.find()
[
  {
    _id: ObjectId("64677ad73b3f78129efe51a6"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("64677aed3b3f78129efe51a7"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId("64677aff3b3f78129efe51a8"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("64677b0d3b3f78129efe51a9"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("64677b3a3b3f78129efe51aa"),
    name: 'Solnara',
    loves: [ 'apple', 'carrot', 'chocolate' ],
```

Рисунок 4 – Практическое задание 8.1.1

Рисунок 5 – Практическое задание 8.1.1

```
    vampires: 54
  },
  {
    _id: ObjectId("64677c4a3b3f78129efe51b0"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("64677cb43b3f78129efe51b1"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
```

Рисунок 6 – Практическое задание 8.1.1

Практическое задание 8.1.2:

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: 'm'},{_id: 0, name: 1}).sort({name: 1}).toArray()
[
  { name: 'Dunx' },
  { name: 'Horny' },
  { name: 'Kenny' },
  { name: 'Pilot' },
  { name: 'Raleigh' },
  { name: 'Roooooodles' },
  { name: 'Unicrom' }
]
learn> db.unicorns.find({gender: 'f'},{_id: 0, name: 1}).sort({name: 1}).limit(3).toArray()
[ { name: 'Aurora' }, { name: 'Ayna' }, { name: 'Leia' } ]
```

Рисунок 7 – Практическое задание 8.1.2

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

```
learn> db.unicorns.find({gender: 'f',loves: 'carrot'}).limit(1)
[
  {
    _id: ObjectId("64677aed3b3f78129efe51a7"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn> db.unicorns.findOne({gender: 'f',loves: 'carrot'})
{
  _id: ObjectId("64677aed3b3f78129efe51a7"),
  name: 'Aurora',
  loves: [ 'carrot', 'grape' ],
  weight: 450,
  gender: 'f',
  vampires: 43
}
```

Рисунок 8 – Практическое задание 8.1.2

Практическое задание 8.1.3:

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

```
learn> db.unicorns.find({gender: 'm'},{loves: 0, gender: 0})
[
  {
    _id: ObjectId("64677ad73b3f78129efe51a6"),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId("64677aff3b3f78129efe51a8"),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId("64677b0d3b3f78129efe51a9"),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId("64677b593b3f78129efe51ac"),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId("64677b633b3f78129efe51ad"),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId("64677c383b3f78129efe51af"),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  },
  {
    _id: ObjectId("64677cb43b3f78129efe51b1"),
    name: 'Dunx',
    weight: 704,
    vampires: 165
  }
]
```

Рисунок 9 – Практическое задание 8.1.3

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
learn> db.unicorns.find({},{_id: 0, name: 1}).sort({$natural: -1}).toArray()
[
  { name: 'Dunx' },
  { name: 'Nimue' },
  { name: 'Pilot' },
  { name: 'Leia' },
  { name: 'Raleigh' },
  { name: 'Kenny' },
  { name: 'Ayna' },
  { name: 'Solnara' },
  { name: 'Roooooodles' },
  { name: 'Unicrom' },
  { name: 'Aurora' },
  { name: 'Horny' }
]
```

Рисунок 10 – Практическое задание 8.1.4

Практическое задание 8.1.5 Вывести список единорогов с названием первого любимого предпочтения

```
learn> db.unicorns.find({},{_id: 0, loves: {$slice: 1}}).sort({$natural: -1}).toArray()
[
  {
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  { name: 'Nimue', loves: [ 'grape' ], weight: 540, gender: 'f' },
  {
    name: 'Pilot',
    loves: [ 'apple' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    name: 'Leia',
    loves: [ 'apple' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  },
  {
    name: 'Raleigh',
    loves: [ 'apple' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    name: 'Ayna',
    loves: [ 'strawberry' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
```

```
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    name: 'Solnara',
    loves: [ 'apple' ],
    weight: 550,
    gender: 'f',
    vampires: 80
  },
  {
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    name: 'Unicrom',
    loves: [ 'energon' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    name: 'Aurora',
    loves: [ 'carrot' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
]
```

Рисунок 11– Практическое задание 8.1.5

Практическое задание 8.1.6:

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

```
learn> db.unicorns.find({gender: 'f', weight: {$gt:500,$lt: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>
```

Рисунок 12 – Практическое задание 8.1.6

Практическое задание 8.1.7:

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

```
learn> db.unicorns.find({gender: 'm', weight: {$gt:500},loves: ['grape','lemon']},{_id: 0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Рисунок 13 – Практическое задание 8.1.7

Можно использовать $all, чтобы порядок предпочтений не влиял на выбор

```
learn> db.unicorns.find({gender: 'm', weight: {$gt:500},loves: {$all:['lemon','grape']}},{_id: 0})
[
  {
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Рисунок 14 – Практическое задание 8.1.7

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({vampires: {$exists: false}},{_id:0})
[
  {
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
```

Рисунок 15 – Практическое задание 8.1.8

Практическое задание 8.1.9:

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

```
learn> db.unicorns.find({gender: 'm'},{_id: 0,name: 1, loves: {$slice: 1}}).sort({name: 1}).toArray()
[
  { 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' ] }
]
```

Рисунок 16 – Практическое задание 8.1.8

Практическое задание 8.2.1:

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"}}

```
test> use learn
switched to db learn
learn> db.createCollection("towns")
{ ok: 1 }
```

Рисунок 17 – Практическое задание 8.2.1

```
learn> 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("6468981d861899e6fc052aea"),
    '1': ObjectId("6468981d861899e6fc052aeb"),
    '2': ObjectId("6468981d861899e6fc052aec")
  }
}
```

Рисунок 18 – Практическое задание 8.2.1

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

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

Рисунок 19 – Практическое задание 8.2.1

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

Рисунок 20 – Практическое задание 8.2.1

Практическое задание 8.2.2:

1. Сформировать функцию для вывода списка самцов единорогов.

```
learn> fn = function() {return this.gender == 'm';}
[Function: fn]
learn> var cursor = db.unicorns.find({"$where": fn}).sort({name: 1}).limit(2);null;
null
learn> cursor.forEach(function(obj) {print(obj) })
{
  _id: ObjectId("64677cb43b3f78129efe51b1"),
  name: 'Dunx',
  loves: [ 'grape', 'watermelon' ],
  weight: 704,
  gender: 'm',
  vampires: 165
}
{
  _id: ObjectId("64677ad73b3f78129efe51a6"),
  name: 'Horny',
  loves: [ 'carrot', 'papaya' ],
  weight: 600,
  gender: 'm',
  vampires: 63
}
```

Рисунок 21 – Практическое задание 8.2.2

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг

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

Рисунок 22 – Практическое задание 8.2.3

Практическое задание 8.2.4:

Вывести список предпочтений.

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

Рисунок 23 – Практическое задание 8.2.4

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

Рисунок 24 – Практическое задание 8.2.4

Можно использовать как такие кавычки “, так и такие кавычки ‘. На выполнение это не влияет

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

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

Рисунок 25 – Практическое задание 8.2.5

Практическое задание 8.2.6:

1. Выполнить команду:

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

2. Проверить содержимое коллекции unicorns.

Было количество:

```
learn> db.unicorns.count()
DeprecationWarning: Collection.count() is deprecated. Use countDocuments or estimatedDocumentCount.
12
learn> db.unicorns.countDocuments()
12
```

Рисунок 26 – Исходное состояние коллекции

Стало после save:

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

Необходимо использовать insertOne

```
learn> db.unicorns.insertOne({ name: 'Barny', loves: ['grape'], weight: 340, gender: 'm' })
{
  acknowledged: true,
  insertedId: ObjectId("64689f2d861899e6fc052aed")
}
```

Стало количество:

```
learn> db.unicorns.countDocuments()
13
learn>
```

Рисунок 27 – Конечное состояние коллекции

Таким образом, метод save() является устаревшим, и его нельзя применить в данном задании

Практическое задание 8.2.7:

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

2. Проверить содержимое коллекции unicorns. db.unicorns.update({name : "Ayna"}, {weight: 800, vampires: 51})

```
learn> db.unicorns.update({name: 'Ayna'},{$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
}
```

Рисунок 28 – Практическое задание 8.2.7

Метод update() является устаревшим, необходимо использовать updateOne

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

Рисунок 29 – Практическое задание 8.2.7

Проверяем данные:

```
learn> db.unicorns.find({name: 'Ayna'},{_id: 0})
[
  {
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
```

Рисунок 30 – Проверка

Практическое задание 8.2.8:

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

2. Проверить содержимое коллекции unicorns.

Вносим изменения:

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

Рисунок 31 – Внесение изменений

Проверка содержимого:

```
learn> db.unicorns.find({name: 'Raleigh'},{_id: 0})
[
  {
    name: 'Raleigh',
    loves: [ 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

Рисунок 32 – Проверка

Практическое задание 8.2.9:

1. Всем самцам единорогов увеличить количество убитых вампиров на 5.

2. Проверить содержимое коллекции unicorns

Было:

```
learn> db.unicorns.find({gender: 'm'},{_id: 0, name: 1, vampires: 1}).toArray()
[
  { name: 'Horny', vampires: 63 },
  { name: 'Unicrom', vampires: 182 },
  { name: 'Roooooodles', vampires: 99 },
  { name: 'Kenny', vampires: 39 },
  { name: 'Raleigh', vampires: 2 },
  { name: 'Pilot', vampires: 54 },
  { name: 'Dunx', vampires: 165 },
  { name: 'Barny' }
]
```

Рисунок 33 – Исходное состояние коллекции

Увеличиваем количество на 5:

```
learn> db.unicorns.updateMany({gender: 'm'},{$inc: {vampires: 5}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
```

Рисунок 34 – Внесение изменений

Проверяем:

```
learn> db.unicorns.find({gender: 'm'},{_id: 0, name: 1, vampires: 1}).toArray()
[
  { name: 'Horny', vampires: 68 },
  { name: 'Unicrom', vampires: 187 },
  { name: 'Roooooodles', vampires: 104 },
  { name: 'Kenny', vampires: 44 },
  { name: 'Raleigh', vampires: 7 },
  { name: 'Pilot', vampires: 59 },
  { name: 'Dunx', vampires: 170 },
  { name: 'Barny', vampires: 5 }
]
learn>
```

Рисунок 35 – Проверка

Практическое задание 8.2.10:

1. Изменить информацию о городе Портланд: мэр этого города теперь беспартийный.

2. Проверить содержимое коллекции towns.

Было:

```
learn> db.towns.find({name: 'Portland'})
[
  {
    _id: ObjectId("6468981d861899e6fc052aec"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
```

Рисунок 36 – Исходное состояние коллекции

Изменение информации о городе

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

Рисунок 37 – Внесение изменений

Стало:

```
learn> db.towns.find({name: 'Portland'})
[
  {
    _id: ObjectId("6468981d861899e6fc052aec"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
```

Практическое задание 8.2.11:

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

Было:

```
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId("64677c383b3f78129efe51af"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

Рисунок 38 – Исходное состояние коллекции

Внесение изменений:

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

Рисунок 39 – Внесение изменений

Стало:

```
learn> db.unicorns.find({name: 'Pilot'})
[
  {
    _id: ObjectId("64677c383b3f78129efe51af"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
```

Рисунок 40 – Проверка

Практическое задание 8.2.12:

1. Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны.

2. Проверить содержимое коллекции unicorns.

Было:

```
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId("64677aed3b3f78129efe51a7"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Рисунок 41 – Исходное состояние коллекции

Внесение изменений:

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

Рисунок 42 – Внесение изменений

Стало:

```
learn> db.unicorns.find({name: 'Aurora'})
[
  {
    _id: ObjectId("64677aed3b3f78129efe51a7"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

Рисунок 43 – Проверка

Практическое задание 8.2.13:

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"], 22 mayor: { name: "Sam Adams", party: "D"}}

2. Удалите документы с беспартийными мэрами.

3. Проверьте содержание коллекции.

4. Очистите коллекцию.

5. Просмотрите список доступных коллекций.

Коллекция towns уже была создана в предыдущем задании. В коллекции есть 3 документа.

```
learn> db.towns.countDocuments()
3
learn>
```

Рисунок 44 – Количество документов

Удаление документов с беспартийными мэрами

```
learn> db.towns.deleteMany({'mayor.party':{$exists: false}})
{ acknowledged: true, deletedCount: 2 }
```

Рисунок 45 - Удаление документов

Проверка

```
learn> db.towns.countDocuments()
1
learn>
```

Рисунок 46 - Проверка

Очистить коллекцию и просмотр коллекции

```
learn> db.towns.drop()
true
learn> show collections
unicorns
```

Рисунок 47 - Практическое задание 8.2.13

Практическое задание 8.3.1:

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 единорогов

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

```
learn> db.createCollection('areas')
{ ok: 1 }
learn> db.areas.insertOne({_id: 'a', name: 'area_1', description: 'forest'})
{ acknowledged: true, insertedId: 'a' }
learn> db.areas.insertOne({_id: 'b', name: 'area_2', description: 'mountains'})
{ acknowledged: true, insertedId: 'b' }
learn> db.areas.insertOne({_id: 'c', name: 'area_3', description: 'lake'})
{ acknowledged: true, insertedId: 'c' }
learn> db.areas.find()
[
  { _id: 'a', name: 'area_1', description: 'forest' },
  { _id: 'b', name: 'area_2', description: 'mountains' },
  { _id: 'c', name: 'area_3', description: 'lake' }
]
```

Рисунок 48 – Создание коллекции. Практическое задание 8.3.1

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

```
learn> db.unicorns.updateMany({name: {$in: ['Horny','Ayna', 'Unicrom']}},{$set: {area: {$ref: 'areas', $id: 'c'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: 'Solnara'},{$set: {area: {$ref: 'areas', $id: 'a'}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Рисунок 49 – Включение единорогов в документы

3. Проверьте содержание коллекции единорогов

```
learn> db.unicorns.find({},{_id: 0, name: 1, area: 1})
[
  { name: 'Horny', area: DBRef("areas", 'c') },
  { name: 'Aurora' },
  { name: 'Unicrom', area: DBRef("areas", 'c') },
  { name: 'Roooooodles' },
  { name: 'Solnara', area: DBRef("areas", 'a') },
  { name: 'Ayna', area: DBRef("areas", 'c') },
  { name: 'Kenny' },
  { name: 'Raleigh' },
  { name: 'Leia' },
  { name: 'Pilot' },
  { name: 'Nimue' },
  { name: 'Dunx' },
  { name: 'Barny' }
]
```

Рисунок 50 – Проверка. Практическое задание 8.3.1

Практическое задание 8.3.2:

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

Содержание коллекции 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'});

db.unicorns.insert {name: 'Dunx', dob: new Date(1976, 6, 18, 18, 18), loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165

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

Рисунок 51– Проверка. Практическое задание 8.3.2

Вывод: да, можно задать для коллекции unicorns индекс для ключа name с флагом unique.

Практическое задание 8.3.3:

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.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1', unique: true }
]
learn> db.unicorns.dropIndex('name_1')
{ nIndexesWas: 2, ok: 1 }
learn> db.unicorns.dropIndex('_id_')
MongoServerError: cannot drop _id index
```

Рисунок 52–Практическое задание 8.3.3

Вывод: при удалении индекса '_id_' возникнет ошибка, потому что поле _id используется для идентификации документов в коллекции.

Практическое задание 8.3.4:

1. Создайте объемную коллекцию numbers, задействовав курсор: for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}

```
learn> db.createCollection('numbers')
{ ok: 1 }
learn> for (i = 0; i < 100000; i++) {
...     db.numbers.insert({ value: i })
... }
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6468af6e861899e6fc06b18d") }
}
```

2. Выберите последние четыре документа

```
learn> db.numbers.find().sort({ _id: -1 }).limit(4)
[
  { _id: ObjectId("6468af6e861899e6fc06b18d"), value: 99999 },
  { _id: ObjectId("6468af6e861899e6fc06b18c"), value: 99998 },
  { _id: ObjectId("6468af6e861899e6fc06b18b"), value: 99997 },
  { _id: ObjectId("6468af6e861899e6fc06b18a"), value: 99996 }
]
```

3. Проанализируйте план выполнения запроса 2. Сколько потребовалось executionTimeMillis)

```
learn> db.numbers.find().sort({ _id: -1 }).limit(4).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: '51B6F510',
    planCacheKey: '51B6F510',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'LIMIT',
      limitAmount: 4,
      inputStage: {
        stage: 'FETCH',
        inputStage: {
          stage: 'IXSCAN',
          keyPattern: { _id: 1 },
          indexName: '_id_',
```

4. Создайте индекс для ключа value.
5. Получите информацию о всех индексах коллекции numbres.

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

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

```
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 0,
    totalKeysExamined: 4,
    totalDocsExamined: 4,
    executionStages: {
      stage: 'LIMIT',
      nReturned: 4,
      executionTimeMillisEstimate: 0,
      works: 5,
      advanced: 4,
      needTime: 0,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      limitAmount: 4,
      inputStage: {
        stage: 'FETCH',
        nReturned: 4,
        executionTimeMillisEstimate: 0,
        works: 4,
        advanced: 4,
        needTime: 0,
        needYield: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 0,
        docsExamined: 4,
```

Без индекса запрос выполняется за 27 миллисекунд, с индексом – за ноль. Таким образом, можно сделать вывод, что при использовании индексов значительно ускоряется выполнение запроса.

Выводы

При выполнении данной лабораторной работы были приобретены практические навыки по работе с MongoDB, а именно овладение 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.

Таким образом, был получен опыт по работе и использованию MongoDB.