ЛАБОРАТОРНАЯ РАБОТА 5.2

РАБОТА С БД В СУБД MONGODB

Выполнил: РЫБАЛКО ОЛЕГ К32392

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:

1. Вставка

1.1 Вставка документов в коллекцию

Для начала нам необходимо создать базу данных при помощи команды *use learn.* Затем добавляем документы в коллекцию *unicorns* при помощи функций, описанных в задании. После этого создадим документ и добавим его в базу *unicorns* при помощи двух команд *document=…, db.unicorns.insertOne(document)*

```
test> use learn
switched to db learn
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("647766306ef6ff802fa4a878") }
}
learn> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647766306ef6ff802fa4a879") }
}
learn> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647766306ef6ff802fa4a87a") }
}
learn> db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647766306ef6ff802fa4a87b") }
}
learn> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647766306ef6ff802fa4a87c") }
}
```

```
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.insertOne(document)
{
  acknowledged: true,
  insertedId: ObjectId("647766796ef6ff802fa4a883")
}
```

```
learn> db.unicorns.find({name: {$eq: 'Dunx'}})
[
  {
    _id: ObjectId("647766796ef6ff802fa4a883"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
]
learn>
```

1.2. Выборка данных из БД

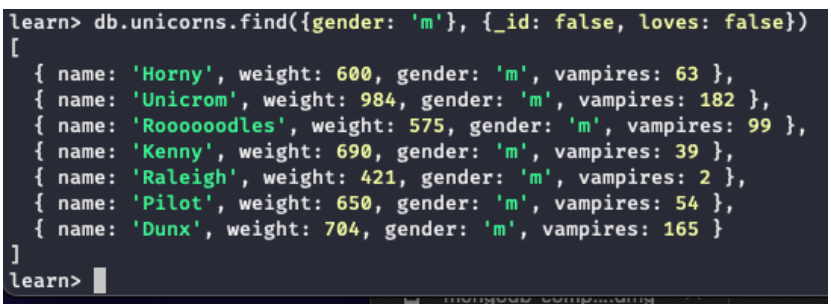
Сформируем запрос для вывода списка самцов и самок единорогов. Для этого воспользуемся функцией *find* и также отсортируем документы по полю имени и выведем первые 3 документа. Для самцов поле *gender = ‘m’,* а для самок – ‘f’.

```
learn> db.unicorns.find({gender: 'm'}).sort({name: 1}).limit(3)
[
  {
    _id: ObjectId("647766796ef6ff802fa4a883"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a878"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87e"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
```

Для нахождения самеи единорога, которая любит морковь, воспользуемся функцией *findOne*

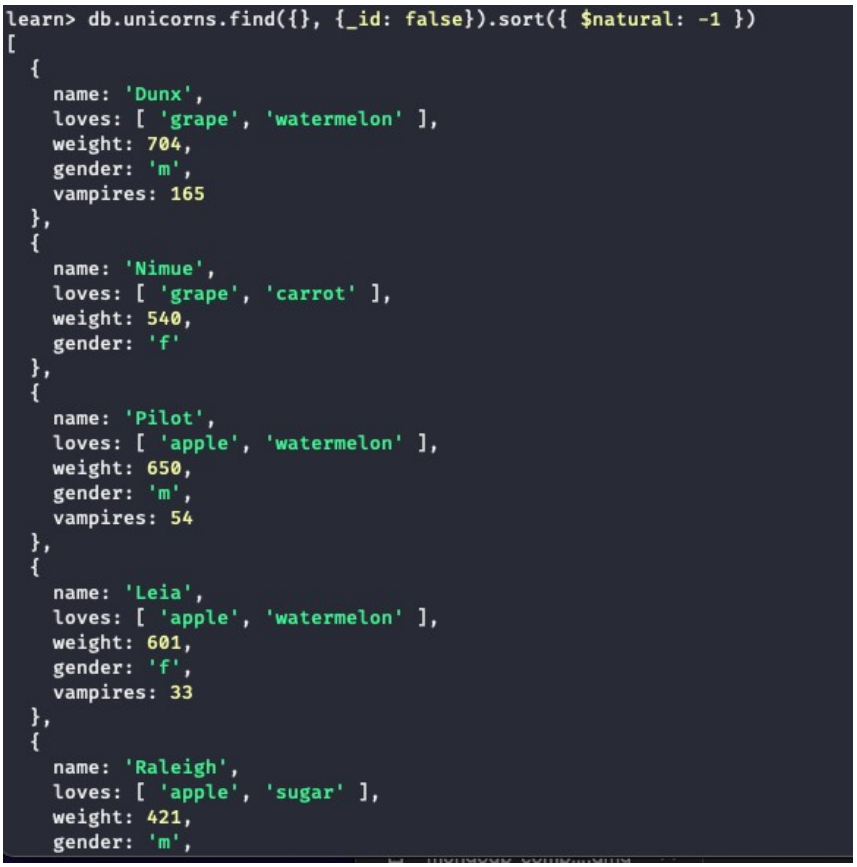
1.3. Модификация запроса

Для вывода списка самцов единорогов, исключив из результата информацию о предпочтениях и поле *_id* воспользуемся функцией *find*



1.4. Вывод списка в обратном порядке добавления

Для вывода списка документов в обратном порядке добавления воспользуемся параметром *$natural.* Также я исключил поле *_id* для более удобного чтения



1.5. Вывод списка любимых предпочтений

Для вывода списка единорогов с их любимого предпочетния, воспользуемся *$slice*

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

1.6. Вывод документов, отфильтровав их при помощи логических операций

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

```
learn> db.unicorns.find({ gender: 'f', weight: { $gte: 500, $lte: 700 } }, { _id: false })
[
  {
    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'
  }
]
```

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

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

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({ vampires: { $exists: false } })
[
  {
    _id: ObjectId("647766336ef6ff802fa4a882"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

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

```
learn> db.unicorns.find({ gender: 'm' }, {name: true, loves: { $slice: 1 }, _id: false}).sort({ name: 1 })
[
  { name: 'Dunx', loves: [ 'grape' ] },
  { name: 'Horny', loves: [ 'carrot' ] },
  { name: 'Kenny', loves: [ 'grape' ] },
  { name: 'Pilot', loves: [ 'apple' ] },
  { name: 'Raleigh', loves: [ 'apple' ] },
  { name: 'Roooooodles', loves: [ 'apple' ] },
  { name: 'Unicrom', loves: [ 'energon' ] }
]
learn>
```

2. Запрос к вложенным объектам

2.1 Создание новой коллекции

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.*

```
learn> db.towns.insert({name: "Punxsutawney ", populatiuon: 6200, last_sensus: ISODate("2008-01-31"), famous_for:
 [""], mayor: {name: "Jim Wehrle" }})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647771a16ef6ff802fa4a884") }
}
learn> db.towns.insert({name: "New York", populatiuon: 22200000, last_sensus: ISODate("2009-07-31"), famous_for:
["status of liberty", "food"], mayor: {name: "Michael Bloomberg", party: "I"}})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("647771bd6ef6ff802fa4a885") }
}
learn> {name: "Portland", populatiuon: 528000, last_sensus: ISODate("2009-07-20"), famous_for: ["beer", "food"],
mayor: {name: "Sam Adams", party: "D"}}
...
learn> db.towns.insert({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("647771e46ef6ff802fa4a886") }
}
learn>
```

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

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

2.2 Функции и курсоры

Сформировать функцию для вывода списка самцов единорогов.

```
  },

learn> fn = function() { return this.gender === "m"; }
[Function: fn]
learn> fn
[Function: fn]
```

```
learn> db.unicorns.find({ $where: fn })
[
  {
    _id: ObjectId("647766306ef6ff802fa4a878"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87a"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87b"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87e"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
```

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

Вывести результат, используя forEach

```
learn> var cursor = db.unicorns.find({ $where: fn }).limit(2).sort({ name: 1 });null;
null
learn> cursor.forEach(function(obj) {print(obj.name)})
Dunx
Horny

learn>
```

3. Агрегированные запросы

Вывести количество самок единорогов весом от полутонны до 600 кг.

Вывести список предпочтений.

Посчитать количество особей единорогов обоих полов.

4. Редактирование данных

Функция save была убрана начиная с версии 4.2
https://www.mongodb.com/docs/v4.4/reference/method/db.collection.save/

```
learn> db.unicorns.insertOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
{
  acknowledged: true,
  insertedId: ObjectId("6477a44c6ef6ff802fa4a887")
}
learn>
```

```
  },
  {
    _id: ObjectId("647766336ef6ff802fa4a882"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  },
  {
    _id: ObjectId("647766796ef6ff802fa4a883"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  },
  {
    _id: ObjectId("6477a44c6ef6ff802fa4a887"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm'
  }
]
learn>
```

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

Проверить содержимое коллекции `unicorns`.

```
learn> db.unicorns.updateOne({name: "Ayna"}, {$set: {weight: 800, vampires: 51}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Ayna"})
[
  {
    _id: ObjectId("647766306ef6ff802fa4a87d"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51
  }
]
learn>
```

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

```
learn> db.unicorns.updateOne({name: "Raleigh", gender: "m"}, { $push: { loves: "redbull" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Raleigh"})
[
  {
    _id: ObjectId("647766336ef6ff802fa4a87f"),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
]
learn>
```

Всем самцам единорогов увеличить количество убитых вапмиров на 5.

До:

```
learn> db.unicorns.find({gender: "m"}).limit(3)
[
  {
    _id: ObjectId("647766306ef6ff802fa4a878"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87a"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87b"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  }
]
learn>
```

После:

```
learn> db.unicorns.updateMany({ gender: "m" }, { $inc: { vampires: 5 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 8,
  modifiedCount: 8,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: "m"}).limit(3)
[
  {
    _id: ObjectId("647766306ef6ff802fa4a878"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87a"),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 187
  },
  {
    _id: ObjectId("647766306ef6ff802fa4a87b"),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 104
  }
]
learn>
```

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

```
learn> db.towns.updateOne({ name: "Portland" }, { $unset: { "mayor.party": 1 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find({name: "Portland"})
[
  {
    _id: ObjectId("647771e46ef6ff802fa4a886"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
learn>
```

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

```
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId("647766336ef6ff802fa4a881"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn> db.unicorns.updateOne({name: "Pilot"}, { $push: { loves: "chocolate" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId("647766336ef6ff802fa4a881"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 59
  }
]
learn>
```

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

```
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId("647766306ef6ff802fa4a879"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', '' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn> db.unicorns.updateOne({name: "Aurora"}, { $addToSet: { loves: {$each: ["sugar", "lemons"]} } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId("647766306ef6ff802fa4a879"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', '', 'sugar', 'lemons' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

5. Удаление данных из коллекции

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

```
[
  {
    _id: ObjectId("6477ab8a6ef6ff802fa4a889"),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId("6477ab986ef6ff802fa4a88a"),
    name: 'Punxsutawney',
    popujatiuon: 6200,
    last_sensus: ISODate("2008-01-31T00:00:00.000Z"),
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId("6477abae6ef6ff802fa4a88b"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn> db.towns.deleteMany({"mayor.party": { $exists: false }})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()
[
  {
    _id: ObjectId("6477ab8a6ef6ff802fa4a889"),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: ISODate("2009-07-31T00:00:00.000Z"),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId("6477abae6ef6ff802fa4a88b"),
    name: 'Portland',
    popujatiuon: 528000,
    last_sensus: ISODate("2009-07-20T00:00:00.000Z"),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

```
learn> db.towns.deleteMany({})
{ acknowledged: true, deletedCount: 2 }
learn>
```

```
learn> db.getCollectionNames()
[ 'unicorns', 'towns' ]
learn>
```

6. Ссылки в БД

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

```
learn> db.regions.insert({ _id: "us-west-1", name: "California", description: "Here influencer unicorns leave" })
{ acknowledged: true, insertedIds: { '0': 'us-west-1' } }
learn> db.regions.find()
[
  {
    _id: 'us-west-1',
    name: 'California',
    description: 'Here influencer unicorns leave'
  }
]
learn> db.regions.insert({ _id: "us-east-1", name: "Miami", description: "Here is the place of hurricane unicorns" })
{ acknowledged: true, insertedIds: { '0': 'us-east-1' } }
learn> db.regions.insert({ _id: "eu-central-1", name: "Frankfurt", description: "Beer unicorns live here" })
{ acknowledged: true, insertedIds: { '0': 'eu-central-1' } }
learn>
```

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

```
learn> db.unicorns.updateOne({name: "Dunx"}, {$set: {region: {$ref: "regions", $id: "us-east-1"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: "Barny"}, {$set: {region: {$ref: "regions", $id: "us-west-1"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.updateOne({name: "Nimue"}, {$set: {region: {$ref: "regions", $id: "eu-central-1"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn>
```

```
  {
    _id: ObjectId("647766336ef6ff802fa4a882"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f',
    region: DBRef("regions", 'eu-central-1')
  },
  {
    _id: ObjectId("647766796ef6ff802fa4a883"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 170,
    region: DBRef("regions", 'us-east-1')
  },
  {
    _id: ObjectId("6477a44c6ef6ff802fa4a887"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm',
    vampires: 5,
    region: DBRef("regions", 'us-west-1')
  }
]
learn>
```

7. Настройка индексов

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

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

8. Управление индексами

Получите информацию о всех индексах коллекции `unicorns` .

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

Удалите все индексы, кроме индекса для идентификатора.

```
learn> db.unicorns.dropIndex("name_1")
{ nIndexesWas: 2, ok: 1 }
learn>
```

Попытайтесь удалить индекс для идентификатора.

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

9. План запроса

Создайте объемную коллекцию `numbers`*, задействовав курсор:*

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

```
learn> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6477b2bf6ef6ff802fa66237") }
}
learn>
```

Выберите последних четыре документа.

```
learn> db.numbers.find().sort({value: -1}).limit(4)
[
  { _id: ObjectId("6477b2bf6ef6ff802fa66237"), value: 99999 },
  { _id: ObjectId("6477b2bf6ef6ff802fa66236"), value: 99998 },
  { _id: ObjectId("6477b2bf6ef6ff802fa66235"), value: 99997 },
  { _id: ObjectId("6477b2bf6ef6ff802fa66234"), value: 99996 }
]
learn>
```

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

76 milliseconds

Создайте индекс для ключа `value`.

Получите информацию о всех индексах коллекции `numbres`.

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

Выполните запрос 2.

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

0 milliseconds

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

Запрос с использованием индекса получился более эффективным

Вывод:

В ходе выполнения данной лабораторной работы удалось познакомиться с NoSQL базой данных MongoDB, утилитой командной строки mongosh и научиться работать с данной базой.