Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ № 5

по теме: Работа с БД в СУБД MongoDB
по дисциплине: Проектирование и реализация баз данных

Специальность:
09.03.03 Мобильные и сетевые технологии

Проверил:
Говорова М.М. _____
Дата: «__» _____ 20__г.
Оценка _____

Выполнил:
студент группы K3240
Козлов И.Д.

Санкт-Петербург 2022

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

Оборудование: компьютерный класс.
Программное обеспечение: СУБД MongoDB 5.0.8., консоль MongoDB compass

ХОД РАБОТЫ

1) *Создайте базу данных learn.*

```
> use learn
< 'switched to db learn'
```

2) *Заполните коллекцию единорогов unicorns:*

```
> db.unicorns.insert({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
  db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
  db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
  db.unicorns.insert({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
  db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
  db.unicorns.insert({name:'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
  db.unicorns.insert({name:'Kenny', loves: ['grape', 'lemon'], weight: 690,  gender: 'm', vampires: 39});
  db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
  db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
  db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
  db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
< 'DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.'
```

3) Проверка коллекции:

```
> show collections
< unicorns
```

4) *Используя второй способ, вставьте в коллекцию единорогов документ:*

```
> doc = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
< {
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
> db.unicorns.insert(doc)
< { acknowledged: true,
    insertedIds: { '0': ObjectId("628b6d610dacdc0b105abac5") } }
```

5) *Проверьте содержимое коллекции с помощью метода find*:

```
> db.unicorns.find({name: 'Dunx'})
< { _id: ObjectId("628b6d610dacdc0b105abac5"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165 }
```

6) *Сформируйте запросы для вывода списков самцов и самок единорогов. Ограничьте список самок первыми тремя особями. Отсортируйте списки по имени.*

```
> db.unicorns.find({gender: 'm'}).sort({name: 1})
< { _id: ObjectId("628b6d610dacdc0b105abac5"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165 }
  { _id: ObjectId("628b6b570dacdc0b105abab8"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63 }
  { _id: ObjectId("628b6b580dacdc0b105ababe"),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39 }
> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
< { _id: ObjectId("628b6b570dacdc0b105abab9"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43 }
  { _id: ObjectId("628b6b580dacdc0b105ababd"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40 }
  { _id: ObjectId("628b6b580dacdc0b105abac0"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33 }
```

7) *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("628b6b570dacdc0b105abab9"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43 }
> db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
< { _id: ObjectId("628b6b570dacdc0b105abab9"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43 }
```

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

```
> db.unicorns.find({gender: 'm'}, {loves: 0, gender: 0}).sort({name: 1})
< { _id: ObjectId("628b6d610dacdc0b105abac5"),
    name: 'Dunx',
    weight: 704,
    vampires: 165 }
  { _id: ObjectId("628b6b570dacdc0b105abab8"),
    name: 'Horny',
    weight: 600,
    vampires: 63 }
```

9) *Вывести список единорогов в обратном порядке добавления.*

```
> db.unicorns.find().sort({ $natural: -1 })
< { _id: ObjectId("628b6d610dacdc0b105abac5"),
    name: 'Dunx',
    loves: [ 'grape', 'watermelon' ],
    weight: 704,
    gender: 'm',
    vampires: 165 }
  { _id: ObjectId("628b6b580dacdc0b105abac2"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f' }
  { _id: ObjectId("628b6b580dacdc0b105abac1"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54 }
  { _id: ObjectId("628b6b580dacdc0b105abac0"),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33 }
```

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

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

11) *Вывести список самок единорогов весом от полтонны до 700 кг, исключив вывод идентификатора.*

```
> db.unicorns.find({weight: {$gt : 500, $lt : 700}, gender: 'f' },{_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' }
```

12) *Вывести список самцов единорогов весом от полтонны и предпочитающих grape и lemon, исключив вывод идентификатора.*

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

13) *Найти всех единорогов, не имеющих ключ vampires*

```
> db.unicorns.find({vampires: {$exists:false}})
< { _id: ObjectId("628b6b580dacdc0b105abac2"),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f' }
```

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

```
> db.unicorns.find({gender: 'm'},{loves: {$slice : 1} }).sort({name: 1})
< { _id: ObjectId("628b6d610dacdc0b105abac5"),
    name: 'Dunx',
    loves: [ 'grape' ],
    weight: 704,
    gender: 'm',
    vampires: 165 }
  { _id: ObjectId("628b6b570dacdc0b105abab8"),
    name: 'Horny',
    loves: [ 'carrot' ],
    weight: 600,
    gender: 'm',
    vampires: 63 }
  { _id: ObjectId("628b6b580dacdc0b105ababe"),
    name: 'Kenny',
    loves: [ 'grape' ],
    weight: 690,
    gender: 'm',
    vampires: 39 }
```

15) Создайте коллекцию towns, включающую следующие документы:

```
> db.towns.insert({name: "Punxsutawney ",
  populatiuon: 6200,
  last_sensus: ISODate("2008-01-31"),
  famous_for: [""],
  mayor: {
     name: "Jim Wehrle"
  }}
  )
< { acknowledged: true,
    insertedIds: { '0': ObjectId("628cdf9c0dacdc0b105abac6") } }
> 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("628cdfc20dacdc0b105abac7") } }
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("628cdff20dacdc0b105abac8") } }
learn>
```

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

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

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

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

18) *Сформировать функцию для вывода списка самцов единорогов*

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

```
> db.unicorns.find(fn)
{ "_id" : ObjectId("628b6b570dacdc0b105abab8"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("628b6b570dacdc0b105ababa"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("628b6b570dacdc0b105ababb"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("628b6b580dacdc0b105ababe"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("628b6b580dacdc0b105ababf"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("628b6b580dacdc0b105abac1"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("628b6d610dacdc0b105abac5"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
>
```

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

```
> var cursor = db.unicorns.find({gender:'m'}).sort({name:1}).limit(2);null;
< null
```

20) *Вывести результат, используя forEach.*

```
> cursor.forEach(function(fn){print(fn.name)})
< 'Dunx'
< 'Horny'
```

21) *Вывести количество самок единорогов весом от полутонны до 600 кг.*

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

22) *Вывести список предпочтений.*

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

23) *Посчитать количество особей единорогов обоих полов.*

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

24) Выполнить команду:

```
> db.unicorns.save({name: 'Barny', loves: ['grape'],
... weight: 340, gender: 'm'})
WriteResult({ "nInserted" : 1 })
```

25) *Проверить содержимое коллекции unicorns*

```
> db.unicorns.find({name: 'Barny'})
< { _id: ObjectId("628cee242d98888bf7838c36"),
    name: 'Barny',
    loves: [ 'grape' ],
    weight: 340,
    gender: 'm' }
```

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

```
> db.unicorns.update({name: 'Ayna', gender: 'f'},{weight: 800, vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

27) *Проверить содержимое коллекции unicorns.*

```
> db.unicorns.find({name: 'Ayna'})
< { _id: ObjectId("628cf09f0dacdc0b105abacb"),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 800,
    gender: 'f',
    vampires: 51 }
```

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

```
> db.unicorns.update({name: 'Raleigh'},{$set:{loves:'рэдбул'} },{multi:true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
> db.unicorns.find({name: 'Raleigh'})
< { _id: ObjectId("628b6b580dacdc0b105ababf"),
    name: 'Raleigh',
    loves: 'рэдбул',
    weight: 421,
    gender: 'm',
    vampires: 2 }
```

29) *Всем самцам единорогов увеличить количество убитых вапмиров на 5*
Проверить содержимое коллекции `unicorns`.

```
> db.unicorns.update({gender: 'm'},{$inc: {vampires:5}},{multi:true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 8,
    modifiedCount: 8,
    upsertedCount: 0 }
> db.unicorns.find({gender: 'm'})
< { _id: ObjectId("628b6b570dacdc0b105abab8"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68 }
```

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

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

```
> db.towns.find().skip(2).limit(3)
< { _id: ObjectId("628cdff20dacdc0b105abac8"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2009-07-20T00:00:00.000Z,
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' } }
```

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

```
> db.unicorns.update({name: 'Pilot'}, {$push: {loves: "шоколад"}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
> db.unicorns.find({name: 'Pilot'})
< { _id: ObjectId("628b6b580dacdc0b105abac1"),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'шоколад' ],
    weight: 650,
    gender: 'm',
    vampires: 59 }
```

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

```
> db.unicorns.update({name: 'Aurora'}, {$addToSet: {loves: {$each: ["сахар", "лимоны"]}}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
> db.unicorns.find({name: 'Aurora'})
< { _id: ObjectId("628b6b570dacdc0b105abab9"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'сахар', 'лимоны' ],
    weight: 450,
    gender: 'f',
    vampires: 43 }
```

33) *Удалите документы с беспартийными мэрами*
Проверьте содержание коллекции.

```
> db.towns.remove({"mayor.party": {$exists: false}})
< 'DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.'
< { acknowledged: true, deletedCount: 3 }
> db.towns.find({"mayor.party": {$exists: false}})
<
```

34) *Очистите коллекцию.*
Просмотрите список доступных коллекций.

```
> db.towns.remove({})
< { acknowledged: true, deletedCount: 3 }
> db.towns.find()
<
```

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

```
> db.zones.insert({"_id" : "IRK", "name" : "Irkutsk", "discription" : "steal water"})
< { acknowledged: true, insertedIds: { '0': 'IRK' } }
> db.zones.insert({"_id" : "SPB", "name" : "Saint-Petersburg", "discription" : "swamp"})
< { acknowledged: true, insertedIds: { '0': 'SPB' } }
```

36) *Включите для нескольких единорогов в документы ссылку на зону обитания, использую второй способ автоматического связывания.*
Проверьте содержание коллекции единорогов.

```
> db.unicorns.update({_id:ObjectId("628b6b570dacdc0b105abab8")},{$set: {zone:{$ref:"zones", $id: "IRK"}}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
> db.unicorns.find({_id:ObjectId("628b6b570dacdc0b105abab8")})
< { _id: ObjectId("628b6b570dacdc0b105abab8"),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    zone: DBRef("zones", 'IRK') }
```

```
> db.unicorns.update({_id:ObjectId("628b6b570dacdc0b105abab9")},{$set: {zone:{$ref:"zones", $id: "SPB"}}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
> db.unicorns.find({_id:ObjectId("628b6b570dacdc0b105abab9")})
< { _id: ObjectId("628b6b570dacdc0b105abab9"),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'сахар', 'лимоны' ],
    weight: 450,
    gender: 'f',
    vampires: 43,
    zone: DBRef("zones", 'SPB') }
```

37) *Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом* unique. - НЕЛЬЗЯ

```
> db.unicorns.ensureIndex({"name": 1}, {"unique": true})
✖ ▸ MongoServerError: E11000 duplicate key error collection: learn.unicorns index: name_1 dup key: { : null }
```

38) *Получите информацию о всех индексах коллекции unicorns*

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

39) *Удалите все индексы, кроме индекса для идентификатора.*

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

40) *Попытайтесь удалить индекс для идентификатора.*

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

41) *Создайте объемную коллекцию numbers, задействовав курсор:*
for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
Выберите последних четыре документа.

```
> db.numbers.find().skip(99996).limit(4)
< { _id: ObjectId("628d031d2d98888bf78512d3"), value: 99996 }
  { _id: ObjectId("628d031d2d98888bf78512d4"), value: 99997 }
  { _id: ObjectId("628d031d2d98888bf78512d5"), value: 99998 }
  { _id: ObjectId("628d031d2d98888bf78512d6"), value: 99999 }
learn>
```

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

```
> db.numbers.explain("executionStats").find().skip(99996).limit(4)
{ executionSuccess: true,
  nReturned: 4,
  executionTimeMillis: 97,
```

43) *Создайте индекс для ключа value.*

```
> db.numbers.createIndex({"value" : 1})
< 'value_1'
```

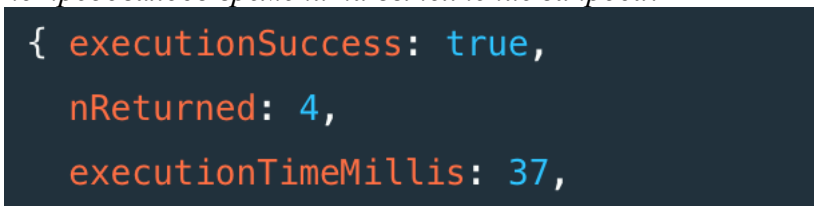
44) *Получите информацию о всех индексах коллекции numbres.*

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

45) *Выберите последних четыре документа*

```
> db.numbers.find().skip(99996).limit(4)
< { _id: ObjectId("628d031d2d98888bf78512d3"), value: 99996 }
  { _id: ObjectId("628d031d2d98888bf78512d4"), value: 99997 }
  { _id: ObjectId("628d031d2d98888bf78512d5"), value: 99998 }
  { _id: ObjectId("628d031d2d98888bf78512d6"), value: 99999 }
```

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



Потребовалось 37 миллисекунд, до создания индекса – 97 миллисекунд. Очевидно, что с индексами запрос работает быстрее.

ВЫВОД

Научился работать с CRUD-операциями в MongoDB.