Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

«Работа с БД в СУБД MongoDB»

по дисциплине «Проектирование и реализация баз данных»

Обучающийся Поляков Александр Владимирович

Факультет прикладной информатики

Группа K3239

Направление подготовки 09.03.03 Прикладная информатика

Образовательная программа Мобильные и сетевые технологии 2023

Преподаватель Говорова Марина Михайловна

Санкт-Петербург

2024/2025

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+, 8.0.4 (последняя).

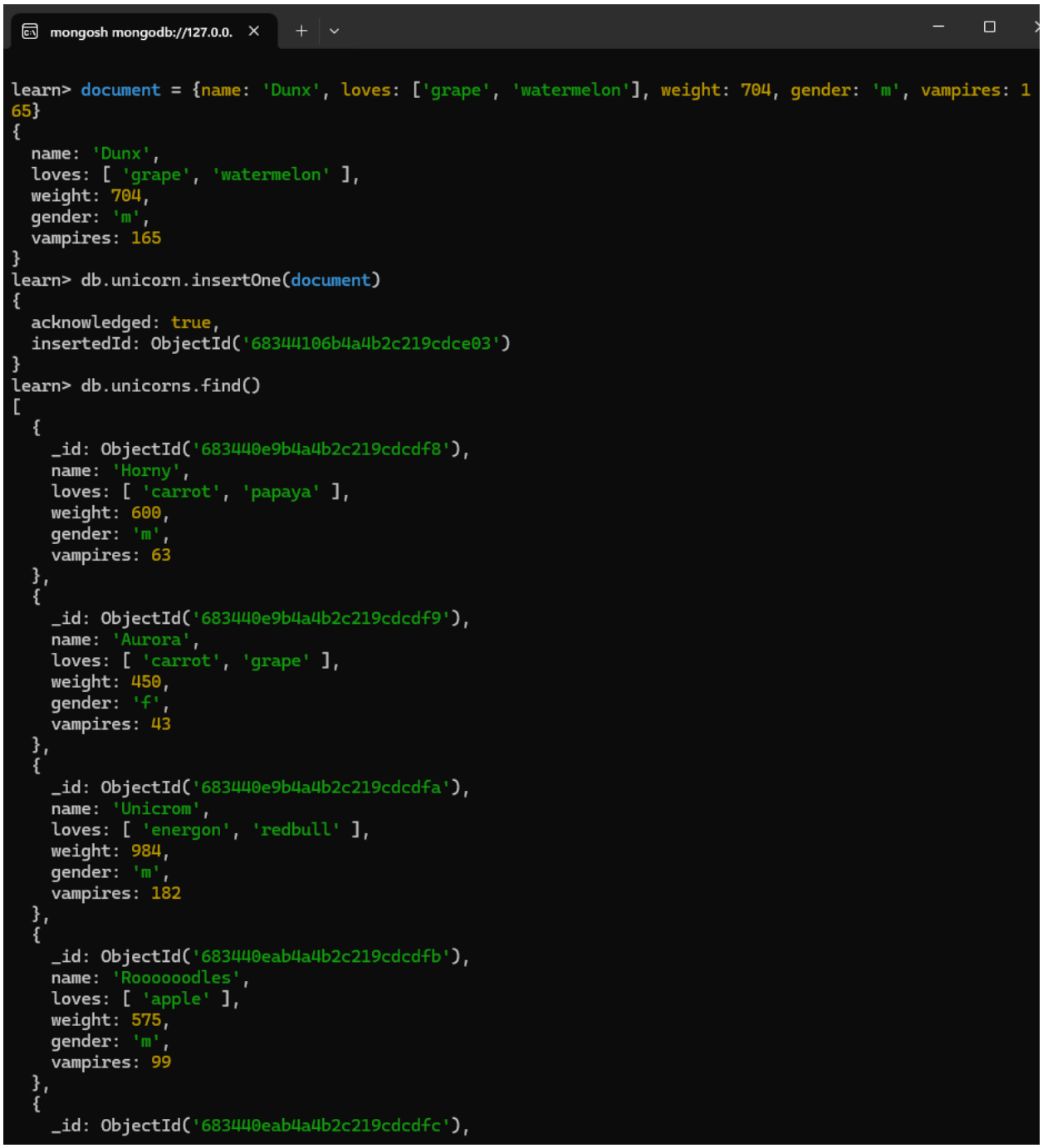
Практическое задание

Задание 2.1.1

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

3. Используя второй способ, вставьте в коллекцию единорогов документ

4. Проверьте содержимое коллекции с помощью метода find.



Задание 2.2.1

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": "f"}).sort({"name": 1}).limit(3)
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf9'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfd'),
    name: 'Ayna',
    loves: [ 'strawberry', 'lemon' ],
    weight: 733,
    gender: 'f',
    vampires: 40
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdce00'),
    name: 'Leia',
    loves: [ 'apple', 'watermelon' ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
]
learn>
```

```
learn> db.unicorns.find({"gender": "m"}).sort({"name": 1})
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf8'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfe'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdce01'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdff'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfb'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdfa'),
```

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

Задание 2.2.2

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

```
learn> db.unicorns.find({"gender": "m"}, {"loves": 0, "gender": 0})
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf8'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  },
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdfa'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfb'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfe'),
    name: 'Kenny',
    weight: 690,
    vampires: 39
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdff'),
    name: 'Raleigh',
    weight: 421,
    vampires: 2
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdce01'),
    name: 'Pilot',
    weight: 650,
    vampires: 54
  }
]
learn>
```

Задание 2.2.3

Вывести список единорогов в обратном порядке добавления.

Задание 2.2.4

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

Задание 2.3.1

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: 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>
```

Задание 2.3.2

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: ObjectId('683440eab4a4b2c219cdcdfe'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
]
learn>
```

Задание 2.3.3

Найти всех единорогов, не имеющих ключ vampires.

```
learn> db.unicorns.find({"vampires": {$exists:false}})
[
  {
    _id: ObjectId('683440eab4a4b2c219cdce02'),
    name: 'Nimue',
    loves: [ 'grape', 'carrot' ],
    weight: 540,
    gender: 'f'
  }
]
learn>
```

Задание 3.1.1

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

```
learn> db.towns.find({"mayor.party": "I"}, {"name": 1, "mayor": 1})
[
  {
    _id: ObjectId('683449fbb4a4b2c219cdce06'),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  }
]
learn>
```

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}}, {"name": 1, "mayor": 1})
[
  {
    _id: ObjectId('683449fbb4a4b2c219cdce05'),
    name: 'Punxsutawney',
    mayor: { name: 'Jim Wehrle' }
  }
]
learn>
```

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.

```
learn> var cursor0 = db.unicorns.find({"gender": "m"}); null;
null
learn> cursor0.sort({name:1}).limit(2);null;
null
learn> cursor0.forEach(function(obj) {print(obj.name)})
Dunx
Horny

learn>
```

Задание 3.2.1

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
learn> db.unicorns.find({"gender": "f", "weight": {$gte: 500, $lte: 600}}).count()
2
learn>
```

Задание 3.2.2

Вывести список предпочтений.

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

Задание 3.2.3

Посчитать количество особей единорогов обоих полов.

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

Задание 3.3.1

Функция save - deprecated

Задание 3.3.2

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

Задание 3.3.3

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

Задание 3.3.4

Всем самцам единорогов увеличить количество убитых вапмиров на 5.

```
learn> db.unicorns.find({gender: "m"})
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf8'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63
  },
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdfa'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfb'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfe'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdff'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
    weight: 421,
    gender: 'm',
    vampires: 2
  },
  {
```

```
learn> db.unicorns.updateOne( { gender: "m" }, { $inc: { vampires: 5 } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({gender: "m"})
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf8'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68
  },
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdfa'),
    name: 'Unicrom',
    loves: [ 'energon', 'redbull' ],
    weight: 984,
    gender: 'm',
    vampires: 182
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfb'),
    name: 'Roooooodles',
    loves: [ 'apple' ],
    weight: 575,
    gender: 'm',
    vampires: 99
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdfe'),
    name: 'Kenny',
    loves: [ 'grape', 'lemon' ],
    weight: 690,
    gender: 'm',
    vampires: 39
  },
  {
    _id: ObjectId('683440eab4a4b2c219cdcdff'),
    name: 'Raleigh',
    loves: [ 'apple', 'sugar', 'redbull' ],
```

Задание 3.3.5

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

```
learn> db.towns.updateMany( {name: "Portland"}, { $unset: { "mayor.party": "" } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.towns.find()
[
  {
    _id: ObjectId('683449fbb4a4b2c219cdce05'),
    name: 'Punxsutawney',
    populatiuon: 6200,
    last_sensus: ISODate('2008-01-31T00:00:00.000Z'),
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' }
  },
  {
    _id: ObjectId('683449fbb4a4b2c219cdce06'),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('683449fbb4a4b2c219cdce07'),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams' }
  }
]
learn>
```

Задание 3.3.6

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

```
learn> db.unicorns.updateOne( { name: "Pilot" }, { $addToSet: { loves: "chocolate" } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Pilot"})
[
  {
    _id: ObjectId('683440eab4a4b2c219cdce01'),
    name: 'Pilot',
    loves: [ 'apple', 'watermelon', 'chocolate' ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
]
learn>
```

Задание 3.3.7

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

```
learn> db.unicorns.updateOne( { name: "Aurora" }, { $addToSet: { loves: {$each: ["sugar", "lemon"]}} }
 )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find({name: "Aurora"})
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf9'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
learn>
```

Задание 3.4.1

1. Удалите документы с беспартийными мэрами.
2. Проверьте содержание коллекции.
3. Очистите коллекцию.
4. Просмотрите список доступных коллекций.

```
learn> db.towns.deleteMany({"mayor.party": {$exists: false}})
{ acknowledged: true, deletedCount: 1 }
learn> db.towns.find()
[
  {
    _id: ObjectId('6834558ab4a4b2c219cdce09'),
    name: 'New York',
    population: 22200000,
    last_sensus: ISODate('2009-07-31T00:00:00.000Z'),
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' }
  },
  {
    _id: ObjectId('6834558ab4a4b2c219cdce0a'),
    name: 'Portland',
    population: 528000,
    last_sensus: ISODate('2009-07-20T00:00:00.000Z'),
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: 'D' }
  }
]
learn>
```

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

learn> show dbs
admin    40.00 KiB
config  108.00 KiB
learn   184.00 KiB
local    40.00 KiB
test     40.00 KiB
learn>
```

Задание 4.1.1

1. Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание.
2. Включите для нескольких единорогов в документы ссылку на зону обитания, использую второй способ автоматического связывания.
3. Проверьте содержание коллекции едиорогов

```
learn> db.unicorns.updateOne({"name": "Horny"}, {$set: {habitat: {$ref: "habitats", $id: "crystal_vall
ey"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
learn> db.unicorns.find().limit(2)
[
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf8'),
    name: 'Horny',
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68,
    habitat: DBRef('habitats', 'crystal_valley')
  },
  {
    _id: ObjectId('683440e9b4a4b2c219cdcdf9'),
    name: 'Aurora',
    loves: [ 'carrot', 'grape', 'sugar', 'lemon' ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
]
```

```
learn> horny.habitat.toJSON().$id
crystal_valley
learn>
```

Задание 4.2.1

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>
```

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.

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

Задание 4.4.1

1. Создайте объемную коллекцию `numbers`, задействовав курсор:

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

2. Выберите последних четыре документа.
3. Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса? (по значению параметра `executionTimeMillis`)
4. Создайте индекс для ключа `value`.
5. Получите информацию о всех индексах коллекции `numbres`.
6. Выполните запрос 2.
7. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?
8. Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

```
learn> for (let i = 0; i < 100000; i++) {
...   db.numbers.insert({ value: i })
... }
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('68345bb2b4a4b2c219cf54aa') }
}
```

```
learn> db.numbers.find().sort({ value: -1 }).limit(4)
[
  { _id: ObjectId('68345bb2b4a4b2c219cf54aa'), value: 99999 },
  { _id: ObjectId('68345bb2b4a4b2c219cf54a9'), value: 99998 },
  { _id: ObjectId('68345bb2b4a4b2c219cf54a8'), value: 99997 },
  { _id: ObjectId('68345bb2b4a4b2c219cf54a7'), value: 99996 }
]
learn> db.numbers.find().sort({ value: -1 }).limit(4).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'learn.numbers',
    parsedQuery: {},
    indexFilterSet: false,
    queryHash: 'BA27D965',
    planCacheShapeHash: 'BA27D965',
    planCacheKey: '7A892B81',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'SORT',
      sortPattern: { value: -1 },
      memLimit: 104857600,
      limitAmount: 4,
      type: 'simple',
      inputStage: { stage: 'COLLSCAN', direction: 'forward' }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 4,
    executionTimeMillis: 43,
    totalKeysExamined: 0,
    totalDocsExamined: 100000,
    executionStages: {
      isCached: false,
      stage: 'SORT',
      nReturned: 4,
      executionTimeMillisEstimate: 43,
      works: 100006,
```

Видно заметное изменения времени выполнения запроса

Выводы:

NoSQL хранилища более гибкие, чем SQL, меньшая строгость к данным приводит к тому, что появляется больше свободы действий. Но, с другой стороны, это добавляет проблем при необходимых требованиях, приходится делать надстройки над процессом передачи данных, например, DTO.

NoSQL хранилища – хороший выбор для конкретных задач, но выбор стоит делать с умом.