ЛАБОРАТОРНАЯ РАБОТА 6.2

Работа с БД в СУБД MongoDB

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).

Выполнил: Мищенко Максим K3239

Практическое задание:

1. Установите MongoDB для обеих типов систем (32/64 бита).
2. Проверьте работоспособность системы запуском клиента mongo.
3. Выполните методы:
    a) db.help()
    b) db.help
    c) db.stats()
4. Создайте БД learn.
5. Получите список доступных БД.
6. Создайте коллекцию unicorns, вставив в нее документ `{name: 'Aurora', gender: 'f', weight: 450}`.
7. Просмотрите список текущих коллекций.
8. Переименуйте коллекцию unicorns.
9. Просмотрите статистику коллекции.
10. Удалите коллекцию.
11. Удалите БД learn.

Выполним шаги по заданию:

<u>Практическое задание 2.2.1:</u>

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

```
> db.unicorns.find({gender: 'f'}).limit(3).sort({name:
< {
    _id: ObjectId('6825d8864a71724db8f4aa54'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa58'),
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa5b'),
    name: 'Leia',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 601,
    gender: 'f',
    vampires: 33
  }
learn >
```

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

Практическое задание 2.2.2:

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

```
> db.unicorns.find({gender: 'm'}, {'gender': 0, loves: 0})
< {
    _id: ObjectId('6825d8864a71724db8f4aa53'),
    name: 'Horny',
    weight: 600,
    vampires: 63
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa55'),
    name: 'Unicrom',
    weight: 984,
    vampires: 182
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa56'),
    name: 'Roooooodles',
    weight: 575,
    vampires: 99
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa59'),
    name: 'Kenny',
    weight: 690,
```

Практическое задание 2.2.3:

Вывести список единорогов в обратном порядке добавления.

```
> db.unicorns.find().sort({$natural: -1})
< {
    _id: ObjectId('6825d8c04a71724db8f4aa5e'),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa5d'),
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
    gender: 'f'
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa5c'),
    name: 'Pilot',
```

Практическое задание 2.1.4:

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

```
> db.unicorns.find({}, {_id: 0, 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'
    ],
```

Практическое задание 2.3.1:

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

```
> db.unicorns.find({weight: {$gte: 500, $lte: 700}}, {_id: 0})
< {
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
  {
    name: 'Roooooodles',
    loves: [
      'apple'
    ],
    weight: 575,
    gender: 'm',
    vampires: 99
  }
  {
    name: 'Solnara',
    loves: [
      'apple',
      'carrot',
      'chocolate'
    ],
    weight: 550,
    gender: 'f',
    vampires: 80
  }
  {
```

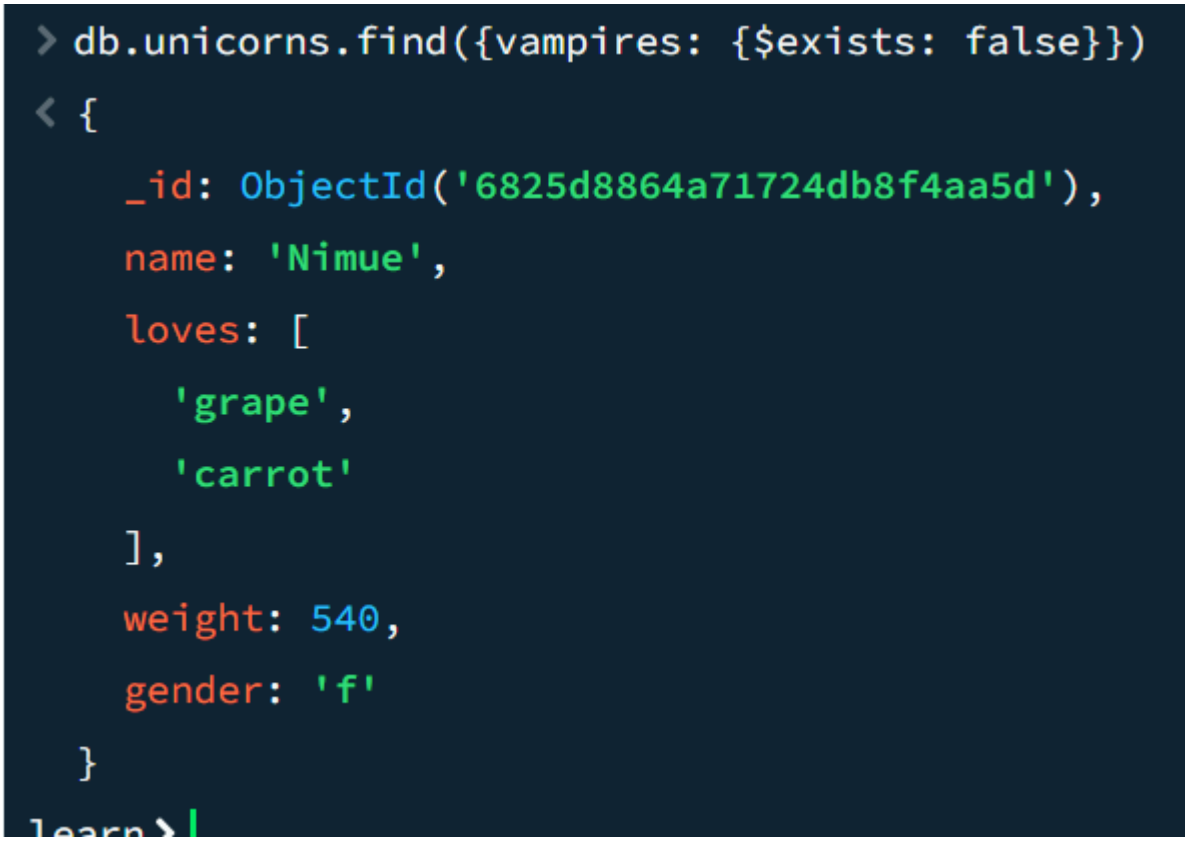
Практическое задание 2.3.2:

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

```
> db.unicorns.find({weight: {$gte: 500}, loves: {$in: ['grape', 'lemon']}}, {_id: 0})
< {
    name: 'Ayna',
    loves: [
      'strawberry',
      'lemon'
    ],
    weight: 733,
    gender: 'f',
    vampires: 40
  }
  {
    name: 'Kenny',
    loves: [
      'grape',
      'lemon'
    ],
    weight: 690,
    gender: 'm',
    vampires: 39
  }
  {
    name: 'Nimue',
    loves: [
      'grape',
      'carrot'
    ],
    weight: 540,
```

Практическое задание 2.3.3:

Найти всех единорогов, не имеющих ключ `vampires`.



Практическое задание 2.3.4:

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

```
> db.unicorns.find({}, {name: 1, _id: 0, loves: {$slice: 1}})
< {
    name: 'Horny',
    loves: [
      'carrot'
    ]
  }
  {
    name: 'Aurora',
    loves: [
      'carrot'
    ]
  }
  {
    name: 'Unicrom',
    loves: [
      'energon'
    ]
  }
  {
    name: 'Roooooodles',
    loves: [
      'apple'
    ]
  }
  {
    name: 'Solnara',
    loves: [
      'apple'
    ]
  }
```

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"}, {name: 1, mayor: 1})
< {
    _id: ObjectId('6825e4544a71724db8f4aa60'),
    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.

```
db.towns.find({"mayor.party": {$exists: false}}, {name: 1, mayor: 1})
{
  _id: ObjectId('6825e4464a71724db8f4aa5f'),
  name: 'Punxsutawney ',
  mayor: {
    name: 'Jim Wehrle'
  }
}
```

Практическое задание 3.1.2:

1) *Сформировать функцию для вывода списка самцов единорогов.*

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

2) *Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке. Вывести результат, используя forEach.*

```
> mycursor = db.unicorns.find(func()).limit(2).sort({name: 1});null;
< null
> mycursor
< {
    _id: ObjectId('6825d8c04a71724db8f4aa5e'),
    name: 'Dunx',
    loves: [
      'grape',
      'watermelon'
    ],
    weight: 704,
    gender: 'm',
    vampires: 165
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa53'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 63
  }
learn>
```

Практическое задание 3.2.1:

Вывести количество самок единорогов весом от полутонны до 600 кг.

```
> db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 600}}).count()
< 4
```

Практическое задание 3.2.2:

Вывести список предпочтений.

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

Практическое задание 3.2.3:

Посчитать количество особей единорогов обоих полов.

```
> db.unicorns.aggregate({"$group": {_id: "$gender", count: {$sum: 1}}})
< {
    _id: 'm',
    count: 14
  }
  {
    _id: 'f',
    count: 10
  }
learn>
```

```
WriteResult({"nInserted" : 1 })
```

Практическое задание 3.3.1:

1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции* `unicorns`.

```
> db.unicorns.replaceOne({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 0,
    modifiedCount: 0,
    upsertedCount: 0
  }
learn >
```

Практическое задание 3.3.2:

1. *Для самки единорога* `Ayna` *внести изменения в БД: теперь ее вес 800, она убила 51 вампира.*
2. *Проверить содержимое коллекции* `unicorns`.


```
> db.users.update({name : "Tom"},
 {$set: {name: "Tom", age : 25, married : false}}, {multi:true})
```

Практическое задание 3.3.3:

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

```
> db.unicorns.updateOne({name: "Raleigh"}, {$set: {loves: ["Redbull"]}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: "Raleigh"})
< {
    _id: ObjectId('6825d8864a71724db8f4aa5a'),
    name: 'Raleigh',
    loves: [
      'Redbull'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
  {
    _id: ObjectId('6825f92fd10b907388ab256e'),
    name: 'Raleigh',
    loves: [
      'apple',
      'sugar'
    ],
    weight: 421,
    gender: 'm',
    vampires: 2
  }
learn >
```

Практическое задание 3.3.4:

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

```
> db.unicorns.updateOne({gender: "m"}, {$inc: {vampires: 5}}
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find()
< {
    _id: ObjectId('6825d8864a71724db8f4aa53'),
    name: 'Horny',
    loves: [
      'carrot',
      'papaya'
    ],
    weight: 600,
    gender: 'm',
    vampires: 68
  }
  {
    _id: ObjectId('6825d8864a71724db8f4aa54'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
```

Практическое задание 3.3.5:

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

```
learn > db.towns.updateOne({name: "Portland"}, {$unset: {"mayor.party": 1}})
```

```
{
  _id: ObjectId('6825e4604a71724db8f4aa61'),
  name: 'Portland',
  populatiuon: 528000,
  last_sensus: 2009-07-20T00:00:00.000Z,
  famous_for: [
    'beer',
    'food'
  ],
  mayor: {
    name: 'Sam Adams'
  }
}
```

Практическое задание 3.3.6:

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

```
> db.unicorns.updateOne({name: "Pilot", gender: 'm'}, {$push: {loves: "chokolate"}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Pilot'})
< {
    _id: ObjectId('6825d8864a71724db8f4aa5c'),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon',
      'chokolate'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
  {
    _id: ObjectId('6825f92fd10b907388ab2570'),
    name: 'Pilot',
    loves: [
      'apple',
      'watermelon'
    ],
    weight: 650,
    gender: 'm',
    vampires: 54
  }
```

Практическое задание 3.3.7:

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

```
> db.unicorns.updateMany({gender: 'f', name:"Aurora"}, {$addToSet: {loves: {$each: ['sugar', 'lemon']}}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 2,
    modifiedCount: 2,
    upsertedCount: 0
  }
> db.unicorns.find({name: 'Aurora'})
< {
    _id: ObjectId('6825d8864a71724db8f4aa54'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
  {
    _id: ObjectId('6825f914d10b907388ab2568'),
    name: 'Aurora',
    loves: [
      'carrot',
      'grape',
      'sugar',
      'lemon'
    ],
    weight: 450,
    gender: 'f',
    vampires: 43
  }
learn>
```

Практическое задание 3.4.1:

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

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

```
db.towns.find()
{
  _id: ObjectId('6825e4464a71724db8f4aa5f'),
  name: 'Punxsutawney ',
  populatiuon: 6200,
  last_sensus: 2008-01-31T00:00:00.000Z,
  famous_for: [
    ''
  ],
  mayor: {
    name: 'Jim Wehrle'
  }
}
{
  _id: ObjectId('6825e4544a71724db8f4aa60'),
  name: 'New York',
  populatiuon: 22200000,
  last_sensus: 2009-07-31T00:00:00.000Z,
  famous_for: [
    'status of liberty',
    'food'
  ],
  mayor: {
    name: 'Michael Bloomberg',
    party: 'I'
  }
}
{
  _id: ObjectId('6825e4604a71724db8f4aa61'),
  name: 'Portland',
  populatiuon: 528000,
  last_sensus: 2009-07-20T00:00:00.000Z,
```

4.1.1

```
db.habitatzones.insertMany([
    { _id: "forest",   name: "Forest",    description: "Густой лес" },
    { _id: "meadow",   name: "Meadow",       description: "Луг" },
    { _id: "mountain", name: "Mountains",   description: "горы" }])
```

```
{ "acknowledged" : true, "insertedIds" : [ "forest", "meadow", "mountain"
] }
```

```
db.unicorns.update({ name: "Horny" }, { $set: { zone: { $ref:
"habitatzones", $id: "forest" } } })

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

db.unicorns.update({ name: "Aurora" }, { $set: { zone: { $ref:
"habitatzones", $id: "meadow" } } })

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

db.unicorns.update({ name: "Unicrom" }, { $set: { zone: { $ref:
"habitatzones", $id: "mountain" } } })

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

```
db.unicorns.find().limit(5).pretty()

{ "_id" : ObjectId("…"), "name" : "Horny", "loves" : [ "carrot","papaya"
], "weight" : 600, "gender" : "m", "vampires" : 63, "zone" : {
"$ref":"habitatzones","$id":"forest" } }

{ "_id" : ObjectId("…"), "name" : "Aurora", "loves" : [ "carrot","grape"
], "weight" : 450, "gender" : "f", "vampires" : 43, "zone" : {
"$ref":"habitatzones","$id":"meadow" } }

{ "_id" : ObjectId("…"), "name" : "Unicrom", "loves" : [
"energon","redbull"], "weight" : 984, "gender" : "m", "vampires" :182,
"zone" : { "$ref":"habitatzones","$id":"mountain" } }
```

4.2.1

```
db.unicorns.createIndex({ name: 1 }, { unique: true })

"name_1"
```

4.3.1

```
db.unicorns.getIndexes()

[

    { v: 2, key: { _id: 1 }, name: "_id_", ns: "test.unicorns" },

    { v: 2, key: { name: 1 }, name: "name_1", ns: "test.unicorns" }

]
```

```
db.unicorns.dropIndex("name_1")

{ "nIndexesWas" : 2, "ok" : 1 }


db.unicorns.getIndexes()

[

    { v: 2, key: { _id: 1 }, name: "_id_", ns: "test.unicorns" }

]


db.unicorns.dropIndex("_id_")

{ "ok" : 0, "errmsg" : "index _id_ cannot be dropped", "code" : 85,
"codeName" : "IndexOptionsConflict" }
```

4.4.1

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


db.numbers.find().sort({ value: -1 }).limit(4).toArray()

[ { "_id": ObjectId("…"), "value": 99999 },

  { "_id": ObjectId("…"), "value": 99998 },

  { "_id": ObjectId("…"), "value": 99997 },

  { "_id": ObjectId("…"), "value": 99996 } ]


var planNoIdx = db.numbers.explain("executionStats").find().sort({ value:
-1 }).limit(4)

planNoIdx.executionStats.executionTimeMillis

85


db.numbers.createIndex({ value: 1 })

"value_1"
```

```
db.numbers.getIndexes()

[

    { key: { _id: 1 },   name: "_id_"   },

    { key: { value: 1 }, name: "value_1" }

]



var   planWithIdx   =   db.numbers.explain("executionStats").find().sort({
value: -1 }).limit(4)

planWithIdx.executionStats.executionTimeMillis

5
```