Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра экономической информатики

Лабораторная работа №6-7
«Системы и методы управления базами данных»

Выполнил: студент гр. 914302

Шпаковский М.Г.

Проверил: Лукашевич А.Э.

Минск 2022

```
> db.createCollection("towns")
< { ok: 1 }
> db.towns.insert({name: "Punxsutawney ",
  populatiuon: 6200,
  last_sensus: ISODate("2022-01-31"),
  famous_for: [""],
  mayor: {
     name: "Jim Wehrle"
  }}
  )
< { acknowledged: true,
    insertedIds: { '0': ObjectId("6385ff7bc69a24a1550ca3c3") } }
> db.towns.find()
< { _id: ObjectId("6385ff7bc69a24a1550ca3c3"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: 2022-01-31T00:00:00.000Z,
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' } }
> db.towns.insert({name: "New York",
  populatiuon: 22200000,
  last_sensus: ISODate("2022-07-31"),
  famous_for: ["status of liberty", "food"],
  mayor: {
     name: "Michael Bloomberg",
  party: "I"}}
```

2.

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

3.

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

Практическое задание 2

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

Практическое задание 3

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

количество особей единорогов обоих полов

```
> db.unicorns.aggregate({"$group":{_id:"$gender",count:{$sum:1}}})
< { _id: 'f', count: 6 }
  { _id: 'm', count: 7 }
```

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

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

```
{ _id: ObjectId("638526f5c69a24a1550ca3b8"),
  name: 'Ayna',
  dob: 1998-03-07T05:30:00.000Z,
  loves: [ 'strawberry', 'lemon' ],
  weight: 800,
  gender: 'f',
  vampires: 51 }
```

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

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

```
> db.unicorns.updateMany({}, {$inc: {vampires: 5}}, {multi:true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 13,
    modifiedCount: 13,
    upsertedCount: 0 }
```

```
> db.unicorns.find()
< { _id: ObjectId("638525bfc69a24a1550ca3b2"),
    name: 'Aurora',
    gender: 'f',
    weight: 450,
    vampires: 5 }
  { _id: ObjectId("638526f5c69a24a1550ca3b3"),
    name: 'Horny',
    dob: 1992-03-13T04:47:00.000Z,
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 68 }
  { _id: ObjectId("638526f5c69a24a1550ca3b4"),
    name: 'Aurora',
    dob: 1991-01-24T10:00:00.000Z,
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
    vampires: 48 }
```

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

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

```
> db.towns.find()
< { _id: ObjectId("6385ff7bc69a24a1550ca3c3"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: 2022-01-31T00:00:00.000Z,
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' } }
  { _id: ObjectId("6385ffb4c69a24a1550ca3c4"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: 2022-07-31T00:00:00.000Z,
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' } }
  { _id: ObjectId("63860013c69a24a1550ca3c5"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2022-07-20T00:00:00.000Z,
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: '' } }
```

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

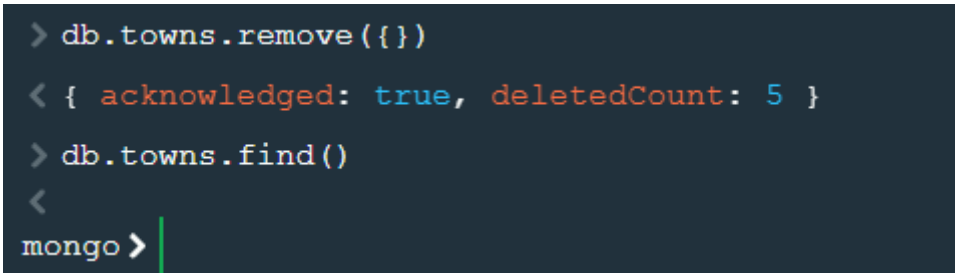
```
> db.unicorns.updateOne({name : "Pilot"}, {$push: {loves: "chocolate"}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
> db.towns.find()
< { _id: ObjectId("6385ff7bc69a24a1550ca3c3"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: 2022-01-31T00:00:00.000Z,
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' } }
  { _id: ObjectId("6385ffb4c69a24a1550ca3c4"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: 2022-07-31T00:00:00.000Z,
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' } }
  { _id: ObjectId("63860013c69a24a1550ca3c5"),
    name: 'Portland',
    populatiuon: 528000,
    last_sensus: 2022-07-20T00:00:00.000Z,
    famous_for: [ 'beer', 'food' ],
    mayor: { name: 'Sam Adams', party: '' } }
```

Удалите документы с беспартийными мэрами

```
> db.towns.remove({"mayor.party": ""})
< 'DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.'
< { acknowledged: true, deletedCount: 1 }
```

```
> db.towns.find()
< { _id: ObjectId("6385ff7bc69a24a1550ca3c3"),
    name: 'Punxsutawney ',
    populatiuon: 6200,
    last_sensus: 2022-01-31T00:00:00.000Z,
    famous_for: [ '' ],
    mayor: { name: 'Jim Wehrle' } }
  { _id: ObjectId("6385ffb4c69a24a1550ca3c4"),
    name: 'New York',
    populatiuon: 22200000,
    last_sensus: 2022-07-31T00:00:00.000Z,
    famous_for: [ 'status of liberty', 'food' ],
    mayor: { name: 'Michael Bloomberg', party: 'I' } }
  { _id: ObjectId("63860536c69a24a1550ca3c6"),
    name: 'Punxsutawney ',
    popujatiuon: 6200,
    last_sensus: 2022-01-31T00:00:00.000Z,
    famous_for: [ 'phil the groundhog' ],
    mayor: { name: 'Jim Wehrle' } }
  { _id: ObjectId("63860552c69a24a1550ca3c7"),
    name: 'New York',
    popujatiuon: 22200000,
    last_sensus: 2022-07-31T00:00:00.000Z,
    famous_for: [ 'status of liberty', 'food' ],
```

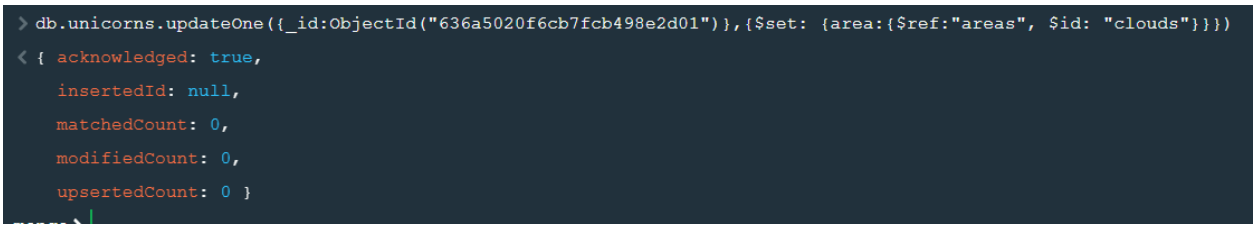
1) *Очистите коллекцию.*



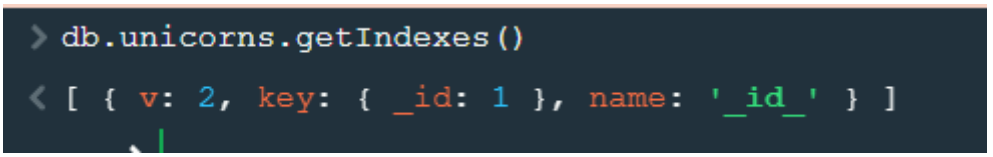
2 часть лаба 7

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

1) *Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания.*



Получите информацию о всех индексах коллекции `unicorns`



Удалите все индексы, кроме индекса для идентификатора.

```
> db.unicorns.dropIndex("name_1")
< {
    ok: 0,
    errmsg: 'index not found with name [name_1]',
    code: 27,
    codeName: 'IndexNotFound'
  }
```

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

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

```
> db.createCollection("numbers")
< { ok: 1 }
```

```
> db.numbers.find().sort({$natural:-1}).limit(4)
< { _id: ObjectId("6386083ec69a24a1550cc808"), value: 9278 }
  { _id: ObjectId("6386083ec69a24a1550cc807"), value: 9277 }
  { _id: ObjectId("6386083ec69a24a1550cc806"), value: 9276 }
  { _id: ObjectId("6386083ec69a24a1550cc805"), value: 9275 }
```

```
> db.numbers.explain("executionStats").find({})
< { explainVersion: '1',
    queryPlanner:
     { namespace: 'mongo.numbers',
       indexFilterSet: false,
       parsedQuery: {},
       queryHash: '17830885',
       planCacheKey: '17830885',
       maxIndexedOrSolutionsReached: false,
       maxIndexedAndSolutionsReached: false,
       maxScansToExplodeReached: false,
       winningPlan: { stage: 'COLLSCAN', direction: 'forward' },
       rejectedPlans: [] },
    executionStats:
     { executionSuccess: true,
       nReturned: 16933,
       executionTimeMillis: 8,
       totalKeysExamined: 0,
       totalDocsExamined: 16933,
       executionStages:
        { stage: 'COLLSCAN',
          nReturned: 16933,
          executionTimeMillisEstimate: 1,
          works: 16935,
```

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

```
> db.numbers.dropIndex("value_1")
< { nIndexesWas: 2, ok: 1 }
```