Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра экономической информатики

ОТЧЕТ
Лабораторные работы №6-7

Выполнил: студент гр.

914302 Борисенко А.И.

Проверил: Лукашевич А.Э.

Минск 2022

1) Создание коллекции towns, включающую следующие документы:

```
{name: "Punxsutawney ",
populatiuon: 6200,
last_sensus: ISODate("2022-01-31"),
famous_for: [""],
mayor: {
  name: "Jim Wehrle"
}}
{name: "New York",
populatiuon: 22200000,
last_sensus: ISODate("2022-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
  name: "Michael Bloomberg",
party: "I"}}
{name: "Portland",
populatiuon: 528000,
last_sensus: ISODate("2022-07-20"),
famous_for: ["beer", "food"],
mayor: {
  name: "Sam Adams",
party: "D"}}
```

```
> db.towns.insertMany([{name: "Portland",
  populatiuon: 528000,
  last_sensus: ISODate("2022-07-20"),
  famous_for: ["beer", "food"],
  mayor: {
     name: "Sam Adams",
  party: "D"}
  }, {name: "New York",
  populatiuon: 22200000,
  last_sensus: ISODate("2022-07-31"),
  famous_for: ["status of liberty", "food"],
  mayor: {
     name: "Michael Bloomberg",
  party: "I"}}]
  )
< { acknowledged: true,
    insertedIds:
     { '0': ObjectId("638284c3ccd148e86cc6aba6"),
       '1': ObjectId("638284c3ccd148e86cc6aba7") } }
> db.towns.insertOne({name: "Punxsutawney ",
  populatiuon: 6200,
  last_sensus: ISODate("2022-01-31"),
  famous_for: [""],
  mayor: {
     name: "Jim Wehrle"
  }}
  )
< { acknowledged: true,
    insertedId: ObjectId("638284eaccd148e86cc6aba8") }
learn>
```

*2) Формирование 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.*

```
> db.towns.find({"mayor.party": "I"}, {name: 1, mayor:1})
< { _id: ObjectId("638284c3ccd148e86cc6aba7"),
    name: 'New York',
    mayor: { name: 'Michael Bloomberg', party: 'I' } }
> db.towns.find({"mayor.party": {$exists: 0}}, {name: 1, mayor:1})
< { _id: ObjectId("638284eaccd148e86cc6aba8"),
    name: 'Punxsutawney ',
    mayor: { name: 'Jim Wehrle' } }
learn> db.towns.find({"mayor.party": {$exists: 0}}, {name: 1, mayor:1})
```

Практическое задание 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.

4) Содержание коллекции 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', 44), 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'});
db.unicorns.insert ({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})

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

Практическое задание 3:

Вывести количество самок единорогов весом от 500 до 600 кг.

Практическое задание 5:

Посчитать количество особей единорогов обоих полов.

```
> db.unicorns.aggregate({"$group":{_id:"$gender",count:{$sum:1}}})
< { _id: 'm', count: 7 }
  { _id: 'f', count: 5 }
> db.unicorns.save({name: 'Barny', loves: ['grape'],
  weight: 340, gender: 'm'})
```

Практическое задание 6:

1. *Выполнить команду:*

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

Проверить содержимое коллекции unicorns.

```
> db.unicorns.save({name: 'Barny', loves: ['grape'], weight: 340, gender: 'm'})
✖ ▸ TypeError: db.unicorns.save is not a function
> db.unicorns.find()
< { _id: ObjectId("63814402ccd148e86cc6ab9a"),
    name: 'Horny',
    dob: 1992-03-13T04:47:00.000Z,
    loves: [ 'carrot', 'papaya' ],
    weight: 600,
    gender: 'm',
    vampires: 63 }
  { _id: ObjectId("63814402ccd148e86cc6ab9b"),
    name: 'Aurora',
    dob: 1991-01-24T10:00:00.000Z,
    loves: [ 'carrot', 'grape' ],
    weight: 450,
    gender: 'f',
```

Практическое задание 7:

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

2. Проверить содержимое коллекции unicorns.

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

Практическое задание 8:

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"]}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
learn>
```

Практическое задание 9:

1. Всем самцам единорогов увеличить количество убитых вапмиров на 5.

2. Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateMany({}, {$inc: {vampires: 5}}, {multi:true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 12,
    modifiedCount: 12,
    upsertedCount: 0 }
learn>
```

Практическое задание 10:

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

2. Проверить содержимое коллекции towns.

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

Практическое задание 11:

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

Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateOne({name : "Pilot"}, {$push: {loves: "chocolate"}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
learn>
```

Практическое задание 12:

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

2. Проверить содержимое коллекции unicorns.

```
> db.unicorns.updateOne({name : "Aurora"}, {$push: {loves: ["sugar", "limons"]}}, {upsert: true})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
learn>
```

Практическое задание 13:

1) Создайте коллекцию towns, включающую следующие документы:

{name: "Punxsutawney ",
popujatiuon: 6200,
last_sensus: ISODate("2022-01-31"),
famous_for: ["phil the groundhog"],

mayor: {
  name: "Jim Wehrle"
}}
{name: "New York",
popujatiuon: 22200000,
last_sensus: ISODate("2022-07-31"),
famous_for: ["status of liberty", "food"],
mayor: {
  name: "Michael Bloomberg",
party: "I"}}
{name: "Portland",
popujatiuon: 528000,
last_sensus: ISODate("2022-07-20"),
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.