# Министерство науки и высшего образования Российской Федерации федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО» Факультет инфокоммуникационных технологий

# Лабораторная работа №5.2 «Работа с БД в СУБД MongoDB» по дисциплине «Проектирование и реализация баз данных»

Выполнил: студент II курса ИКТ группы К3241 Конев А.

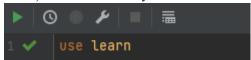
Проверил: Говорова М.М.

**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.

#### Выполнение:

#### Практическое задание 8.1.1:

1) Создайте базу данных learn.



2) Заполните коллекцию единорогов unicorns:

```
db.unicorns.insertOne({name: 'Horny', loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});

db.unicorns.insertOne({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});

db.unicorns.insertOne({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});

db.unicorns.insertOne({name: 'Roooooodles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});

db.unicorns.insertOne({name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});

db.unicorns.insertOne({name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});

db.unicorns.insertOne({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});

db.unicorns.insertOne({name: 'Leia', loves: ['apple', 'sugar'], weight: 601, gender: 'f', vampires: 53});

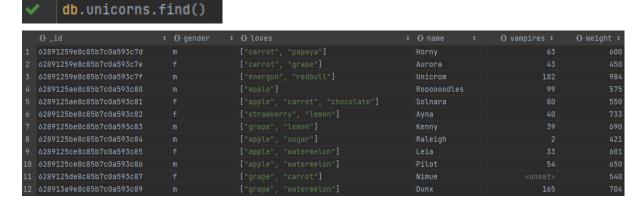
db.unicorns.insertOne({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});

db.unicorns.insertOne({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
```

3) Используя второй способ, вставьте в коллекцию единорогов документ:

```
document=({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
db.unicorns.insertOne(document)
```

4) Проверьте содержимое коллекции с помощью метода find.



#### Практическое задание 8.1.2:

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

```
db.unicorns.find({gender: 'm'}).limit(3).sort({name: 1})
```





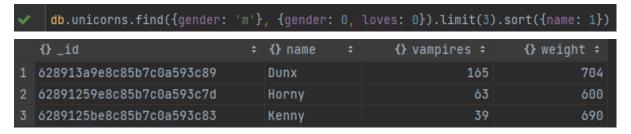
	()_id		{} gender ÷		{} loves	{} name	{} vampires ÷	{} weight ÷
1	62891259e8c85b7c0a593c7e	1	f	[	["carrot", "grape"]	Aurora	43	450
2	6289125be8c85b7c0a593c82	1	f	[	["strawberry", "lemon"]	Ayna		733
3	6289125ce8c85b7c0a593c85	1	f	[	["apple", "watermelon"]	Leia	33	601

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



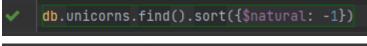
# Практическое задание 8.1.3:

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



# Практическое задание 8.1.4:

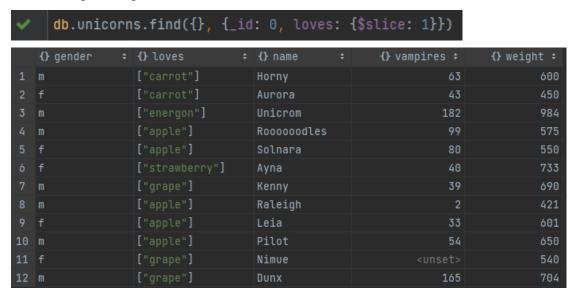
Вывести список единорогов в обратном порядке добавления.



	<pre>{}_id</pre>	{} gender ‡	{} loves		{} vampires ÷	{} weight ‡
1	628913a9e8c85b7c0a593c89			Dunx	165	704
2	6289125de8c85b7c0a593c87			Nimue		540
3	6289125ce8c85b7c0a593c86			Pilot	54	650
4	6289125ce8c85b7c0a593c85			Leia	33	601
5	6289125be8c85b7c0a593c84			Raleigh		421
6	6289125be8c85b7c0a593c83			Kenny		690
7	6289125be8c85b7c0a593c82			Ayna		733
8	6289125ae8c85b7c0a593c81			Solnara	80	550
9	6289125ae8c85b7c0a593c80			Roooooodles	99	575
10	62891259e8c85b7c0a593c7f			Unicrom	182	984
11	62891259e8c85b7c0a593c7e			Aurora	43	450
12	62891259e8c85b7c0a593c7d	m	["carrot", "papaya"]	Horny	63	600

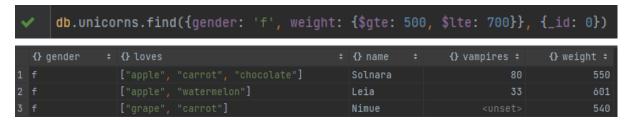
# Практическое задание 8.1.5:

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



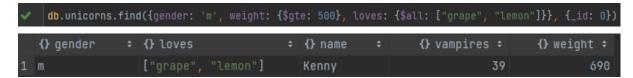
#### Практическое задание 8.1.6:

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



#### Практическое задание 8.1.7:

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



#### Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ vampires.



# Практическое задание 8.1.9:

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

# Практическое задание 8.2.1:

1) Создайте коллекцию towns, включающую следующие документы:

```
db.towns.insertOne({name: "Punxsutawney ",
    populatiuon: 6200,
    last_sensus: ISODate("2008-01-31"),
    famous_for: [""],
    mayor: {
        name: "Jim Wehrle"
    }})
    db.towns.insertOne({name: "New York",
        populatiuon: 22200000,
    last_sensus: ISODate("2009-07-31"),
    famous_for: ["status of liberty", "food"],
    mayor: {
        name: "Michael Bloomberg",
        party: "I"}})
    db.towns.insertOne({name: "Portland",
        populatiuon: 528000,
        last_sensus: ISODate("2009-07-20"),
        famous_for: ["beer", "food"],
        mayor: {
            name: "Sam Adams",
            party: "D"}}
    )
}
```

2) Сформировать 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.

# Практическое задание 8.2.2:

1) Сформировать функцию для вывода списка самцов единорогов.

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

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

3) Вывести результат, используя forEach

#### Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

#### Практическое задание 8.2.4:

Вывести список предпочтений.

```
✓ db.unicorns.distinct("loves")
```



#### Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

#### Практическое задание 8.2.6:

1. Выполнить команду:

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

2. Проверить содержимое коллекции unicorns.



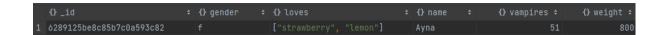
#### Практическое задание 8.2.7:

1. Для самки единорога Ayna внести изменения в BA: теперь ее вес 800, она убила 51 вапмира.

```
db.unicorns.updateOne({"name": "Ayna"}, {$set: {name: "Ayna", loves: ["strawberry", "lemon"], weight: 800, gender: 'f', vampires: 51}})
```

2. Проверить содержимое коллекции unicorns.

```
db.unicorns.find({name: "Ayna"})
```



#### Практическое задание 8.2.8:

 $I.\;\;$ Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул.

```
db.unicorns.updateOne({"gender": 'm', "name": "Raleigh"}, {$set: {loves: ["redbull"]}})
```

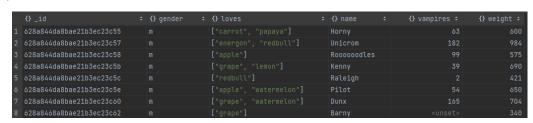
2. Проверить содержимое коллекции unicorns.

```
✓ db.unicorns.find({name: "Raleigh"})
```

O _id	<b>‡</b>	{} gender	<b>‡</b>	{} loves	<b>‡</b>	{} name	<b>‡</b>	{} vampires ÷	{} weig	ht ÷
1 628a844da8bae21b3ec23c5c				["redbull"]		Raleigh		2		421

# Практическое задание 8.2.9:

До



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

```
db.unicorns.updateMany({"gender": 'm'}, {$inc: {vampires: 5}})
```

2. Проверить содержимое коллекции unicorns.



#### Практическое задание 8.2.10:

До

```
      {} _id
      : {} _famous_for
      : {} _fam
```

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

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

2. Проверить содержимое коллекции towns.



#### Практическое задание 8.2.11:

До

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

```
db.unicorns.updateOne({"name": "Pilot"}, {$push: {loves: 'chocolate'}})
```

2. Проверить содержимое коллекции unicorns.

# Практическое задание 8.2.12:

До



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

```
✓ db.unicorns.updateOne({"name": "Aurora"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
```

2. Проверить содержимое коллекции unicorns.

# Практическое задание 8.2.13:

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.

db.towns.find()



5) Просмотрите список доступных коллекций.



#### Практическое задание 8.3.1:

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.

```
db.unicorns.updateOne({name: "Horny"}, {$set: {areas: {$ref: "areas", $id: "cove"}}})

db.unicorns.updateOne({name: "Aurora"}, {$set: {areas: {$ref: "areas", $id: "wellspring"}}})
```

3) Проверьте содержание коллекции едиорогов.



### Практическое задание 8.3.2:

1. Проверьте, можно ли задать для коллекции unicorns индекс для ключа name c флагом unique.

#### Практическое задание 8.3.3:

1) Получите информацию о всех индексах коллекции unicorns.

2) Удалите все индексы, кроме индекса для идентификатора.

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

```
{} nIndexesWas $ {} ok $

1 2 1
```

3) Попытайтесь удалить индекс для идентификатора.

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

Command failed with error 72 (InvalidOptions): 'cannot drop_id index' on server docker.civiltechgroup.ru:27017. The full response is {"ok": 0.0, "errmsg": "cannot drop_id index", "code": 72, "codeName": "InvalidOptions"}
```

#### Практическое задание 8.3.4:

1) Создайте объемную коллекцию numbers, задействовав курсор:

```
for(i = 0; i < 100000; i++){db.numbers.insertOne({value: i})}</pre>
```

2) Выберите последних четыре документа.

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

```
db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)

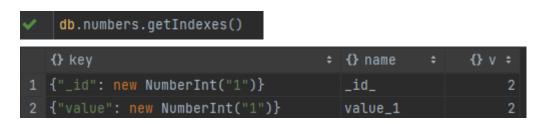
lab> db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)

[2022-05-22 20:11:23] 1 row retrieved starting from 1 in 428 ms (execution: 423 ms, fetching: 5 ms)

"executionStats": {
    "executionSuccess": true,
    "nReturned": 4,
    "executionTimeMillis": 0,
```

4) Создайте индекс для ключа value.

5) Получите информацию о всех индексах коллекции numbers.



6) Выполните запрос 2.



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

```
lob> db.numbers.find().sort({$natural: -1}).limit(4)
[2022-05-22 20:26:18] 4 rows retrieved starting from 1 in 235 ms (execution: 202 ms, fetching: 33 ms)

lob> db.numbers.explain("executionStats").find().sort({$natural: -1}).limit(4)
[2022-05-22 20:19:36] 1 row retrieved starting from 1 in 412 ms (execution: 389 ms, fetching: 23 ms)

"executionStats": {
    "executionSuccess": true,
    "nReturned": 4,
    "executionTimeMillis": 0,
    "totalKeysExamined": 0
```

8) Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

К сожалению DataGrip не смог показать разницу во времени исполнения. Поэтому, ориентируясь на время исполнения запроса, отображающее в консоли, можно говорить о разнице в 29 мс (1 запрос - 231 мс, 2 запрос - 202 мс). Если же говорить о времени выполнения запроса с использованием команды *explain()*, то разница составляет 34 мс. Из этого можно сделать вывод, что для ускорения часто применяемых запросов стоит использовать индексы.

**Вывод:** В ходе выполнения работы были созданы запросы на вставку, изменение, удаление документов, выборку данных по критериям, на работу с индексами и ссылками в MongoDB.