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 K3241
Конев А.

Проверил:
Говорова М.М.

Санкт-Петербург
2022

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.

```
use 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: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
db.unicorns.insertOne({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
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.

```
db.unicorns.find()
```

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	62891259e8c85b7c0a593c7d	m	["carrot", "papaya"]	Horny	63	600
2	62891259e8c85b7c0a593c7e	f	["carrot", "grape"]	Aurora	43	450
3	62891259e8c85b7c0a593c7f	m	["energon", "redbull"]	Unicrom	182	984
4	6289125ae8c85b7c0a593c80	m	["apple"]	Roooooodles	99	575
5	6289125ae8c85b7c0a593c81	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	6289125be8c85b7c0a593c82	f	["strawberry", "lemon"]	Ayna	40	733
7	6289125be8c85b7c0a593c83	m	["grape", "lemon"]	Kenny	39	690
8	6289125be8c85b7c0a593c84	m	["apple", "sugar"]	Raleigh	2	421
9	6289125ce8c85b7c0a593c85	f	["apple", "watermelon"]	Leia	33	601
10	6289125ce8c85b7c0a593c86	m	["apple", "watermelon"]	Pilot	54	650
11	6289125de8c85b7c0a593c87	f	["grape", "carrot"]	Nimue	<unset>	540
12	628913a9e8c85b7c0a593c89	m	["grape", "watermelon"]	Dunx	165	704

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	628913a9e8c85b7c0a593c89	m	["grape", "watermelon"]	Dunx	165	704
2	62891259e8c85b7c0a593c7d	m	["carrot", "papaya"]	Horny	63	600
3	6289125be8c85b7c0a593c83	m	["grape", "lemon"]	Kenny	39	690

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

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	62891259e8c85b7c0a593c7e	f	["carrot", "grape"]	Aurora	43	450
2	6289125be8c85b7c0a593c82	f	["strawberry", "lemon"]	Ayna	40	733
3	6289125ce8c85b7c0a593c85	f	["apple", "watermelon"]	Leia	33	601

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

```
db.unicorns.findOne({gender: 'f', loves: 'carrot'})
```

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	62891259e8c85b7c0a593c7e	f	["carrot", "grape"]	Aurora	43	450

```
db.unicorns.find({gender: 'f', loves: 'carrot'}).limit(1)
```

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	62891259e8c85b7c0a593c7e	f	["carrot", "grape"]	Aurora	43	450

Практическое задание 8.1.3:

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

```
db.unicorns.find({gender: 'm'}, {gender: 0, loves: 0}).limit(3).sort({name: 1})
```

	{} _id	{} name	{} vampires	{} weight
1	628913a9e8c85b7c0a593c89	Dunx	165	704
2	62891259e8c85b7c0a593c7d	Horny	63	600
3	6289125be8c85b7c0a593c83	Kenny	39	690

Практическое задание 8.1.4:

Вывести список единорогов в обратном порядке добавления.

```
db.unicorns.find().sort({$natural: -1})
```

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	628913a9e8c85b7c0a593c89	m	["grape", "watermelon"]	Dunx	165	704
2	6289125de8c85b7c0a593c87	f	["grape", "carrot"]	Nimue	<unset>	540
3	6289125ce8c85b7c0a593c86	m	["apple", "watermelon"]	Pilot	54	650
4	6289125ce8c85b7c0a593c85	f	["apple", "watermelon"]	Leia	33	601
5	6289125be8c85b7c0a593c84	m	["apple", "sugar"]	Raleigh	2	421
6	6289125be8c85b7c0a593c83	m	["grape", "lemon"]	Kenny	39	690
7	6289125be8c85b7c0a593c82	f	["strawberry", "lemon"]	Ayna	40	733
8	6289125ae8c85b7c0a593c81	f	["apple", "carrot", "chocolate"]	Solnara	80	550
9	6289125ae8c85b7c0a593c80	m	["apple"]	Roooooodles	99	575
10	62891259e8c85b7c0a593c7f	m	["energon", "redbull"]	Unicrom	182	984
11	62891259e8c85b7c0a593c7e	f	["carrot", "grape"]	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.

```
db.unicorns.find({}, {_id: 0, loves: {$slice: 1}})
```

	gender	loves	name	vampires	weight
1	m	["carrot"]	Horny	63	600
2	f	["carrot"]	Aurora	43	450
3	m	["energon"]	Unicrom	182	984
4	m	["apple"]	Roooooodles	99	575
5	f	["apple"]	Solnara	80	550
6	f	["strawberry"]	Ayna	40	733
7	m	["grape"]	Kenny	39	690
8	m	["apple"]	Raleigh	2	421
9	f	["apple"]	Leia	33	601
10	m	["apple"]	Pilot	54	650
11	f	["grape"]	Nimue	<unset>	540
12	m	["grape"]	Dunx	165	704

Практическое задание 8.1.6:

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

```
db.unicorns.find({gender: 'f', weight: {$gte: 500, $lte: 700}}, {_id: 0})
```

	gender	loves	name	vampires	weight
1	f	["apple", "carrot", "chocolate"]	Solnara	80	550
2	f	["apple", "watermelon"]	Leia	33	601
3	f	["grape", "carrot"]	Nimue	<unset>	540

Практическое задание 8.1.7:

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

```
db.unicorns.find({gender: 'm', weight: {$gte: 500}, loves: {$all: ["grape", "lemon"]}}, {_id: 0})
```

	gender	loves	name	vampires	weight
1	m	["grape", "lemon"]	Kenny	39	690

Практическое задание 8.1.8:

Найти всех единорогов, не имеющих ключ `vampires`.

```
db.unicorns.find({vampires: {$exists: false}})
```

	_id	gender	loves	name	weight
1	6289125de8c85b7c0a593c87	f	["grape", "carrot"]	Nimue	540

Практическое задание 8.1.9:

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

```
db.unicorns.find({gender: 'm'}, {name: 1, _id: 0, loves: {$slice: 1}}).sort({name: 1})
```

	{} loves	{} name
1	["grape"]	Dunx
2	["carrot"]	Horny
3	["grape"]	Kenny
4	["apple"]	Pilot
5	["apple"]	Raleigh
6	["apple"]	Roooooodles
7	["energon"]	Unicrom

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.*

```
db.towns.find({"mayor.party": "I"}, {_id: 0, name: 1, mayor: 1})
```

	{} mayor	{} name
1	{"name": "Michael Bloomberg", "party": "I"}	New York

3) *Сформировать запрос, который возвращает список беспартийных мэров (`party` отсутствует). Вывести только название города и информацию о мэре.*

```
db.towns.find({"mayor.party": {$exists: false}}, {_id: 0, name: 1, mayor: 1})
```

	{} mayor	{} name
1	{"name": "Jim Wehrle"}	Punxsutawney

Практическое задание 8.2.2:

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

```
fn = function() {return this.gender == 'm';}
```

	{} result
1	js function

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*

```
cursor.forEach(function(fn){print(fn.name);})
```

	{} 0
1	Dunx
2	Horny

Практическое задание 8.2.3:

Вывести количество самок единорогов весом от полутонны до 600 кг.

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

	{} result
1	2

Практическое задание 8.2.4:

Вывести список предпочтений.

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

	{} result
1	apple
2	carrot
3	chocolate
4	energon
5	grape
6	lemon
7	papaya
8	redbull
9	strawberry
10	sugar
11	watermelon

Практическое задание 8.2.5:

Посчитать количество особей единорогов обоих полов.

```
db.unicorns.aggregate([{$group: {_id: "$gender", count: {$sum: 1}}}])
```

	{} _id	{} count
1	f	5
2	m	7

Практическое задание 8.2.6:

1. *Выполнить команду:*

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

2. *Проверить содержимое коллекции* `unicorns`.

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

	{} _id	{} gender	{} loves	{} name	{} weight
1	62895f614d20cb595a2b5cd6	m	["grape"]	Barny	340

Практическое задание 8.2.7:

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

```
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"})
```

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	6289125be8c85b7c0a593c82	f	["strawberry", "lemon"]	Ayna	51	800

Практическое задание 8.2.8:

1. *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"})
```

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	628a844da8bae21b3ec23c5c	m	["redbull"]	Raleigh	2	421

Практическое задание 8.2.9:

До

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	628a844da8bae21b3ec23c55	m	["carrot", "papaya"]	Horny	63	600
2	628a844da8bae21b3ec23c57	m	["energon", "redbull"]	Unicrom	182	984
3	628a844da8bae21b3ec23c58	m	["apple"]	Roooooodles	99	575
4	628a844da8bae21b3ec23c5b	m	["grape", "lemon"]	Kenny	39	690
5	628a844da8bae21b3ec23c5c	m	["redbull"]	Raleigh	2	421
6	628a844da8bae21b3ec23c5e	m	["apple", "watermelon"]	Pilot	54	650
7	628a844da8bae21b3ec23c60	m	["grape", "watermelon"]	Dunx	165	704
8	628a8468a8bae21b3ec23c62	m	["grape"]	Barny	<unset>	340

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

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

2. *Проверить содержимое коллекции* `unicorns`.

	{} _id	{} gender	{} loves	{} name	{} vampires	{} weight
1	628a844da8bae21b3ec23c55	m	["carrot", "papaya"]	Horny	68	600
2	628a844da8bae21b3ec23c57	m	["energon", "redbull"]	Unicrom	187	984
3	628a844da8bae21b3ec23c58	m	["apple"]	Roooooodles	104	575
4	628a844da8bae21b3ec23c5b	m	["grape", "lemon"]	Kenny	44	690
5	628a844da8bae21b3ec23c5c	m	["redbull"]	Raleigh	7	421
6	628a844da8bae21b3ec23c5e	m	["apple", "watermelon"]	Pilot	59	650
7	628a844da8bae21b3ec23c60	m	["grape", "watermelon"]	Dunx	170	704
8	628a8468a8bae21b3ec23c62	m	["grape"]	Barny	5	340

Практическое задание 8.2.10:

До

	{} _id	{} famous_for	{} last_sensus	{} mayor	{} name	{} populatiuon
1	62893c32e8c85b7c0a593c93	["beer", "food"]	2009-07-20T00:00:00.000Z	{"name": "Sam Adams", "party": "D"}	Portland	528000

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`.

```
db.towns.find({name: "Portland"})
```

	_id	famous_for	last_sensus	mayor	name	populatiuon
1	62893c32e8c85b7c0a593c93	["beer", "food"]	2009-07-20T00:00:00.000Z	{"name": "Sam Adams"}	Portland	528000

Практическое задание 8.2.11:

До

	_id	gender	loves	name	vampires	weight
1	6289125ce8c85b7c0a593c86	m	["apple", "watermelon"]	Pilot	59	650

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.`

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

	_id	gender	loves	name	vampires	weight
1	6289125ce8c85b7c0a593c86	m	["apple", "watermelon", "chocolate"]	Pilot	59	650

Практическое задание 8.2.12:

До

	_id	gender	loves	name	vampires	weight
1	62891259e8c85b7c0a593c7e	f	["carrot", "grape"]	Aurora	43	450

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.`

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

	_id	gender	loves	name	vampires	weight
1	62891259e8c85b7c0a593c7e	f	["carrot", "grape", "sugar", "lemon"]	Aurora	43	450

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.

```
db.towns.deleteOne({"mayor.party": {$exists: false}})
```

	{} acknowledged	{} deletedCount
1	true	1

3) Проверьте содержание коллекции.

```
db.towns.find()
```

	{} _id	{} famous_for	{} last_sensus	{} mayor	{} name	{} populatiuon
1	62893c26e8c85b7c0a593c91	["status of liberty", "food"]	2009-07-31T00:00:00.000Z	{"name": "Michael Bloomberg", "party": "I"}	New York	22200000
2	62893c32e8c85b7c0a593c93	["beer", "food"]	2009-07-20T00:00:00.000Z	{"name": "Sam Adams", "party": "D"}	Portland	528000

4) Очистите коллекцию.

```
db.towns.deleteMany({})
```

	{} acknowledged	{} deletedCount
1	true	2

5) Просмотрите список доступных коллекций.

```
show collections
```

```
towns
unicorns
```

Практическое задание 8.3.1:

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

```
db.areas.insertMany([{_id: "cove", name: "Oceanus Cove", desc: "Located in south-central part of Azsuna"},
{_id: "wellspring", name: "Nor'Danil Wellspring", desc: "Located northeast across the waters from the island of Faronaar"}])
```

	{} _id	{} desc	{} name
1	cove	Located in south-central part of Azsuna	Oceanus Cove
2	wellspring	Located northeast across the waters from the island of Farona…	Nor'Danil Wellspring

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.*

	{} _id	{} areas	{} gender	{} loves	{} name	{} vampires	{} weight
1	628a29a9e56d311699b782e1	{ "$ref" : "areas", "$id" : "cove" }	m	["carrot", "papaya"]	Horny	63	600
2	628a29a9e56d311699b782e2	{ "$ref" : "areas", "$id" : "wellspring" }	f	["carrot", "grape"]	Aurora	43	450
3	628a29a9e56d311699b782e3	<unset>	m	["energon", "redbull"]	Unicrom	182	984
4	628a29a9e56d311699b782e4	<unset>	m	["apple"]	Roooooodles	99	575
5	628a29a9e56d311699b782e5	<unset>	f	["apple", "carrot", "chocolate"]	Solnara	80	550
6	628a29a9e56d311699b782e6	<unset>	f	["strawberry", "lemon"]	Ayna	40	733
7	628a29a9e56d311699b782e7	<unset>	m	["grape", "lemon"]	Kenny	39	690
8	628a29a9e56d311699b782e8	<unset>	m	["apple", "sugar"]	Raleigh	2	421
9	628a29a9e56d311699b782e9	<unset>	f	["apple", "watermelon"]	Leia	33	601
10	628a29a9e56d311699b782ea	<unset>	m	["apple", "watermelon"]	Pilot	54	650
11	628a29a9e56d311699b782eb	<unset>	f	["grape", "carrot"]	Nimue	<unset>	540
12	628a29a9e56d311699b782ec	<unset>	m	["grape", "watermelon"]	Dunx	165	704

Практическое задание 8.3.2:

1. *Проверьте, можно ли задать для коллекции* `unicorns` *индекс для ключа* `name` *с флагом* `unique`.

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

	{} result
1	name_1

Практическое задание 8.3.3:

1) *Получите информацию о всех индексах коллекции* `unicorns` .

```
db.unicorns.getIndexes()
```

	{} key	{} name	{} v	{} unique
1	{"_id": new NumberInt("1")}	_id_	2	<unset>
2	{"name": new NumberInt("1")}	name_1	2	true

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})}
```

2) *Выберите последних четыре документа.*

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

	{} _id	{} value
1	628a51c3a8bae21b3ec23c4f	99999
2	628a51c3a8bae21b3ec23c4e	99998
3	628a51c3a8bae21b3ec23c4d	99997
4	628a51c3a8bae21b3ec23c4c	99996

```
lab> db.numbers.find().sort({$natural: -1}).limit(4)
[2022-05-22 20:05:45] 4 rows retrieved starting from 1 in 19 s 500 ms (execution: 231 ms, fetching: 19 s 269 ms)
```

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`*.*

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

	{} result
1	value_1

5) *Получите информацию о всех индексах коллекции* `numbers`*.*

```
db.numbers.getIndexes()
```

	{} key	{} name	{} v
1	{"_id": new NumberInt("1")}	_id_	2
2	{"value": new NumberInt("1")}	value_1	2

6) Выполните запрос 2.

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

	{} _id	{} value
1	628a51c3a8bae21b3ec23c4f	99999
2	628a51c3a8bae21b3ec23c4e	99998
3	628a51c3a8bae21b3ec23c4d	99997
4	628a51c3a8bae21b3ec23c4c	99996

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

```
lab> 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)
```

```
lab> 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.