**ДЗ по теме 10 «NoSQL»**

1. *В базе данных Redis подберите коллекцию для подсчета посещений  
с определенных IP-адресов.*

/\* создаем коллекцию с теми IP, которые разрешены в самом Redis \*/

> [**SSAD id: open**](http://try.redis.io/#run) **192.168.1.4 192.168.1.5 8.8.4.4**

**(integer) 3**

> [**SMEMBERS ip: open**](http://try.redis.io/#run) **--** посмотрим коллекцию

**1)"8.8.4.4"**

**2)"192.168.1.4"**

**3)"192.168.1.5"**

/\* проверяем, есть ли у нас некий IP в созданном наборе\*/

> [**SISMEMBERS ip: open**](http://try.redis.io/#run) **192.168.1.4**

**(integer) 1**

> [**SISMEMBERS ip: open**](http://try.redis.io/#run) **8.8.4.5**

**(integer) 0**

/\* Если IP есть в наборе, то делаем инкримент \*/

> [**INCR ip:**](http://try.redis.io/#run) **8.8.4.4**

**(integer) 1**

> [**GET ip:**](http://try.redis.io/#run) **8.8.4.4**

**"1"**

> [**INCR ip:**](http://try.redis.io/#run) **8.8.4.4**

**(integer) 2**

2. *При помощи базы данных Redis решите задачу поиска имени пользователя  
по электронному адресу и наоборот, поиск электронного адреса пользователя по его имени.*

-- Создадим хэш-таблицу, задав все пары «ключ – значение» командой HMSET

> [**HMSET user usermail "name" username "email**](http://try.redis.io/#run)**"**

/\* user – ключ хэша (пользователь), usermail – ключ 1 (эл. адрес пользователя), “name” – значение ключа 1 (имя пользователя), username **–** ключ 2   
(имя пользователя), “email” – значение ключа 2 (эл. адрес пользователя) \*/

> [**HGET user usermail**](http://try.redis.io/#run)-- поиск имени по почте

**"name"** – искомое имя

> [**HGET user username**](http://try.redis.io/#run) -- поиск почты по имени

**"email"** – искомая почта

/\* Наглядный пример (можно нажать ctrl и щелкать ссылки, после чего скопировать команды в окне демонстрационной работы Redis try.redis.io) \*/

> [**HMSET user pochta@mail.ru "Artyom\_Artyomov" Artyom\_Artyomov "pochta@mail.ru"**](http://try.redis.io/#run)

**OK**

> [**HGET user pochta@mail.ru**](http://try.redis.io/#run)

**"Artyom\_Artyomov"**

> [**HGET user Artyom\_Artyomov**](http://try.redis.io/#run)

**"pochta@mail.ru"**

-- Вопрос можно решить аналогичным образом через две хэш-таблицы

> [**HSET user pochta@mail.ru "Artyom\_Artyomov**](http://try.redis.io/#run)**"**

**(integer) 1**

> [**HSET user Artyom\_Artyomov "pochta@mail.ru**](http://try.redis.io/#run)**"**

**(integer) 1**

> [**HGET user pochta@mail.ru**](http://try.redis.io/#run)

**"Artyom\_Artyomov"**

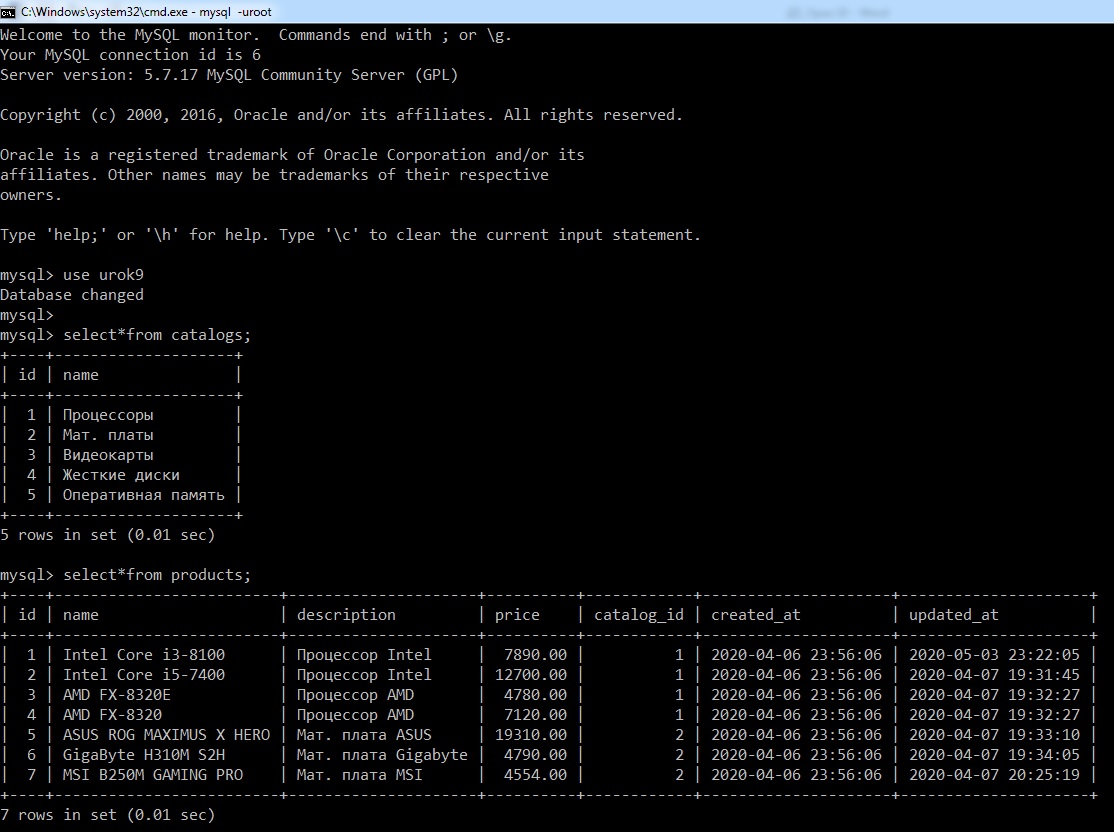
> [**HGET user Artyom\_Artyomov**](http://try.redis.io/#run)

**"pochta@mail.ru"**

*3. Организуйте хранение категорий и товарных позиций учебной базы данных shop в СУБД MongoDB.*

*/\** То есть требуется создать в MongoDB таблицы shop.catalogs и shop.products.

Содержимое у этих таблиц у меня следующее \*/



/\* Решение задачи представлено ниже \*/

use shop -- создаем БД shop

/\* Создаем и наполняем документами коллекцию catalogs \*/

catalogs = ([{\_id: 1, name: 'Процессоры'},{\_id: 2, name: 'Мат. платы'},   
{\_id: 3, name: 'Видеокарты'}, {\_id: 4, name: 'Жесткие диски'},  
{\_id: 5, name: 'Оперативная память'}])

db.shop.insert(‘catalogs’) -- добавляем коллекцию catalogs в БД shop

/\* Создаем и наполняем документами коллекцию products \*/

products = ([{\_id: 1, name: ’Intel Core i3-8100’, description: ‘Процессор Intel’,

price: 7890.00, catalog\_id: 1, created\_at: new Date(), updated\_at: new Date()},

{\_id: 2, name: ’Intel Core i5-7400’, description: ‘Процессор Intel’,

price: 12700.00, catalog\_id: 1, created\_at: new Date(), updated\_at: new Date()},

{\_id: 3, name: ’AMD FX-8320E’, description: ‘Процессор AMD’, price: 4780.00,

catalog\_id: 1, created\_at: new Date(), updated\_at: new Date()},

{\_id: 4, name: ’AMD FX-8320’, description: ‘Процессор AMD’, price: 7120.00,

catalog\_id: 1, created\_at: new Date(), updated\_at: new Date()},

{\_id: 5, name: ’ASUS ROG MAXIMUS X HERO’, description: ‘Мат. плата ASUS’,

price: 19310.00, catalog\_id: 2, created\_at: new Date(), updated\_at: new Date()},

{\_id: 6, name: ‘GigaByte H310M S2H’, description: ‘Мат. плата GigaByte’,  
 price: 4790.00, catalog\_id: 2, created\_at: new Date(), updated\_at: new Date()},

{\_id: 7, name: ‘MSI B250M GAMING PRO’, description: ‘Мат. Плата MSI’,  
 price: 4554.00, catalog\_id: 2, created\_at: new Date(), updated\_at: new Date()}])

db.shop1.insert(‘products’) -- добавляем коллекцию products в БД shop1

-- shop1 создается автоматически

/\* Если запихаем в таком виде catalogs и products в одну БД shop, будет ошибка из-за дупликатов “\_id”. Делать ссылки (по типу механизма связи первичный ключ – внешний ключ в MySQL) я не стал. Нас просили воссоздать shop.catalogs и shop.products в MongoDB – я создал \*/

/\* Смотрим, что у нас получилось \*/

db.shop.find().pretty() – содержимое БД shop c коллекцией catalogs

db.shop1.find().pretty() – содержимое БД shop1 c коллекцией products

/\* В целом, можно решить разными способами, например, методом insertMany(). Я выбрал способ, который не рассматривался в лекции /\*