

Практическое задание по теме «Оптимизация запросов»

Задание 1. Выполнено. Создал таблицу logs типа Archive. При каждом создании записи в таблицах users, catalogs и products в таблицу logs помещается время и дата создания записи, название таблицы, идентификатор первичного ключа и содержимое поля name.

```
USE shop;
DROP TABLE IF EXISTS logs;
CREATE TABLE logs (
    created_at DATETIME,
    table_name VARCHAR(255),
    identificator BIGINT,
    name_field_content VARCHAR(255)
) ENGINE=Archive;

DELIMITER //
DROP TRIGGER IF EXISTS log_products_insert //
CREATE TRIGGER log_products_insert AFTER INSERT ON products
FOR EACH ROW BEGIN
    INSERT INTO shop.logs VALUES(NOW(), 'products', NEW.id, NEW.name);
END//

DROP TRIGGER IF EXISTS log_catalogs_insert //
CREATE TRIGGER log_catalogs_insert AFTER INSERT ON catalogs
FOR EACH ROW BEGIN
    INSERT INTO shop.logs VALUES(NOW(), 'catalogs', NEW.id, NEW.name);
END//

DROP TRIGGER IF EXISTS log_users_insert //
CREATE TRIGGER log_users_insert AFTER INSERT ON users
FOR EACH ROW BEGIN
    INSERT INTO shop.logs VALUES(NOW(), 'users', NEW.id, NEW.name);
END//

DELIMITER ;

INSERT INTO products (id, name, catalog_id) VALUES(10, 'Apricot', 1);
INSERT INTO catalogs (id, name) VALUES(NULL, 'grocery');
INSERT INTO users VALUES(NULL, 'FedorFedotov');

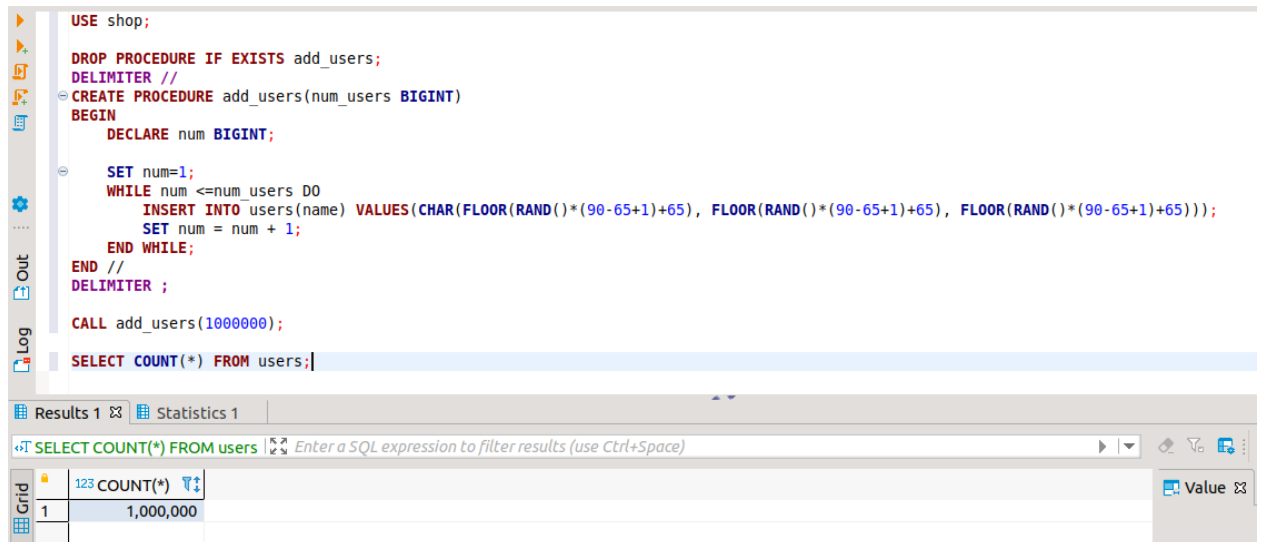
SELECT * FROM logs;
```

logs 1 Statistics 1

SELECT * FROM logs

	created_at	table_name	identificator	name_field_content
1	2021-07-02 06:20:54	products	10	Apricot
2	2021-07-02 06:20:54	catalogs	4	grocery
3	2021-07-02 06:20:54	users	4	FedorFedotov

Задание 2. Выполнено. Создал SQL-запрос, который помещает в таблицу users миллион записей. Запрос выполнялся примерно за 15 минут (использую ubuntu на виртуальной машине). При этом стоит отметить, что таблица logs типа Archive, в которую по триггеру записывалось больше данных при каждой вставке значений в таблицу users занимает значительно (более чем в пять раз) меньше места по сравнению с таблицей users.



```
USE shop;

DROP PROCEDURE IF EXISTS add_users;
DELIMITER //
CREATE PROCEDURE add_users(num_users BIGINT)
BEGIN
    DECLARE num BIGINT;

    SET num=1;
    WHILE num <=num_users DO
        INSERT INTO users(name) VALUES(CHAR(FLOOR(RAND()*(90-65+1)+65), FLOOR(RAND()*(90-65+1)+65), FLOOR(RAND()*(90-65+1)+65)));
        SET num = num + 1;
    END WHILE;
END //
DELIMITER ;

CALL add_users(1000000);

SELECT COUNT(*) FROM users;
```

Results 1 Statistics 1

SELECT COUNT(*) FROM users

Grid	123 COUNT(*)
1	1,000,000

Value

Для ускорения выполнил следующее (в соответствии с рекомендациями официальной документации MySQL):

1. Задал новое значение системной переменной innodb_buffer_pool_size=1024M.
2. Отключил автокоммит (SET autocommit=0; ... COMMIT;).
3. Сделал вставку по несколько строк в одном INSERT.

Данные действия позволили существенно ускорить вставку данных (время, требуемое для выполнения вставки 1 млн. записей, сократилось с 15 минут до 2 минут).

Практическое задание по теме «NoSQL»

Задание 1. Выполнено. В базе данных Redis подобрал коллекцию для подсчета посещений с определенных IP-адресов. Наиболее подходящим типом данных в базе данных Redis для хранения посещений с определенных IP-адресов является хэш-таблица.

```
andrey@andrey-virtual-machine: ~/Desktop
127.0.0.1:6379> hset ip_addr 192.168.0.1 5
(integer) 1
127.0.0.1:6379> hset ip_addr 192.168.0.10 25
(integer) 1
127.0.0.1:6379> hset ip_addr 192.168.0.100 45
(integer) 1
127.0.0.1:6379> hgetall ip_addr
1) "192.168.0.1"
2) "5"
3) "192.168.0.10"
4) "25"
5) "192.168.0.100"
6) "45"
127.0.0.1:6379> 
```

Задание 2. Выполнено. При помощи базы данных Redis решил задачу поиска имени пользователя по электронному адресу и наоборот, поиск электронного адреса пользователя по его имени. Для этого создал две таблицы, поскольку Redis предназначен для поиска по ключам.

```
andrey@andrey-virtual-machine: ~/Desktop
127.0.0.1:6379> hset users Andrey Andrushka@mail.ru
(integer) 1
127.0.0.1:6379> hset users Sergey Serg@ya.ru
(integer) 1
127.0.0.1:6379> hset users Vladimir Volodya@gmail.ru
(integer) 1
127.0.0.1:6379> hset emails Andrushka@mail.ru Andrey
(integer) 1
127.0.0.1:6379> hset emails Serg@ya.ru Sergey
(integer) 1
127.0.0.1:6379> hset emails Volodya@gmail.ru Vladimir
(integer) 1
127.0.0.1:6379> hget users Andrey
"Andrushka@mail.ru"
127.0.0.1:6379> hget emails Volodya@gmail.ru
"Vladimir"
127.0.0.1:6379> 
```

Задание 3. Выполнено. Организовал хранение категорий и товарных позиций учебной базы данных shop в СУБД MongoDB. Реализацию сделал с внешним ключом.

```
> use shop
switched to db shop
> db.shop.catalogs.insert({_id: 1, name: 'fruits'}, {_id: 2, name: 'vegetables'})
WriteResult({ "nInserted" : 1 })
> db.shop.products.insert({name: 'apple', catalog: 1, count: 10}, {name: 'tomato', catalog: 2, count:
  100})
WriteResult({ "nInserted" : 1 })
> db.shop.catalogs.find()
{ "_id" : 1, "name" : "fruits" }
> db.shop.products.find()
{ " _id" : ObjectId("60df0888f2db191a713223d8"), "name" : "apple", "catalog" : 1, "count" : 10 }
> █
```