Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №5 «Процедуры, функции, триггеры в PostgreSQL»

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Зотов М.Д.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

Цель работы

Овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

Практическое задание

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).

2. Создать авторский триггер по варианту индивидуального задания.

Создайте хранимые процедуры/функции

1. Вывести сведения о заказах заданного официанта на заданную дату.

```
CREATE OR REPLACE FUNCTION get_orders_by_waiter_and_date (
    arg_id INTEGER,
    arg_date DATE
)
RETURNS TABLE (
    order_id INTEGER,
    "date" DATE,
    "name" VARCHAR
) AS $$
BEGIN
    RETURN QUERY
    SELECT
            o.order_code AS order_id,
            o.date AS "date",
            d.name AS "name"
    FROM restaurant_scheme.order o
    JOIN restaurant_scheme.dish d ON d.dish_code = o.dish_code
    JOIN restaurant_scheme.table tb ON o.number_of_table = tb.number_of_table
    JOIN restaurant_scheme.employee e ON e.personal_number = tb.waiter_id
    WHERE o.date = arg_date AND e.personal_number = arg_id;
END;
$$ LANGUAGE plpgsql;
```

```
restaurant=# CREATE OR REPLACE FUNCTION get_orders_by_waiter_and_date (
restaurant(# arg_id INTEGER,
restaurant(# arg_date DATE
restaurant(# )
restaurant-# RETURNS TABLE (
restaurant(# order_id INTEGER,
restaurant(# "date" DATE,
restaurant(# "name" VARCHAR
restaurant(# ) AS $$
restaurant$# BEGIN
restaurant$# RETURN QUERY
restaurant$# SELECT
restaurant$# o.order_code AS order_id,
restaurant$# o.date AS "date",
restaurant$# d.name AS "name"
restaurant$# FROM restaurant_scheme.order o
restaurant$# JOIN restaurant_scheme.dish d ON d.dish_code = o.dish_code
restaurant$# JOIN restaurant_scheme.table tb ON o.number_of_table = tb.number_of_table
restaurant$# JOIN restaurant_scheme.employee e ON e.personal_number = tb.waiter_id
restaurant$# WHERE o.date = arg_date AND e.personal_number = arg_id;
restaurant$# END;
restaurant$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

```
СТРОКА 1: SELECT * FROM get_orders_by_waiter_and_date;
                      ^
restaurant=# SELECT * FROM get_orders_by_waiter_and_date(4, '2024-02-23');
 order_id |    date    |       name
----------+------------+------------------
        5 | 2024-02-23 | Тирамису
        4 | 2024-02-23 | Лосось с терияки
(2 строки)
```

2. Выполнить расчет стоимости заданного заказа.

```
CREATE OR REPLACE FUNCTION get_order_price(
    arg_id INTEGER
)
RETURNS INTEGER AS $$
DECLARE price INTEGER:=0;
BEGIN
    SELECT SUM(d.price) INTO price
    FROM restaurant_scheme.order o
    JOIN restaurant_scheme.dish d ON d.dish_code = o.dish_code
    WHERE arg_id = o.order_code;
    RETURN price;
END;
$$ LANGUAGE plpgsql;
```

```
restaurant=# CREATE OR REPLACE FUNCTION get_order_price(
restaurant(# arg_id INTEGER
restaurant(# )
restaurant-# RETURNS INTEGER AS $$
restaurant$# DECLARE price INTEGER:=0;
restaurant$# BEGIN
restaurant$# SELECT SUM(d.price) INTO price
restaurant$# FROM restaurant_scheme.order o
restaurant$# JOIN restaurant_scheme.dish d ON d.dish_code = o.dish_code
restaurant$# WHERE arg_id = o.order_code;
restaurant$# RETURN price;
restaurant$# END;
restaurant$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

```
restaurant=# SELECT * FROM get_order_price(6);
 get_order_price
-----------------
             650
(1 строка)
```

3. **Повышения оклада заданного сотрудника на 30 % при повышении его категории.**

```
CREATE OR REPLACE PROCEDURE employee_salary_increase(
        arg_personal_number INTEGER
) AS
$$
DECLARE cat INTEGER;
BEGIN
        SELECT category INTO cat FROM restaurant_scheme.employee
        WHERE personal_number = arg_personal_number;
        IF cat < 5 THEN
                UPDATE restaurant_scheme.employee
                SET
                        salary = salary * 1.3,
                        category = category + 1
                WHERE personal_number = arg_personal_number;
        END IF;
END;
$$ LANGUAGE plpgsql;
```

```
restaurant=# CREATE OR REPLACE PROCEDURE employee_salary_increase(
restaurant(# arg_personal_number INTEGER
restaurant(# ) AS
restaurant-# $$
restaurant$# BEGIN
restaurant$# UPDATE restaurant_scheme.employee
restaurant$# SET
restaurant$# salary = salary * 1.3,
restaurant$# category = category + 1
restaurant$# WHERE personal_number = arg_personal_number;
restaurant$# END;
restaurant$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
```

ДО:

```
restaurant=# SELECT * FROM restaurant_scheme.employee WHERE personal_number = 4;
 personal_number |   full_name    | passport_data | job_id | category | salary
-----------------+----------------+---------------+--------+----------+--------
               4 | Анна Смирнова | 23 43 825322  |      2 |        3 |  35000
(1 строка)
```

ПОСЛЕ:

```
restaurant=# CALL employee_salary_increase(4);
CALL
restaurant=# SELECT * FROM restaurant_scheme.employee WHERE personal_number = 4;
 personal_number |   full_name    | passport_data | job_id | category | salary
-----------------+----------------+---------------+--------+----------+--------
               4 | Анна Смирнова | 23 43 825322  |      2 |        4 |  45500
(1 строка)
```

Триггеры

Триггер на обновление цены блюда, при изменении его состава.

```
CREATE OR REPLACE FUNCTION calculate_price_after_composition_change()
RETURNS TRIGGER AS
$$
DECLARE
   ingredient_price INTEGER;
BEGIN
   IF TG_OP = 'INSERT' THEN
      SELECT i.price * NEW.count_of_ingredient / 1000 INTO ingredient_price
      FROM restaurant_scheme.ingredient i
      WHERE i.ingredient_code = NEW.ingredient_code;

      UPDATE restaurant_scheme.dish
      SET price = price + ingredient_price * 1.40
      WHERE dish_code = NEW.dish_code;

      RETURN NEW;
   ELSIF TG_OP = 'UPDATE' THEN
      SELECT
         CASE
            WHEN NEW.count_of_ingredient > OLD.count_of_ingredient THEN
               i.price * (NEW.count_of_ingredient - OLD.count_of_ingredient) / 1000
            ELSE
               - i.price * (OLD.count_of_ingredient - NEW.count_of_ingredient) / 1000
         END
      INTO ingredient_price
      FROM restaurant_scheme.ingredient i
      WHERE i.ingredient_code = NEW.ingredient_code;

      UPDATE restaurant_scheme.dish
      SET price = price + ingredient_price * 1.40
      WHERE dish_code = NEW.dish_code;
```

```
    RETURN NEW;
  ELSIF TG_OP = 'DELETE' THEN
    SELECT i.price * OLD.count_of_ingredient / 1000 INTO ingredient_price
    FROM restaurant_scheme.ingredient i
    WHERE i.ingredient_code = OLD.ingredient_code;

    UPDATE restaurant_scheme.dish
    SET price = price - ingredient_price * 1.40
    WHERE dish_code = OLD.dish_code;

    RETURN OLD;
  END IF;
END;
$$ LANGUAGE plpgsql;
```

```
restaurant=# CREATE OR REPLACE FUNCTION calculate_price_after_composition_change()
restaurant-# RETURNS TRIGGER AS
restaurant-# $$
restaurant$# DECLARE sum_ingredients INTEGER;
restaurant$# BEGIN
restaurant$# IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
restaurant$# SELECT SUM(i.price * cd.count_of_ingredient / 1000) INTO sum_ingredients
restaurant$# FROM restaurant_scheme.ingredient i
restaurant$# JOIN restaurant_scheme.composition_of_the_dish cd ON i.ingredient_code = cd.ingredient_code
restaurant$# JOIN restaurant_scheme.dish d ON d.dish_code = cd.dish_code
restaurant$# WHERE d.dish_code = NEW.dish_code;
restaurant$#
restaurant$# UPDATE restaurant_scheme.dish
restaurant$# SET price = sum_ingredients * 1.40
restaurant$# WHERE dish_code = NEW.dish_code;
restaurant$# RETURN NEW;
restaurant$# ELSIF TG_OP = 'DELETE' THEN
restaurant$# SELECT SUM(i.price * cd.count_of_ingredient / 1000) INTO sum_ingredients
restaurant$# FROM restaurant_scheme.ingredient i
restaurant$# JOIN restaurant_scheme.composition_of_the_dish cd ON i.ingredient_code = cd.ingredient_code
restaurant$# JOIN restaurant_scheme.dish d ON d.dish_code = cd.dish_code
restaurant$# WHERE d.dish_code = OLD.dish_code;
restaurant$#
restaurant$# UPDATE restaurant_scheme.dish
restaurant$# SET price = sum_ingredients * 1.40
restaurant$# WHERE dish_code = OLD.dish_code;
restaurant$# RETURN NEW;
restaurant$# END IF;
restaurant$# END;
restaurant$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

```
CREATE TRIGGER update_dish_price
AFTER INSERT OR UPDATE OR DELETE ON restaurant_scheme.composition_of_the_dish
FOR EACH ROW
```

EXECUTE FUNCTION calculate_price_after_composition_change();

```
restaurant=# CREATE TRIGGER update_dish_price
restaurant-# AFTER INSERT OR UPDATE OR DELETE ON restaurant_scheme.composition_of_the_dish
restaurant-# FOR EACH ROW
restaurant-# EXECUTE FUNCTION calculate_price_after_composition_change();
CREATE TRIGGER
```

Тест:

INSERT:

```
restaurant=# SELECT * FROM restaurant_scheme.dish
restaurant-# WHERE dish_code = 16;
 dish_code |     name      | volume | price
-----------+---------------+--------+-------
        16 | Курица гриль |    500 |    91
(1 строка)
```

```
restaurant=# INSERT INTO restaurant_scheme.composition_of_the_dish
restaurant-# (dish_code, ingredient_code, volume_of_ingredients, count_of_ingredient)
restaurant-# VALUES (16, 11, 'г', 100);
INSERT 0 1
```

```
restaurant=# SELECT * FROM restaurant_scheme.dish
restaurant-# WHERE dish_code = 16;
 dish_code |     name      | volume | price
-----------+---------------+--------+-------
        16 | Курица гриль |    500 |  96.6
(1 строка)
```

UPDATE:

```
restaurant=# UPDATE restaurant_scheme.composition_of_the_dish
restaurant-# SET count_of_ingredient = 200
restaurant-# WHERE composition_of_the_dish_id = 49;
UPDATE 1
```

```
restaurant=# SELECT * FROM restaurant_scheme.dish
restaurant-# WHERE dish_code = 16;
 dish_code |     name      | volume | price
-----------+---------------+--------+-------
        16 | Курица гриль |    500 | 102.2
(1 строка)
```

DELETE:

```
restaurant=# DELETE FROM restaurant_scheme.composition_of_the_dish
restaurant-# WHERE composition_of_the_dish_id = 52;
DELETE 1
```

```
restaurant=# SELECT * FROM restaurant_scheme.dish
restaurant-# WHERE dish_code = 16;
 dish_code |     name      | volume | price
-----------+---------------+--------+-------
        16 | Курица гриль |    500 |    91
(1 строка)
```

Вывод

Овладел практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.