Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
“Национальный исследовательский университет ИТМО”

Факультет инфокоммуникационных технологий

ЛАБОРАТОРНАЯ РАБОТА №5

Процедуры, функции, триггеры в PostgreSQL
по дисциплине:
«Проектирование и реализация баз данных»

Выполнил:
Бабичев Леонид Анатольевич
Группа K32422
Проверила:
Говорова Марина Михайловна

Санкт-Петербург
2023

ЦЕЛЬ РАБОТЫ:

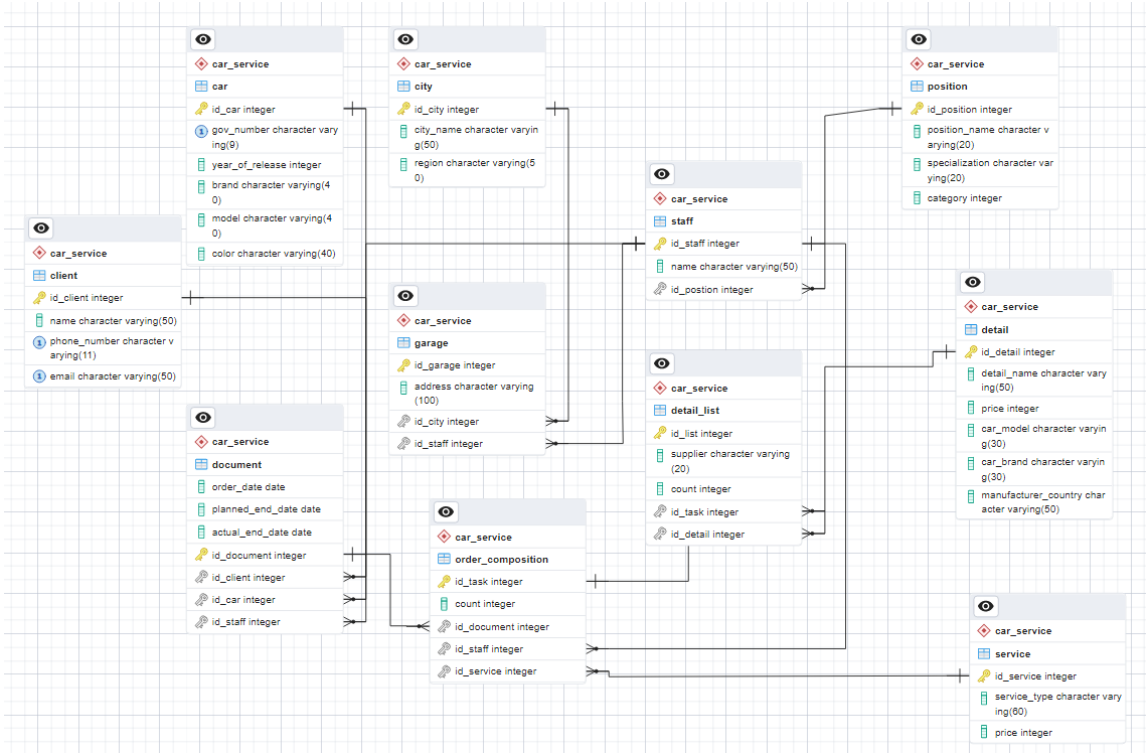
Овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

Оборудование: компьютерный класс.
Программное обеспечение: СУБД PostgreSQL, SQL Shell (psql).

Вариант 1
Практическое задание:

I. Создать процедуры/функции согласно индивидуальному заданию и (согласно индивидуальному заданию, часть 4).
II. Создать триггер для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL (согласно индивидуальному заданию, часть 5). Допустимо создать универсальный триггер или отдельные триггеры на логирование действий.

База данных:



Выполнение:

Задание 1. Создайте хранимые процедуры.

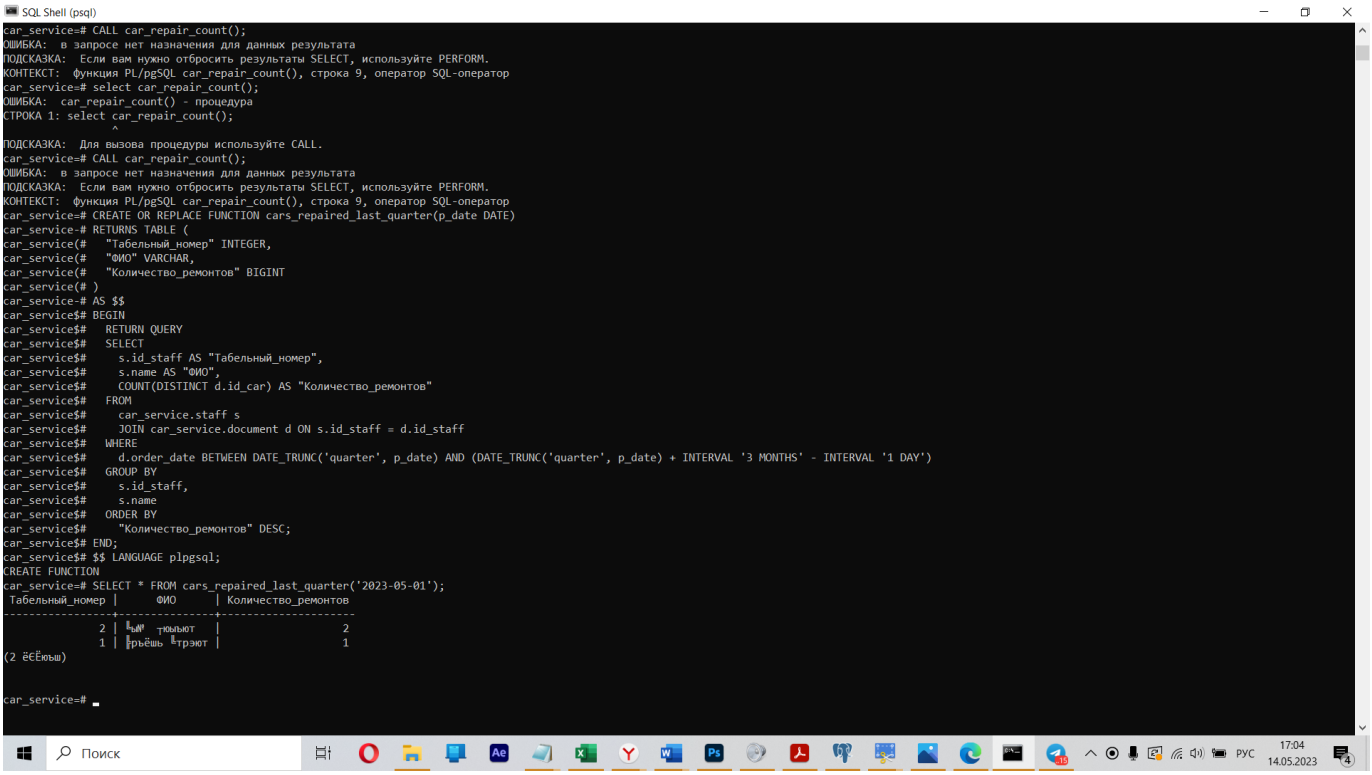
1) Повышения цены деталей на 10 %

```
car_service=# select * from car_service.detail;
 id_detail |    detail_name    | price |     car_model     |  car_brand   | manufacturer_country
-----------+-------------------+-------+-------------------+--------------+----------------------
         4 | Фара              | 15000 | V90 Cross Country | Volvo        | Швеция
         3 | Тормозная колодка | 50000 | Valkyrie          | Aston Martin | Великобритания
         1 | Аккумулятор       | 30000 | Impreza Sport     | Subaru       | Япония
(3 строки)


car_service=# CREATE OR REPLACE PROCEDURE public.increase_subaru_parts_price(increase_percent NUMERIC)
car_service-# LANGUAGE plpgsql
car_service-# AS $$
car_service$# BEGIN
car_service$# UPDATE car_service.detail
car_service$# SET price = price * (1 + increase_percent/100)
car_service$# WHERE car_brand = 'Subaru';
car_service$# END;
car_service$# $$;
CREATE PROCEDURE
car_service=# CALL public.increase_subaru_parts_price(5.0);
CALL
car_service=# select * from car_service.detail;
 id_detail |    detail_name    | price |     car_model     |  car_brand   | manufacturer_country
-----------+-------------------+-------+-------------------+--------------+----------------------
         4 | Фара              | 15000 | V90 Cross Country | Volvo        | Швеция
         3 | Тормозная колодка | 50000 | Valkyrie          | Aston Martin | Великобритания
         1 | Аккумулятор       | 31500 | Impreza Sport     | Subaru       | Япония
(3 строки)
```

2) Для повышения разряда тех мастеров, которые отремонтировали больше 3 автомобилей.

```
car_service=# select * from car_service.position;
 id_position | position_name |  specialization   | category
-------------+---------------+-------------------+----------
           1 | Менеджер      | закупка запчастей |        1
           3 | Парковщик     | парковщик авто    |        1
           2 | Мойщик        | мойщик салона     |        2
(3 строки)


car_service=# CREATE OR REPLACE PROCEDURE public.increase_cat(min_cars INTEGER)
car_service-# LANGUAGE plpgsql
car_service-# AS $$
car_service$# BEGIN
car_service$#     UPDATE car_service.position
car_service$#     SET category = category + 1
car_service$#     WHERE id_position IN (
car_service$#         SELECT position.id_position
car_service$#         FROM car_service.staff s
car_service$#         JOIN car_service.position ON position.id_position = s.id_position
car_service$#         JOIN car_service.document d ON s.id_staff = d.id_staff
car_service$#         GROUP BY position.id_position
car_service$#         HAVING COUNT(DISTINCT id_car) >= min_cars
car_service$#     );
car_service$# END;
car_service$# $$;
CREATE PROCEDURE
car_service=# CALL public.increase_cat(3);
CALL
car_service=# select * from car_service.position;
 id_position | position_name |  specialization   | category
-------------+---------------+-------------------+----------
           1 | Менеджер      | закупка запчастей |        1
           3 | Парковщик     | парковщик авто    |        1
           2 | Мойщик        | мойщик салона     |        3
(3 строки)
```

3) Сколько автомобилей отремонтировал каждый механик за истекший квартал.



Задание 2. Создать триггер для логирования событий вставки, удаления, редактирования

Создана таблица logs

```
car_service=# CREATE TABLE car_service.log_changes (id SERIAL PRIMARY KEY,
car_service(#  TABLE_NAME TEXT NOT NULL,
car_service(#  change_type TEXT NOT NULL,
car_service(#  change_time TIMESTAMP NOT NULL);
CREATE TABLE
car_service=# CREATE OR REPLACE FUNCTION log_changes() RETURNS TRIGGER AS $$
car_service$# BEGIN
car_service$#  IF TG_OP = 'DELETE' THEN
car_service$#  INSERT INTO car_service.log_changes (table_name, change_type, change_time)
car_service$#  VALUES (TG_TABLE_NAME, 'DELETE', NOW());
car_service$#  ELSIF TG_OP = 'INSERT' THEN
car_service$#  INSERT INTO car_service.log_changes (table_name, change_type, change_time)
car_service$#  VALUES (TG_TABLE_NAME, 'INSERT', NOW());
car_service$#  ELSIF TG_OP = 'UPDATE' THEN
car_service$#  INSERT INTO car_service.log_changes (table_name, change_type, change_time)
car_service$#  VALUES (TG_TABLE_NAME, 'UPDATE', NOW());
car_service$#  END IF;
car_service$#  RETURN NEW;
car_service$# END;
car_service$# $$ LANGUAGE PLPGSQL;
CREATE FUNCTION
car_service=#
car_service=# CREATE TRIGGER log_changes_document_trigger AFTER
car_service-# INSERT
car_service-# OR
car_service-# UPDATE
car_service-# OR
car_service-# DELETE ON car_service.document
car_service-# FOR EACH ROW
car_service-# EXECUTE FUNCTION log_changes();
CREATE TRIGGER
car_service-#
```

```
1  select * from car_service.log_changes
```

Data Output | Messages | Notifications

	id [PK] integer	table_name text	change_type text	change_time timestamp without time zone
1	1	document	INSERT	2023-05-14 18:52:00.772929
2	2	document	DELETE	2023-05-31 10:26:12.131612
3	4	document	UPDATE	2023-05-31 10:27:34.207045

Вывод:

В данной работе были изучены функции и процедуры, и созданы триггеры для корректного хранения данных.