

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»  
Факультет инфокоммуникационных технологий

**ОТЧЕТ**  
**О ЛАБОРАТОРНОЙ РАБОТЕ № 3**

по теме:

*«Процедуры, функции, триггеры в PostgreSQL»*

по дисциплине: Проектирование и реализация баз данных

Специальность:

09.03.03 Мобильная и сетевая разработка

Проверила: Говорова

М.М. Дата: «...» ...

2023 г.

Оценка \_\_\_\_\_

Выполнила:

студентка группы

K32392

Барталевич Е.В.

**Цель работы:** овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

**Практическое задание:**

Вариант 1:

1. Создать процедуры/функции согласно индивидуальному заданию и (согласно индивидуальному заданию, часть 4).
2. 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.

**Выполнение работы:**

Предметная область – Автомастерская (вариант 11).

Наименование БД – Garage

## Создать хранимые процедуры:

1. Для повышения цены деталей для автомобиля “Ford” на 10 %.  
Запрос:

```
CREATE OR REPLACE FUNCTION increase_ford_parts_price()
RETURNS VOID AS $$
BEGIN
    UPDATE price p
    SET price = price * 1.1
    FROM details d, model m, car c
    WHERE p.detailcode = d.detailcode
        AND d.modelcode = m.modelcode
        AND c.modelcode = m.modelcode
        AND m.brand = 'Ford';
END;
$$ LANGUAGE plpgsql;
```

Вывод:

price integer	price integer
900	900
1760	1936
1760	1936

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

Запрос:

```
CREATE OR REPLACE FUNCTION promote_masters()
RETURNS VOID AS $$
BEGIN
    UPDATE employee e
    SET "postCode" = "postCode" + 1
    WHERE e.personalnum IN (
        SELECT e.personalnum
        FROM employee e
        JOIN serviceslist sl ON e.personalnum = sl.personalnum
        JOIN contract c ON sl.contractnumber = c.contractnumber
        GROUP BY e.personalnum
        HAVING COUNT(DISTINCT c.registrationnumber) > 3
    );
END;
```

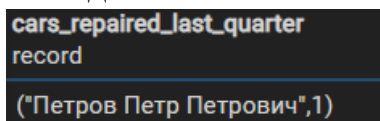
```
$$ LANGUAGE plpgsql;
```

3. Для получения количества автомобилей, отремонтированных каждым механиком за истекший квартал.

Запрос:

```
CREATE OR REPLACE FUNCTION cars_repaired_last_quarter()
RETURNS TABLE (
    mechanic_fullname VARCHAR(50),
    cars_repaired INTEGER
) AS $$
BEGIN
    RETURN QUERY
    SELECT e.fullname, COUNT(DISTINCT c.registrationnumber) AS
cars_repaired
    FROM employee e
        JOIN serviceslist sl ON e.personalnum = sl.personalnum
        JOIN contract c ON sl.contractnumber = c.contractnumber
    WHERE c.enddate >= current_date - INTERVAL '3 months'
    GROUP BY e.fullname;
END;
$$ LANGUAGE plpgsql;
```

Вывод:



cars_repaired_last_quarter	record
Петров Петр Петрович	1

## Создание Триггеров.

1. Триггер, который не даёт добавить больше главных механиков, чем обычных.

Запрос:

```
CREATE OR REPLACE FUNCTION check_mechanics() RETURNS TRIGGER
AS $$
DECLARE
    head_count INTEGER;
    mechanic_count INTEGER;
BEGIN
    SELECT COUNT(*) INTO head_count FROM employee WHERE postCode =
1;
    SELECT COUNT(*) INTO mechanic_count FROM employee WHERE
postCode = 8;
```

```

    IF (NEW.postCode = 1 AND head_count >= mechanic_count) THEN
        RAISE EXCEPTION 'Cannot add more head mechanics than regular
mechanics';
    END IF;

```

```

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER mechanics_trigger
BEFORE INSERT ON employee
FOR EACH ROW
EXECUTE FUNCTION check_mechanics();

```

2. Триггер, запрещающий добавлять несколько пересекающихся во времени контрактов с одной машиной.

Запрос:

```

CREATE OR REPLACE FUNCTION check_overlap() RETURNS TRIGGER AS
$$
BEGIN
    IF EXISTS (SELECT 1
        FROM contract
        WHERE registrationnumber = NEW.registrationnumber
            AND ((NEW.startdate BETWEEN startdate AND enddate)
                OR (NEW.enddate BETWEEN startdate AND enddate)
                OR (NEW.startdate <= startdate AND NEW.enddate >= enddate)))
    THEN
        RAISE EXCEPTION 'Overlap with existing contract is not allowed.';
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER check_overlap_before_insert_or_update
BEFORE INSERT OR UPDATE OF startdate, enddate, registrationnumber ON
contract
FOR EACH ROW
EXECUTE FUNCTION check_overlap();

```

Вывод:

```

ERROR: ОШИБКА:  Overlap with existing contract is not allowed.
CONTEXT:  функция PL/pgSQL check_overlap(), строка 9, оператор RAISE

```

Вывод: при выполнении данной лабораторной работы были приобретены практические навыки по работе с таблицами в базе данных PostgreSQL, выполнением процедур и функций, а также использованием триггеров.