

**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: Чернышев М.П.

Факультет: ИКТ

Группа: К3241 Преподаватель:

Говорова М.М.



Санкт-Петербург 2023

## Оглавление

Цель работы:.....	3
Практическое задание.....	3
Выполнение.....	3
Вывод.....	6

**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).

**Практическое задание:**

**Вариант 2 (max - 8 баллов)**

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. Создать авторский триггер по варианту индивидуального задания.

**Выполнение**

**Создайте хранимые процедуры:**

- Для повышения оклада сотрудников, выполнивших задания с трехдневным опережением графика на заданный процент.

```
CREATE OR REPLACE FUNCTION increase_salary_for_completed_tasks(  
    IN employee_id INT,  
    IN percentage_increase DECIMAL  
)  
RETURNS TABLE (  
    Табельный_номер INT,  
    Оклад DOUBLE PRECISION  
)  
LANGUAGE PLPGSQL  
AS $$  
BEGIN  
    RETURN QUERY  
        SELECT Сотрудник.Табельный_номер, Должность.Оклад  
        FROM Должность  
        JOIN Сотрудник_в_отделе ON Сотрудник_в_отделе.Номер_должности =  
Должность.Номер_должности  
        JOIN Сотрудник ON Сотрудник.Табельный_номер =  
Сотрудник_в_отделе.Табельный_номер  
        WHERE Сотрудник.Табельный_номер = employee_id;  
  
    UPDATE Должность  
    SET Оклад = Должность.Оклад * (1 + percentage_increase / 100)  
    FROM Сотрудник_в_отделе  
    JOIN Задача ON Задача.Номер_сотрудника_в_отделе =  
Сотрудник_в_отделе.Номер_сотрудника_в_отделе
```

```

WHERE Сотрудник_в_отделе.Номер_должности = Должность.Номер_должности
AND Сотрудник_в_отделе.Табельный_номер = employee_id AND
Задача.Дата_окончания_выполнения - Сотрудник_в_отделе.Конец_действия_договора > 3;
END;
$$;

```

В psql

```

[postgres=# SELECT * FROM increase_salary_for_completed_tasks(2,10);
Табельный_номер | Оклад
-----+-----
                2 | 13068.0000000000004
(1 row)

```

- Для вычисления количества проектов, в выполнении которых участвует сотрудник.

```

CREATE OR REPLACE FUNCTION count_projects_for_employee(
    IN employee_id INT
)
RETURNS INTEGER
LANGUAGE PLPGSQL
AS $$
DECLARE
    project_count INT;
BEGIN
    SELECT COUNT(DISTINCT Договор.Номер_проекта)
    INTO project_count
    FROM Договор
    JOIN Сотрудник_в_отделе ON Сотрудник_в_отделе.Номер_проекта =
Договор.Номер_проекта
    JOIN Сотрудник ON Сотрудник.Табельный_номер = Сотрудник_в_отделе.Табельный_номер
    WHERE Сотрудник.Табельный_номер = employee_id;

    RETURN project_count;
END;
$$;

```

В psql

```

[postgres=# SELECT * FROM count_projects_for_employee(7);
count_projects_for_employee
-----
                            1
(1 row)

```

```
[postgres=# SELECT * FROM count_projects_for_employee(11);
 count_projects_for_employee
-----
                                0
(1 row)
```

- Для поиска номера телефона сотрудника (телефон установлен в каждом отделе).

```
CREATE OR REPLACE FUNCTION find_employee_phone(
    IN employee_id INT
)
RETURNS TABLE (
    Табельный_номер INT,
    Номер_телефон BIGINT
)
LANGUAGE PLPGSQL
AS $$
BEGIN
    RETURN QUERY
    SELECT Сотрудник.Табельный_номер, Отдел.Номер_телефона::BIGINT
    FROM Сотрудник
    JOIN Сотрудник_в_отделе ON Сотрудник_в_отделе.Табельный_номер =
Сотрудник.Табельный_номер
    JOIN Отдел ON Отдел.Номер_отдела = Сотрудник_в_отделе.Номер_отдела
    WHERE Сотрудник.Табельный_номер = employee_id;
END;
$$;
```

В psql

```
[postgres=# SELECT * FROM find_employee_phone(1);
Табельный_номер | Номер_телефон
-----+-----
1 | 89001111111
(1 row)
```

**Триггер:**

- Функционал триггера позволяет автоматически изменять статус выполнения заказа при непрохождении контроля.

```
CREATE OR REPLACE FUNCTION update_stage_status()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE Этап_проекта
    SET Статус_выполнения = NEW.Статус_контроля
    WHERE Этап_проекта.Номер_этапа = NEW.Номер_этапа;


    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER update_stage_status_trigger
AFTER UPDATE ON Контроль_выполнения
```

FOR EACH ROW

EXECUTE FUNCTION update\_stage\_status();

Теперь при изменении статуса контроля автоматически меняется статус выполнения этапа проекта.

	Статус_выполнения character varying	Статус_контроля character varying (20) 
1	Не выполнен	Не выполнен

- Функционал триггера позволяет автоматически пересчитывать оплату работы сотрудника при изменении его оклада в таблице должность.

CREATE OR REPLACE FUNCTION update\_payment\_amount()

RETURNS TRIGGER AS \$\$

BEGIN

UPDATE Оплата\_работы

SET Сумма = NEW.Оклад \* Сотрудник\_в\_отделе.Доля\_ставки

FROM Сотрудник\_в\_отделе

WHERE Оплата\_работы.Табельный\_номер = Сотрудник\_в\_отделе.Табельный\_номер;

RETURN NEW;

END;

\$\$ LANGUAGE plpgsql;


CREATE TRIGGER update\_payment\_amount\_trigger

AFTER UPDATE ON Должность

FOR EACH ROW

EXECUTE FUNCTION update\_payment\_amount();

Теперь при изменении оклада автоматически пересчитывается сумма оплаты работы сотрудника.

	Сумма integer 
1	45000

## Вывод

В ходе лабораторной работы была освоена работа с процедурами и триггера.

