## 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»

#### Выполнила:

Коник Анастасия Александровна Группа K3241 **Проверила:** 

Говорова Марина Михайловна

#### ЦЕЛЬ РАБОТЫ:

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

Оборудование: компьютерный класс.

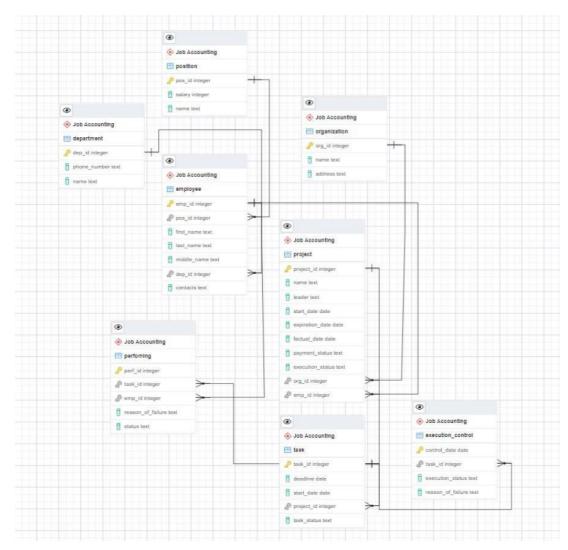
Программное обеспечение: СУБД PostgreSQL, SQL Shell (psql).

#### 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: "Учет выполнения заданий"



#### Выполнение:

#### Задание 1. Создайте хранимые процедуры.

1) Для повышения оклада сотрудников, выполнивших задания с трехдневным опережением графика на заданный процент (10%).

4	emp_id [PK] integer	pos_id integer		first_name text	last_name text	middle_name text	dep_id integer	•	contacts text	project_id integer	salary double precision
1	1		1	Anastasia	Vorobieva	Alexandrovna	1	1	sbssu@mail.ru	111	8500
2	2		2	Olga	Minaeva	Igorevna	2	2	hjkaw@gmail.com	222	9500
3	3		3	Alex	Kross	[null]	3	3	dguws@ya.ru	111	6000
4	4		2	Anna	Karavaeva	Dmitrievna	1	1	tefas@mail.ru	111	9000
5	5		1	Lev	Korolyev	[null]	2	2	huhd@ya.ru	222	10000
6	6		3	Andrey	Smirnov	Alexandrovich	3	3	egysv@gmail.com	333	5000
7	7		2	Daniil	Mironenko	Fedorovich	1	1	dwuid@mail.ru	111	11000
8	8		3	Svetlana	Pupirishkina	Gennadievna	2	2	feuhi@mail.ru	222	5500
9	9		1	Marina	Konovalova	Danilovna	3	3	kcjae@gmail.com	222	8000
10	10		1	Igor	Sivaev	Sergeevich	2	2	wafcz@gmail.com	[null]	105000

```
CREATE PROCEDURE salary_raise()

AS

$$
BEGIN
UPDATE j_acc.employee
SET salary = salary*1.1
WHERE salary IN (SELECT employee.salary FROM j_acc.employee
JOIN j_acc.task ON employee.emp_id = task.emp_id WHERE (task.deadline - task.factual_end_date) = 3);
END;

$$ language plpgsql;

Data Output Messages

CREATE PROCEDURE

Query returned successfully in 172 msec.
```

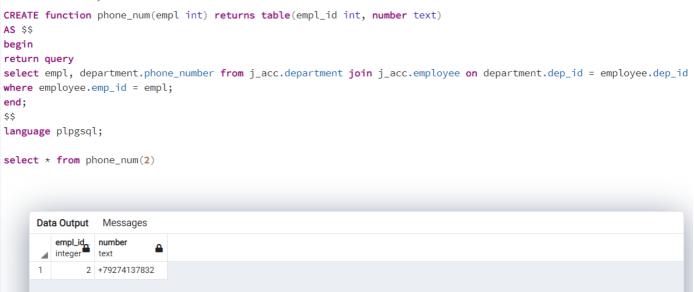
#### Data Output Messages

4	emp_id [PK] intege	pos_id integer	first_name text	text	middle_name text	dep_id integer	contacts text	project_id integer	salary integer
1	1	1	Anastasia	Vorobieva	Alexandrovna	1	sbssu@mail.ru	111	85000
2	2	2	Olga	Minaeva	Igorevna	2	hjkaw@gmail.com	222	104500
3	3	3	Alex	Kross	[null]	3	dguws@ya.ru	111	60000
4	4	2	Anna	Karavaeva	Dmitrievna	1	tefas@mail.ru	111	90000
5	5	1	Lev	Korolyev	[null]	2	huhd@ya.ru	222	100000
6	6	3	Andrey	Smirnov	Alexandrovich	3	egysv@gmail.com	333	50000
7	7	2	Daniil	Mironenko	Fedorovich	1	dwuid@mail.ru	111	110000

2) Для вычисления количества проектов, в выполнении которых участвует сотрудник.

```
CREATE function emp_pr(empl int) returns table(empl_id int, number_of_pr bigint)
AS $$
begin
return query
select empl, count(employee.project_id) from j_acc.employee where employee.emp_id = empl;
$$
language plpgsql;
select * from emp_pr(3)
     Data Output Messages
        empl_id_
                number_of_pr_
      bigint
      1
              3
                         1
```

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



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

#### Создана таблица logs

```
CREATE OR REPLACE FUNCTION add_to_log() RETURNS TRIGGER AS $$
                                                                                        CREATE FUNCTION
DECLARE
   mstr varchar(30);
                                                                                        Query returned successfully in 72 msec.
   astr varchar(100):
   retstr varchar(254);
BEGIN
       TG_OP = 'INSERT' THEN
       astr = NEW;
       mstr := 'Add new data';
       retstr := mstr||astr;
       INSERT INTO j_acc.logs(text,added,table_name) values (retstr,NOW(),TG_TABLE_NAME);
       RETURN NEW;
   ELSIF TG_OP = 'UPDATE' THEN
       astr = NEW:
       mstr := 'Update data ';
       retstr := mstr||astr;
       INSERT INTO j_acc.logs(text,added,table_name) values (retstr,NOW(),TG_TABLE_NAME);
       RETURN NEW;
   ELSIF TG_OP = 'DELETE' THEN
       astr = OLD:
       mstr := 'Remove data':
       retstr := mstr || astr;
       INSERT INTO j_acc.logs(text,added,table_name) values (retstr,NOW(),TG_TABLE_NAME);
       RETURN OLD:
   END IF;
END;
$$ LANGUAGE plpgsql;
CREATE TRIGGER t_emp AFTER INSERT OR UPDATE OR DELETE
                                                                                    CREATE TRIGGER
ON j_acc.employee FOR EACH ROW EXECUTE PROCEDURE add_to_log ();
                                                                                    Query returned successfully in 107 msec.
INSERT INTO j_acc.department(dep_id, phone_number, name) VALUES (4, '+789198342782', 'Something');
UPDATE j_acc.department SET name = 'Anything' where dep_id = 4;
DELETE from j_acc.department where dep_id = 4;
select * from j_acc.logs
```

#### **Data Output** Messages text added table\_name dext date text 1 Add new data(4,+789191253612,Something) 2022-05-03 department 2 Update data (4,+789191253612,Anything) 2022-05-03 department department 3 Remove data(4,+789191253612,Anything) 2022-05-03

#### Вывод:

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