## 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: Котовщиков А. Р.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.



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).

## Практическое задание:

#### Вариант 2 (тах - 8 баллов)

- 1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
- 2. Модифицировать триггер на проверку корректности входа и выхода сотрудника.
- 3. Создать авторский триггер по варианту индивидуального задания.

#### Выполнение

## Создайте хранимые процедуры:

• Для получения расписания занятий для групп на определенный день недели.

```
create or replace function lab_3_2.get_group_schedule_by_week_day(
    name_of_group varchar,
    day_of_week lab_3_2.schedule_week_day
)

returns table(
    subject_name varchar,
    class_time varchar,
    address varchar,
    classroom varchar,
    group_name varchar,
    week_day lab_3_2.schedule_week_day,
    classroom_date date
)

language plpgsql
```

```
as $$
declare
      nearest date date;
begin
      nearest date := (SELECT MIN(s.datetime::date) FROM lab 3 2.schedule AS
            WHERE s.week_day = day_of_week
                  AND s.group_name = name_of_group
                        AND s.datetime::date >= CURRENT DATE);
      return query (
            SELECT
                  subj.name,
                  to char(s.datetime, 'HH24:MI')::varchar AS class time,
                  bs.address,
                  c.number,
                  s.group name,
                  s.week day,
                  s.datetime::date AS classroom_date
            FROM lab 3 2.schedule AS s
                  JOIN lab 3 2.classroom AS c
                        ON s.classroom code = c.code
                  JOIN lab_3_2.branch_site AS bs
                        ON c.branch site id = bs.id
                  JOIN lab 3 2.curriculum subject AS cs
                        ON s.curriculum_subject_id = cs.id
                  JOIN lab_3_2.subject AS subj
```

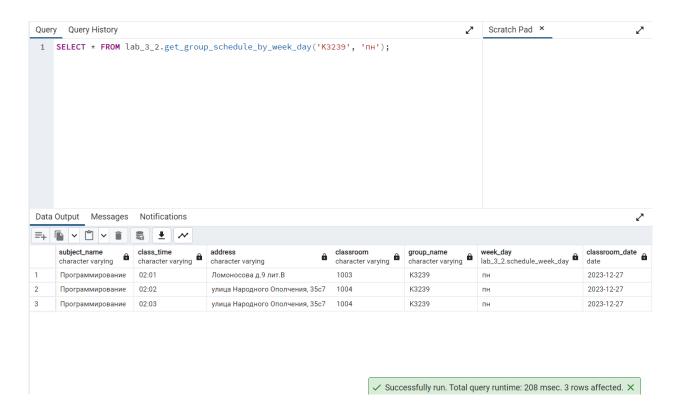
```
ON cs.subject_code = subj.code

WHERE s.datetime::date = nearest_date

AND s.group_name = name_of_group

ORDER BY class_time

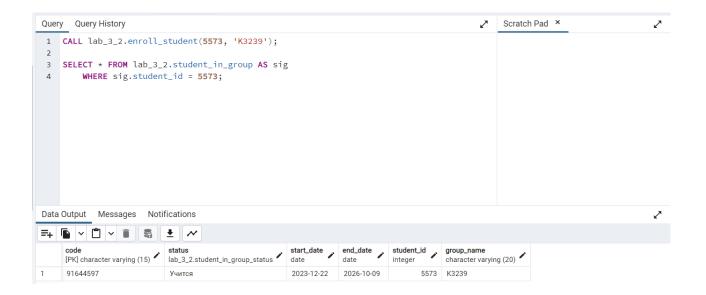
);
end;$$;
```



#### • Для записи на курс слушателя.

```
create or replace procedure lab 3 2.enroll student(
      student id int,
      group name varchar
language plpgsql
as $$
declare
      is student exists bool;
      is_group_exists bool;
begin
      is student exists := (
           SELECT
                 CASE WHEN COUNT(*) = 0
                       THEN false
                 ELSE true
                 END
            FROM lab_3_2.student AS s
                 WHERE s.id = student id);
```

```
IF NOT is student exists
        THEN RAISE EXCEPTION 'Студент с идентификатором % не
существует!', student id;
     END IF;
     is group exists := (
           SELECT CASE WHEN COUNT(*) = 0
                      THEN false
                ELSE true
                END
           FROM lab_3_2.group AS g
                WHERE g.name = group name);
     IF NOT is_group_exists
           THEN RAISE EXCEPTION 'Группа % не существует!', group name;
     END IF;
     INSERT INTO lab 3 2.student in group
           status,
           start date,
           end date,
           student id,
           group_name
     )
     VALUES
      (
           'Учится',
           NOW(),
           (SELECT ic.end date FROM lab 3 2.implemented curriculum AS ic
WHERE EXISTS (
                SELECT 1 FROM lab 3 2.group AS g WHERE g.name =
group_name
                      AND g.implemented_curriculum_id = ic.id)
           ),
           student id,
           group_name
     );
end; $$;
```



• Для получения перечня свободных лекционных аудиторий на любой день недели.

```
create or replace function lab 3 2.get free classrooms or throw exception(
      day of week lab 3 2.schedule week day
returns table (
      code varchar,
     number varchar,
     type lab_3_2.classroom_type,
     capacity smallint,
     address varchar
language plpgsql
as $$
declare
      free classroom count int;
begin
      free classroom count := (SELECT COUNT(*)
           FROM lab 3 2.classroom AS c
           JOIN lab 3 2.branch site AS bs ON c.branch site id = bs.id
           WHERE NOT EXISTS (
                 SELECT 1 FROM lab 3 2.schedule AS s
                       WHERE s.classroom code = c.code
                             AND s.week day = day of week)
     );
      IF free classroom count = 0
           THEN RAISE EXCEPTION 'Свободных аудиторий на % нет!',
day_of_week;
     END IF;
      return query(
           SELECT
                 c.code,
                 c.number,
```

```
c.type,
                          c.capacity,
                          bs.address
                 FROM lab 3 2.classroom AS c
                 JOIN lab 3 2.branch site AS bs ON c.branch site id = bs.id
                 WHERE NOT EXISTS (
                          SELECT 1 FROM lab 3 2.schedule AS s
                                  WHERE s.classroom code = c.code
                                           AND s.week day = day of week)
        );
end; $$;
Query Query History
                                                                                  Scratch Pad ×
 1 SELECT * FROM lab_3_2.get_free_classrooms_or_throw_exception('cp');
 Data Output Messages Notifications
 =+ □ ∨ □ ∨ □ ■ ■ ...
     code number character varying a number
                               type | capacity | smallint | address | character varying
                                                                                   â
     Aud-1015
                  1015
                                                      84 ул. Свободы, 192
17
     Aud-1016
                  1016
                                Лабораторная
                                                     149 ул. Свободы, 192
    Aud-1017
                  1017
18
                                Лекционная
                                                     172 пер. Горького, 16, микрорайон Центральный
19
    Aud-1018
                  1018
                                                      74 Ломоносова д.9 лит.В
                               Лекционная
   Aud-1019
                  1019
                                                     191 Ломоносова д.9 лит.В
20
                               Лекционная
```

# Модифицировать триггер на проверку корректности входа и выхода сотрудника:

Лабораторная

Практическая

Лабораторная

21

22

23

Aud-1020

Aud-1021

Aud-1022

24 Aud-1023

1021

1022

1023

```
create or replace function fn_check_time_punch()
returns trigger
language plpgsql
as $psql$
begin
-- Валидация входов
IF NEW.is_out_punch = false THEN
-- Проверка дата входа >= текущей даты (иначе null)
IF NEW.punch_time < NOW()
THEN return null;

-- Нельзя войти еще раз, пока ты не вышел
ELSIF (
SELECT punch_count - out_punch_count FROM (
SELECT
```

135 ул. Свободы, 192

124 ул. Свободы, 192

81 Ломоносова д.9 лит.В

57 пер. Горького, 16, микрорайон Центральный

```
(SELECT COUNT(*) FROM time_punch
                           WHERE is out punch = false
                                 AND employee id = NEW.employee id
                     ) AS punch count,
                     (SELECT COUNT(*) FROM time punch
                           WHERE is out punch = true
                                 AND employee id = NEW.employee id
                     ) AS out punch count
               ) AS table alias
         ) <> 0
              THEN return null;
         ELSE
         END IF;
   END IF;
   -- Валидация выходов
   IF NEW.is out punch = true THEN
         -- Количество входов должно быть на 1 больше, чем выходов
               SELECT punch count - out punch count FROM (
                     SELECT DISTINCT
                     (SELECT COUNT(*) FROM time punch
                           WHERE is out punch = false
                                 AND employee_id = NEW.employee_id
                     ) AS punch count,
                     (SELECT COUNT(*) FROM time punch
                           WHERE is_out_punch = true
                                 AND employee id = NEW.employee id
                     ) AS out punch count
               ) AS table alias
         ) <> 1
               THEN return null;
         -- Дата последнего входа должна быть меньше даты выхода
         ELSIF
               (SELECT punch time FROM time punch AS tp
                     WHERE tp.is out punch = false
                           AND tp.employee_id = NEW.employee_id
                ORDER BY punch time DESC
                LIMIT 1) >= NEW.punch time
                     THEN return null;
         ELSE
         END IF;
  END IF;
  return NEW;
end;
$psql$;
CREATE TRIGGER punch time checker
BEFORE INSERT
```

```
ON time_punch
FOR EACH ROW EXECUTE FUNCTION fn check time punch();
```

## Создайте необходимые триггеры:

91644597

• Триггер, который автоматически генерирует уникальный код для студента в группе.

```
create or replace function generate code for student in group()
returns trigger
language plpgsql
AS $$
declare
                       student in group code varchar;
begin
                       student in group code := (
                                              SELECT MAX(sig.code::int) + 1
                                                                    FROM lab 3 2.student in group AS sig);
                       IF student_in_group_code IS NULL
                                             THEN student in group code := 1;
                      END IF;
                      NEW.code = student in group code;
                      return NEW;
end; $$;
CREATE OR REPLACE TRIGGER generate code for student in group trigger
BEFORE INSERT ON lab 3 2.student in group
FOR EACH ROW EXECUTE FUNCTION generate code for student in group();
                                                                                                                                                                                                                 Scratch Pad ×
 Query Query History
   1 -- CALL lab_3_2.enroll_student(5573, 'K3239');
    3 SELECT * FROM lab_3_2.student_in_group AS sig
                   WHERE sig.student_id = 5573;
  Data Output Messages Notifications
  타 🖺 v 📋 v 📋 📽 👤 ~
             code [PK] character varying (15) status | start_date | end_date | date | date | student_in_egroup_name | character varying (15) | status | status | date | date | date | date | date | date | student_in_egroup_name | character varying | status | st
                                                                                                                                                                                                character varying (20)
```

2023-12-22 2026-10-09

5573 K3239

# Вывод

В рамках данной лабораторной работы мы познакомились с возможностями процедурного программирования в языке SQL, научились создавать процедуры, функции, а также использовать триггеры.