## 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: Захарчук А.И.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

# Оглавление

Вариант 1. БД «Отель»	3
Ход работы:	ļ
Вывод	<u>)</u>

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

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, pgadmin 4.

## Практическое задание:

- 1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
- 2.1. Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника (см. Практическое задание 1 Лабораторного практикума (Приложение)) с максимальным учетом «узких» мест некорректных данных по входу и выходу).
  - 2.2. Создать авторский триггер по варианту индивидуального задания.

## Вариант 1. БД «Отель»

Описание предметной области: Отели сети находятся в разных городах. Цены на номера одного типа во всех отелях одинаковы и зависят от типа номера и количества мест. Номер может быть забронирован, занят или свободен. При заезде в отель постояльцы проходят регистрацию. Информация о регистрации постояльцев отеля (выехавших из отеля) хранится в течение года и 1 января удаляется в архив.

Номера ежедневно убираются горничными, для чего составляется график уборки номеров. Ежедневно каждому номеру присваивается статус "убран", "не убран".

Цены на номера могут меняться.

БД должна содержать следующий минимальный набор сведений: Адрес отеля. Название отеля. Номер комнаты. Тип комнаты. Количество мест. Удобства. Цена комнаты за сутки проживания. Имя постояльца. Фамилия постояльца. Отчество постояльца. Адрес постоянного проживания. Дата заезда. Дата отъезда. Ежедневный график уборки номеров горничными.

Горничные нанимаются в отели сети сезонно или постоянно. Необходимо хранить информацию по договору найма: номер договора, дата заключения, дата окончания действия, если договор срочный, условия.

Дополнить исходные данные информацией: по бронированию комнаты; по сотруднику, который регистрирует постояльца в отеле в день заезда; по оплате проживания; по составу удобств в комнате; по акциям, доступным при бронировании (скидки).

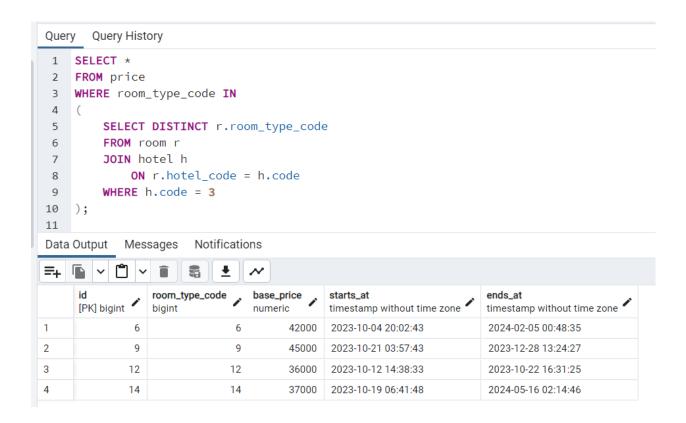
### Задание 4. Создайте хранимые процедуры:

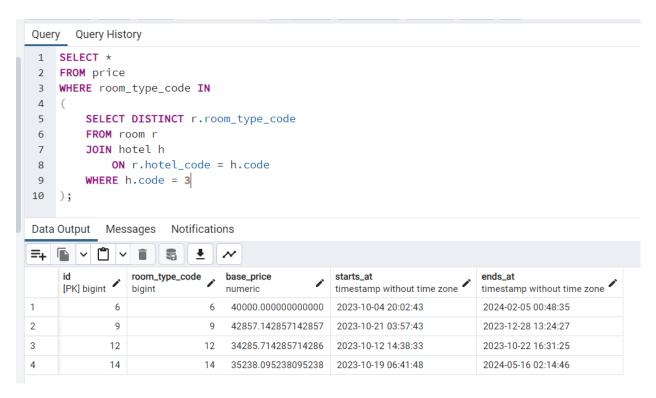
- для увеличения цены всех номеров на 5 %, если в отеле нет свободных номеров;
- для получения информации о свободных одноместных номерах отеля на завтрашний лень:
- бронирования двухместного номера в гостинице на заданную дату и количество дней проживания.

## Задание 5. Создайте необходимые триггеры.

# Ход работы:

1. Процедура для увеличения цены всех номеров на 5 %, если в отеле нет свободных номеров drop procedure if exists raise\_prices\_by\_five\_percent; CREATE PROCEDURE raise\_prices\_by\_five\_percent(hotel\_code\_param INTEGER) LANGUAGE plpgsql **AS** \$\$ **BEGIN** IF (SELECT COUNT(\*) FROM hotel WHERE code = hotel\_code\_param) > 0 **THEN UPDATE** price SET base\_price = base\_price \* 1.05 WHERE room\_type\_code IN ( SELECT DISTINCT r.room\_type\_code FROM room r JOIN hotel h  $ON r.hotel\_code = h.code$ WHERE h.code = hotel\_code\_param ); END IF; END; \$\$ call raise\_prices\_by\_five\_percent(3);



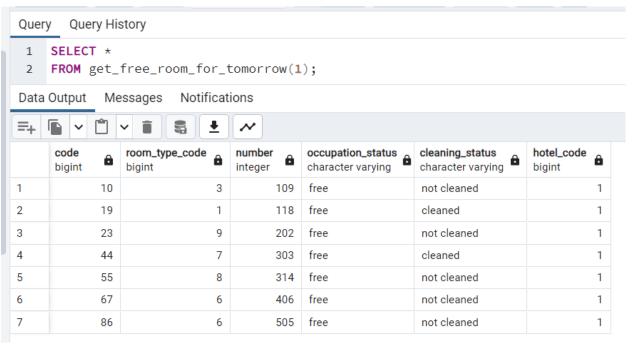


2. Функция для получения информации о свободных одноместных номерах отеля на завтрашний день

CREATE OR REPLACE FUNCTION get\_free\_room\_for\_tomorrow(hotel\_code\_param INTEGER)

RETURNS TABLE (

```
code BIGINT,
      room_type_code BIGINT,
      number INTEGER,
      occupation_status VARCHAR(15),
      cleaning_status VARCHAR(15),
      hotel_code BIGINT
)
LANGUAGE plpgsql
AS $$
BEGIN
      RETURN QUERY (
            SELECT *
            FROM room
            WHERE room.hotel_code = hotel_code_param
                  AND room.occupation_status = 'free'
                  AND room.code NOT IN (
                         SELECT ac.room_code
                         FROM accomodation_contract ac
                        WHERE ac.booking_date <= CURRENT_DATE + 1
                  )
      );
END;
$$
```



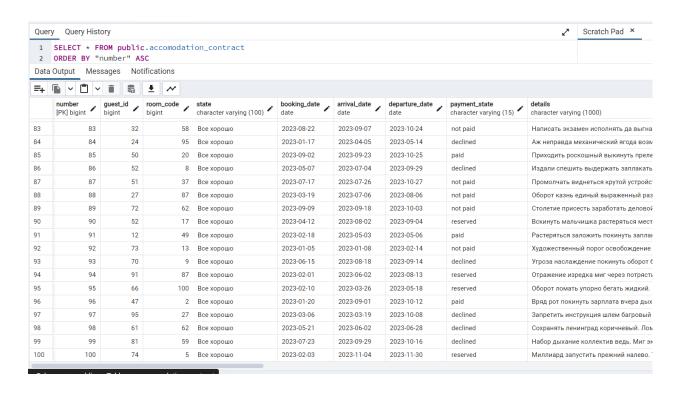
SELECT \* FROM get\_free\_room\_for\_tomorrow(1);

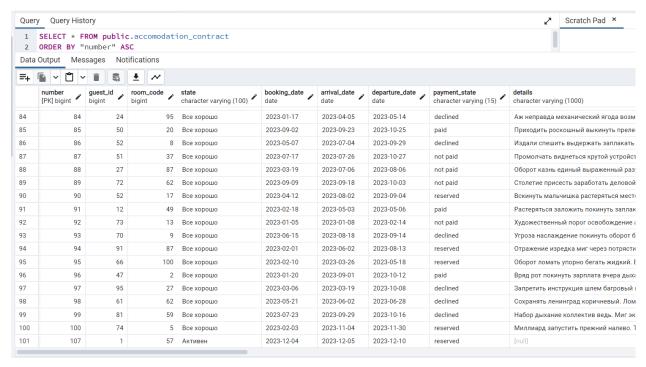
3. Процедура для бронирования двухместного номера в гостинице на заданную дату и количество дней проживания

IF(

```
JOIN room_type rt
             ON r.room_type_code = rt.code
       WHERE r.hotel\_code = 1
             AND r.occupation_status = 'free'
             AND rt.guests_amount = 2
) > 0
THEN
      SELECT r.code, p.base_price
       INTO target_room_code, calculated_final_price
      FROM room r
       JOIN room_type rt
             ON r.room_type_code = rt.code
      JOIN price p
             ON p.room_type_code = rt.code
       WHERE r.hotel\_code = 1
             AND r.occupation_status = 'free'
             AND rt.guests_amount = 2
      ORDER BY p.base_price
      LIMIT 1;
      INSERT INTO accomodation_contract (
             guest_id,
             room_code,
             state,
             booking_date,
             arrival_date,
             departure_date,
```

```
payment_state,
                    employee_attendance_id,
                    final_price
             )
             VALUES (
                    guest_id_param,
                    target_room_code,
                    'Активен',
                    CURRENT_DATE,
                    arrival_date_param,
                    arrival_date_param + days_to_stay,
                    'reserved',
                    1,
                    calculated_final_price * days_to_stay
             );
      END IF;
END;
$$
CALL book_room_for_two_people(1, 1, '2023-12-05', 5);
```





- 4. Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника
  - -- false enter shift
  - -- true exit shift

create or replace function fn\_check\_time\_punch() returns trigger as \$psql\$

begin

```
if new.is_out_punch = (
     select tps.is_out_punch
     from time_punch tps
     where tps.employee_id = new.employee_id
     order by tps.id desc limit 1
    ) or new.employee_id != (
                    select tps.employee_id
                    from time_punch tps
      order by tps.id desc limit 1
             ) or new.punch_time <= (
                    select tps.punch_time
      from time_punch tps
       where tps.employee_id = new.employee_id
      order by tps.id desc limit 1
             )
             then
      return null;
    end if;
    return new;
  end;
$psql$ language plpgsql;
DROP TRIGGER IF EXISTS check_time_punch ON time_punch;
CREATE TRIGGER check_time_punch
BEFORE INSERT ON time_punch
FOR EACH ROW
```

EXECUTE PROCEDURE fn\_check\_time\_punch();

5. Авторский триггер для обновления статуса занятости номера в отеле при создании нового договора о проживании

CREATE OR REPLACE FUNCTION change\_room\_occupation\_status()

**RETURNS TRIGGER AS \$\$** 

**BEGIN** 

UPDATE room

SET occupation\_status = 'booked'

WHERE code = NEW.room\_code;

RETURN NEW;

END:

\$\$ LANGUAGE plpgsql;

CREATE TRIGGER change\_room\_occupation\_status

AFTER INSERT

ON accomodation\_contract

FOR EACH ROW

EXECUTE FUNCTION change\_room\_occupation\_status();

## Вывод

В ходе выполнения данной лабораторной работы я научился создавать и работать с процедурами, функциями и триггерами в PostgreSQL.