Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе «Процедуры, функции, триггеры в PostgreSQL»

по дисциплине «**Базы данных**»

Автор: Циминтия Н

Факультет: ИКТ

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

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

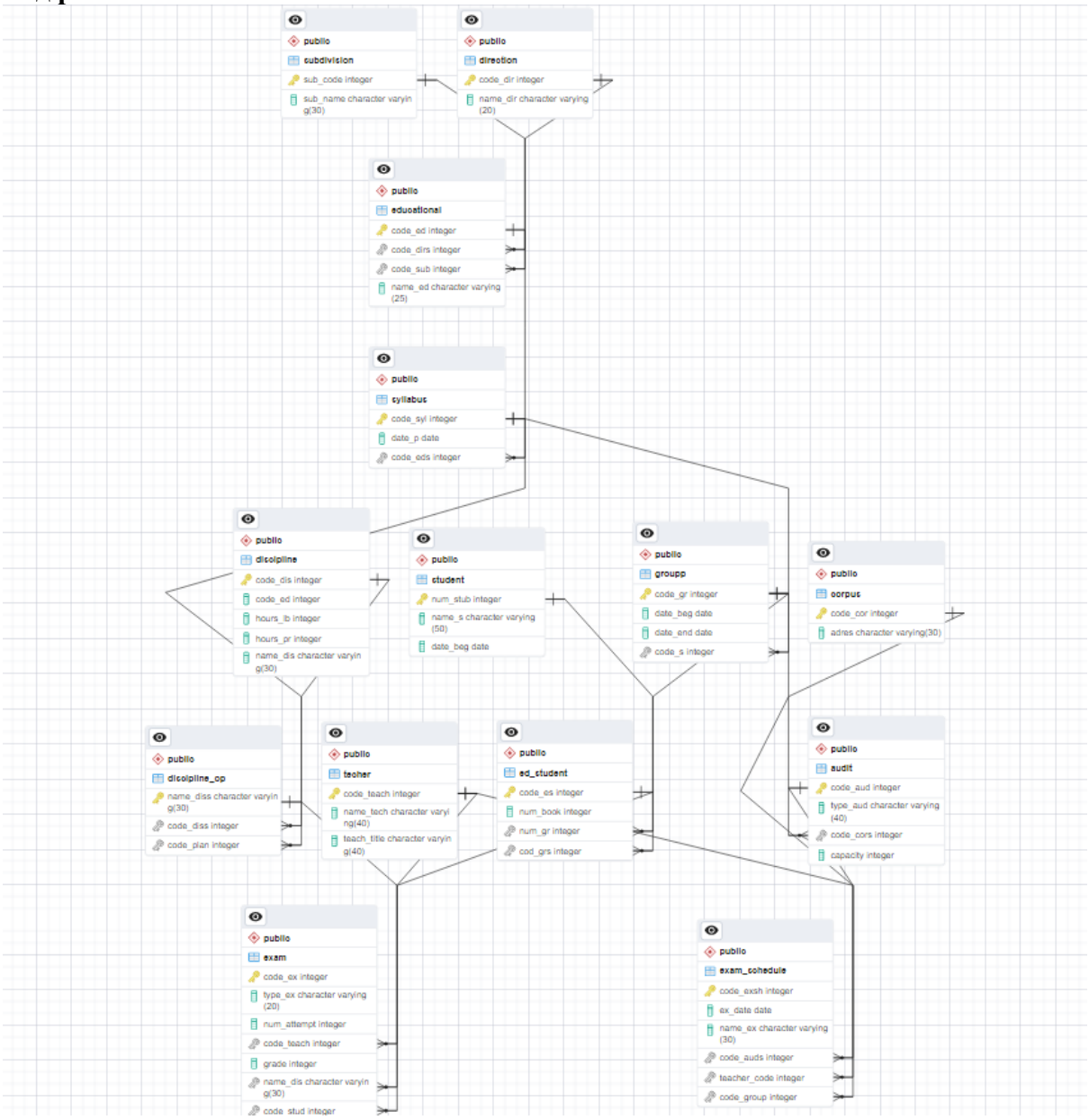
Оборудование: компьютерный класс.

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

Практическое задание:
Вариант 1

- скрипты кода разработанных объектов (процедур/функций и триггера на логирование действий) и подтверждающие скриншоты работы и результатов в psql согласно индивидуальному заданию (часть 4 и 5).

Ход работы:



Создать хранимые процедуры:
для увеличения цены всех номеров на 5 %, если в отеле нет свободных номеров

```
CREATE OR REPLACE FUNCTION increase_price_if_no_available_rooms()
RETURNS VOID AS
$$
DECLARE
  has_available_rooms BOOLEAN;
BEGIN
  -- Проверяем наличие свободных номеров
  SELECT EXISTS (
    SELECT 1
    FROM room
    LEFT JOIN orders ON room.code_rooms = orders.code_room
    WHERE orders.code_room IS NULL
  ) INTO has_available_rooms = TRUE ;

  IF has_available_rooms = false THEN
    -- Если свободных номеров нет, увеличиваем цену на 5%
    UPDATE orders
    SET prices = prices * 1.05;
  END IF;
END;
$$
LANGUAGE plpgsql;
```

для получения информации о свободных одноместных номерах отеля на завтрашний день

```
CREATE OR REPLACE FUNCTION get_available_single_rooms_tomorrow()
RETURNS TABLE (
  room_number varchar(50)
) AS
$$
BEGIN
  RETURN QUERY
  SELECT room.code_rooms
  FROM room
  WHERE room.code_type_r = '13' -- Замените на соответствующее значение для типа вашего номера
    AND room.code_rooms NOT IN (
      SELECT orders.code_room
      FROM orders
      WHERE beg = CURRENT_DATE + INTERVAL '1 day'
    );
END;
$$
LANGUAGE plpgsql;
```

Вывод:

```
1  SELECT * FROM get_available_single_rooms_tomorrow();
2
```

Data Output | Сообщения | Notifications

	room_number character varying
1	13

бронирования двухместного номера в гостинице на заданную дату и количество дней проживания

```
CREATE OR REPLACE FUNCTION book_double_room(check_in_date DATE, duration INTEGER)
RETURNS VOID AS
$$
BEGIN
  -- Проверяем наличие свободного двухместного номера на заданную дату
  IF EXISTS (
    SELECT 1
    FROM room
    WHERE code_type_r = '15' -- Замените на соответствующее значение для типа вашего номера
      AND code_rooms NOT IN (
        SELECT code_room
        FROM orders
        WHERE beg <= check_in_date + duration - 1
          AND ends >= check_in_date
      )
  ) THEN
    -- Если есть свободный номер, выполняем бронирование
    INSERT INTO orders (code_room, beg, ends)
    VALUES (
      (SELECT code_rooms
       FROM room
       WHERE code_type_r = '15' -- Замените на соответствующее значение для типа вашего номера
         AND code_rooms NOT IN (
           SELECT code_room
           FROM orders
           WHERE beg <= check_in_date + duration - 1
             AND ends >= check_in_date
         )
       LIMIT 1),
      check_in_date,
      check_in_date + duration - 1
    );
  END IF;
END;
$$
LANGUAGE plpgsql;
```

Триггер для логирования событий вставки, удаления и обновления данных в таблице

Вывод

```sql
-- Создание таблицы логирования
CREATE TABLE client_log (
    log_id SERIAL PRIMARY KEY,
    event_type VARCHAR(20) NOT NULL,
    event_timestamp TIMESTAMP DEFAULT current_timestamp,
    client_id INTEGER,
    old_data JSONB,
    new_data JSONB
);

-- Создание треггера для логирования взаимодействий с таблицей client
CREATE OR REPLACE FUNCTION client_log_trigger_function()
RETURNS TRIGGER AS
$$
BEGIN
    IF (TG_OP = 'INSERT') THEN
        INSERT INTO client_log (event_type, client_id, new_data)
        VALUES ('INSERT', NEW.client_id, to_jsonb(NEW));

    ELSIF (TG_OP = 'UPDATE') THEN
        INSERT INTO client_log (event_type, client_id, old_data, new_data)
        VALUES ('UPDATE', NEW.client_id, to_jsonb(OLD), to_jsonb(NEW));

    ELSIF (TG_OP = 'DELETE') THEN
        INSERT INTO client_log (event_type, client_id, old_data)
        VALUES ('DELETE', OLD.client_id, to_jsonb(OLD));
    END IF;

    RETURN NEW;
END;
$$
LANGUAGE plpgsql;

-- Создание треггера
CREATE TRIGGER client_log_trigger
AFTER INSERT OR UPDATE OR DELETE ON client
FOR EACH ROW
EXECUTE FUNCTION client_log_trigger_function();
```

В ходе лабораторной работы я научился создавать и использовать процедуры, функции и триггеры в базе данных PostgreSQL. Также, я понял, что функции и процедуры в SQL недостаточно гибкие, как в ЯП.