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

Отчет

По Лабораторной работе 3
«Процедуры, функции, триггеры в PostgreSQL»

по дисциплине «Проектирование и реализация баз данных»

Автор:Казанков И

Факультет:ИКТ

Группа: К32402

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 1

1. 2.Создать процедуры/функции согласно индивидуальному заданию и (согласно индивидуальному заданию, часть 4).
2. 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.
    1.

Выполнение:

Хранимые процедуры:

Для повышения цен в пригородные поезда на 20%:

```sql
CREATE OR REPLACE FUNCTION increase_suburban_prices()
RETURNS VOID AS $$
BEGIN
    UPDATE regular_schedule
    SET price_ticket = price_ticket * 1.2
    WHERE type_train = 'Express';
END;
$$ LANGUAGE plpgsql;

```

Data Output | Сообщения | Notifications

```
CREATE FUNCTION

Запрос завершён успешно, время выполнения: 42 msec.
```

Для создания нового рейса на поезд:

```sql
CREATE OR REPLACE FUNCTION create_train_route(
    p_train_name VARCHAR,
    p_arrival_time TIME,
    p_departure_time TIME,
    p_train_arrival_point VARCHAR,
    p_train_departure_point VARCHAR
)
RETURNS VOID AS $$
DECLARE
    v_train_number INTEGER;
BEGIN
    INSERT INTO regular_schedule (arrival_time, departure_time, train_arrival_point, train_departure_point)
    VALUES (p_arrival_time, p_departure_time, p_train_arrival_point, p_train_departure_point)
    RETURNING train_number INTO v_train_number;
    INSERT INTO train (train_name, train_number, fact_time_sent, fact_time_transformation, arrival_date, departure_date, type_train)
    VALUES (p_train_name, v_train_number, CURRENT_TIME, CURRENT_TIME, CURRENT_DATE, CURRENT_DATE, 'suburban');
END;
$$ LANGUAGE plpgsql;

```

Data Output | Сообщения | Notifications

```
CREATE FUNCTION

Запрос завершён успешно, время выполнения: 41 msec.
```

```sql
CREATE OR REPLACE FUNCTION calculate_daily_revenue()
RETURNS INTEGER AS $$
DECLARE
    v_total_revenue INTEGER;
BEGIN
    SELECT SUM(price_ticket) INTO v_total_revenue
    FROM ticket
    WHERE date_of_purchase = CURRENT_DATE;

    RETURN v_total_revenue;
END;
$$ LANGUAGE plpgsql;
```

Data Output | Сообщения | Notifications

```
CREATE FUNCTION

Запрос завершён успешно, время выполнения: 39 msec.
```

Создание триггера для логирования событий вставки, удаления и редактирования данных в базе данных PostgreSQL

Создание триггера и функции для таблицы regular_schedule:

```sql
CREATE TABLE data_log (
    log_id SERIAL PRIMARY KEY,
    table_name VARCHAR(50) NOT NULL,
    action_type VARCHAR(20) NOT NULL,
    action_timestamp TIMESTAMP NOT NULL,
    old_data JSONB,
    new_data JSONB
);

CREATE OR REPLACE FUNCTION log_data_changes()
RETURNS TRIGGER AS $$
DECLARE
    v_old_data JSONB;
    v_new_data JSONB;
BEGIN
    IF TG_OP = 'DELETE' THEN
        v_old_data = row_to_json(OLD);
        INSERT INTO data_log (table_name, action_type, action_timestamp, old_data)
        VALUES (TG_TABLE_NAME, TG_OP, current_timestamp, v_old_data);
```

```
    ELSIF TG_OP = 'UPDATE' THEN
        v_old_data = row_to_json(OLD);
        v_new_data = row_to_json(NEW);
        INSERT INTO data_log (table_name, action_type, action_timestamp, old_data, new_data)
        VALUES (TG_TABLE_NAME, TG_OP, current_timestamp, v_old_data, v_new_data);
    ELSIF TG_OP = 'INSERT' THEN
        v_new_data = row_to_json(NEW);
        INSERT INTO data_log (table_name, action_type, action_timestamp, new_data)
        VALUES (TG_TABLE_NAME, TG_OP, current_timestamp, v_new_data);
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE TRIGGER regular_schedule_trigger
AFTER INSERT OR UPDATE OR DELETE ON regular_schedule
FOR EACH ROW
EXECUTE FUNCTION log_data_changes();
```

```
CREATE TRIGGER

Запрос завершён успешно, время выполнения: 44 msec.
```

Выводы:

По результатам данной лабораторной работы были получены навыки создания функций, процедур и триггеров в PostgreSQL, созданы необходимые функции в соответствии с заданием, а также авторский триггер.