

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: Гнеушев В. А.

Факультет: ИКТ

Группа: K3239

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

| | |
|----------------------------|---|
| Цель работы | 3 |
| Практическое задание | 3 |
| Описание проекта..... | 3 |
| Функции..... | 3 |
| Триггер..... | 5 |
| Вывод..... | 6 |

Цель работы

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

Практическое задание

Вариант 1 (маx - 6 баллов)

1. Создать процедуры/функции 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.

Вариант 2 (маx - 8 баллов)

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

Описание проекта

В этом отчете я буду описывать функции и триггер для спроектированной и созданной мной на работе базе данных.

Система, для которой создавалась база данных – это обучающая платформа, интегрированная в Telegram бота. У пользователей платформы есть 3 роли – ученик, учитель и администратор.

Ученики получают доступ к образовательной платформе, оплатив обучение и авторизовавшись в Telegram боте при помощи почты, указанной при оплате.

К отчету прилагается модель ."erwin" базы данных, с которой я работал.

Функции

Функция 1 – Получение суммарной чистой прибыли ученика за весь курс

```
CREATE OR REPLACE FUNCTION student_net_income(telegram_id VARCHAR)
RETURNS DECIMAL AS
$code$
    SELECT COALESCE(SUM(investment - revenue), 0)
    FROM user_report
    WHERE telegram_user_id = student_net_income.telegram_id;
$code$
LANGUAGE SQL;
```

```
postgres=# select id, student_net_income(id) as income from telegram_user order by income desc;
```

| id | income |
|-------------|--------|
| 243816260 | 550 |
| 987654321 | 0 |
| 818181818 | 0 |
| 243816261 | 0 |
| 243816262 | 0 |
| 243816263 | 0 |
| 243816264 | 0 |
| 243816265 | 0 |
| 243816266 | 0 |
| 243816267 | 0 |
| 243816268 | 0 |
| 243816269 | 0 |
| 2438162601 | 0 |
| 2438162602 | 0 |
| 2438162603 | 0 |
| 2438162604 | 0 |
| 2438162605 | 0 |
| 2438162606 | 0 |
| 2438162607 | 0 |
| 2438162608 | 0 |
| 2438162609 | 0 |
| 24381626011 | 0 |
| 24381626012 | 0 |

Функция 2 – Получение количества покупок курса

```
CREATE OR REPLACE FUNCTION amount_course_purchases() RETURNS
INTEGER AS
$code$
DECLARE
    total_count INTEGER;
BEGIN
    SELECT COUNT(email) INTO total_count
    FROM user_status_by_email
    WHERE status = 'active';

    RETURN total_count;
END;
$code$
LANGUAGE plpgsql;
```

```
postgres=# select amount_course_purchases();
```

| amount_course_purchases |
|-------------------------|
| 2 |

(1 row)

Функция 3 – Удаление пользователя из админов (обнуление привилегий)

```
CREATE OR REPLACE PROCEDURE remove_admin_permissions(IN telegram_id
VARCHAR) AS
$code$
BEGIN
    UPDATE telegram_user SET role = NULL WHERE id = telegram_id and
role = 'admin';
END;
$code$
LANGUAGE plpgsql;
```

```
postgres=# select id, role from telegram_user where id = '243816260';
   id      | role
-----+-----
 243816260 | admin
(1 row)

postgres=# call remove_admin_permissions('243816260');
CALL
postgres=# select id, role from telegram_user where id = '243816260';
   id      | role
-----+-----
 243816260 |
(1 row)
```

Триггер

Триггер для удаления привилегий пользователя, после того как он был удален из таблицы админов.

```
CREATE OR REPLACE TRIGGER remove_admin
AFTER DELETE ON admin
FOR EACH ROW
EXECUTE PROCEDURE remove_admin_trigger_function();
```

Триггер использует вспомогательную функцию, которая вызовет функцию *remove_admin_permissions*:

```
CREATE OR REPLACE FUNCTION remove_admin_trigger_function()
RETURNS TRIGGER AS
$code$
BEGIN
    CALL remove_admin_permissions(OLD.telegram_user_id);
    RETURN NULL;
```

```
END;  
$code$  
LANGUAGE plpgsql;
```

```
postgres=# select id, role from telegram_user where id = '243816260';  
   id   | role  
-----+-----  
 243816260 | admin  
(1 row)  
  
postgres=# delete from admin where telegram_user_id = '243816260';  
DELETE 1  
postgres=# select id, role from telegram_user where id = '243816260';  
   id   | role  
-----+-----  
 243816260 |  
(1 row)
```

Вывод

В ходе данной лабораторной работы я научился создавать и использовать функции и триггеры в PostgreSQL.