Практическое занятие “Работа с триггерами и функциями” (Полухин Александр К3241)

Команда для корректного отображения кириллицы

```
postgres=# \! chcp 1251
Текущая кодовая страница: 1251
```

Задание №1

- Создание emp_time и подключение к базе данных

```
postgres=# CREATE DATABASE emp_time;
CREATE DATABASE
postgres=# \c emp_time;
Вы подключены к базе данных "emp_time" как пользователь "postgres".
```

- Проверка активных баз данных

```
postgres=# \l
                                                    Список баз данных
       Имя        | Владелец | Кодировка | Провайдер локали |     LC_COLLATE      |      LC_CTYPE       | локаль ICU | Правила ICU |
    Права доступа
-------------------+----------+-----------+------------------+---------------------+---------------------+------------+-------------+
----------------------
 Passenger         | postgres | UTF8      | libc             | Russian_Russia.1251 | Russian_Russia.1251 |            |             |
 Passenger_restore | postgres | UTF8      | libc             | Russian_Russia.1251 | Russian_Russia.1251 |            |             |
 emp_time          | postgres | UTF8      | libc             | Russian_Russia.1251 | Russian_Russia.1251 |            |             |
 postgres          | postgres | UTF8      | libc             | Russian_Russia.1251 | Russian_Russia.1251 |            |             |
 template0         | postgres | UTF8      | libc             | Russian_Russia.1251 | Russian_Russia.1251 |            |             |
 =c/postgres         +
                   |          |           |                  |                     |                     |            |             |
 postgres=CTc/postgres
 template1         | postgres | UTF8      | libc             | Russian_Russia.1251 | Russian_Russia.1251 |            |             |
 =c/postgres         +
                   |          |           |                  |                     |                     |            |             |
 postgres=CTc/postgres
(6 строк)
```

- Создание таблицы employee

```
emp_time=# CREATE TABLE employee (
emp_time(#     id SERIAL PRIMARY KEY,
emp_time(#     username VARCHAR NOT NULL
emp_time(# );
CREATE TABLE
```

- Создание таблицы time_punch

```
emp_time=# CREATE TABLE time_punch (
emp_time(#     id SERIAL PRIMARY KEY,
emp_time(#     employee_id INT NOT NULL,
emp_time(#     is_out_punch BOOLEAN DEFAULT false,
emp_time(#     punch_time TIMESTAMP DEFAULT NOW(),
emp_time(#     FOREIGN KEY (employee_id) REFERENCES employee(id)
emp_time(# );
CREATE TABLE
```

- Добавление рабочих данных в таблицу

```
emp_time=# INSERT INTO employee (username) VALUES ('Михаил');
INSERT 0 1
emp_time=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES
emp_time-# (1, false, '2021-01-01 10:00:00'),
emp_time-# (1, true, '2021-01-01 11:30:00');
INSERT 0 2
```

- Проверка рабочих данных в таблице

```
emp_time=# SELECT * FROM employee;
 id | username
----+----------
  1 | Михаил
(1 строка)


emp_time=# SELECT * FROM time_punch;
 id | employee_id | is_out_punch |     punch_time
----+-------------+--------------+---------------------
  1 |           1 | f            | 2021-01-01 10:00:00
  2 |           1 | t            | 2021-01-01 11:30:00
(2 строки)
```

- Вычисление время работы сотрудника на примере Михаила

```
emp_time=# SELECT tp1.punch_time - tp2.punch_time AS time_worked
emp_time-# FROM time_punch tp1
emp_time-# JOIN time_punch tp2 ON tp2.id = (
emp_time(#     SELECT tps.id
emp_time(#     FROM time_punch tps
emp_time(#     WHERE tps.id < tp1.id
emp_time(#     AND tps.employee_id = tp1.employee_id
emp_time(#     AND NOT tps.is_out_punch
emp_time(#     ORDER BY tps.id DESC
emp_time(#     LIMIT 1
emp_time(# )
emp_time-# WHERE tp1.employee_id = 1
emp_time-# AND tp1.is_out_punch;
 time_worked
-------------
 01:30:00
(1 строка)
```

Задание №2

- Создание функции

```
emp_time=# CREATE OR REPLACE FUNCTION fn_check_time_punch() RETURNS TRIGGER AS $$
emp_time$# BEGIN
emp_time$#     IF NEW.is_out_punch = (
emp_time$#             SELECT tps.is_out_punch
emp_time$#             FROM time_punch tps
emp_time$#             WHERE tps.employee_id = NEW.employee_id
emp_time$#             ORDER BY tps.id DESC
emp_time$#             LIMIT 1
emp_time$#         ) THEN
emp_time$#         RETURN NULL;
emp_time$#     END IF;
emp_time$#
emp_time$#     RETURN NEW;
emp_time$# END;
emp_time$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

- Создание триггера для функции

```
emp_time=# CREATE TRIGGER check_time_punch
emp_time-# BEFORE INSERT ON time_punch
emp_time-# FOR EACH ROW
emp_time-# EXECUTE FUNCTION fn_check_time_punch();
CREATE TRIGGER
```

- Проверка работы на ошибочных данных

```
emp_time=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES
emp_time-#     ((SELECT id FROM employee WHERE username = 'Михаил'), TRUE, '2021-01-01 12:00');
INSERT 0 0
emp_time=# SELECT * FROM time_punch;
 id | employee_id | is_out_punch |     punch_time
----+-------------+--------------+---------------------
  1 |           1 | f            | 2021-01-01 10:00:00
  2 |           1 | t            | 2021-01-01 11:30:00
(2 строки)
```

Задание №3

- Добавление корректных данных для проверки триггера

```
emp_time=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES
emp_time-#     ((SELECT id FROM employee WHERE username = 'Михаил'), FALSE, '2021-01-02 09:00'),
emp_time-#     ((SELECT id FROM employee WHERE username = 'Михаил'), TRUE, '2021-01-02 18:00');
INSERT 0 2
```

- Добавление некорректных данных для проверки триггера

```
emp_time=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES
emp_time-#     ((SELECT id FROM employee WHERE username = 'Михаил'), FALSE, '2021-01-03 10:00'),
emp_time-#     ((SELECT id FROM employee WHERE username = 'Михаил'), FALSE, '2021-01-03 10:30');
INSERT 0 1
```

- Просмотр данных в таблице учёта времени

```
emp_time=# SELECT * FROM time_punch;
 id | employee_id | is_out_punch |     punch_time
----+-------------+--------------+---------------------
  1 |           1 | f            | 2021-01-01 10:00:00
  2 |           1 | t            | 2021-01-01 11:30:00
  4 |           1 | f            | 2021-01-02 09:00:00
  5 |           1 | t            | 2021-01-02 18:00:00
  6 |           1 | f            | 2021-01-03 10:00:00
(5 строк)
```

Задание №4 и 5

- Создана таблица для хранения логов `logs` с определённой структурой

```
emp_time=# CREATE TABLE logs (
emp_time(#     text TEXT,
emp_time(#     added TIMESTAMP WITHOUT TIME ZONE
emp_time(# );
CREATE TABLE
```

- Создание функции триггера

```
emp_time=# CREATE OR REPLACE FUNCTION add_to_log() RETURNS TRIGGER AS $$
emp_time$# DECLARE
emp_time$#     mstr VARCHAR(30);
emp_time$#     astr VARCHAR(100);
emp_time$#     retstr VARCHAR(254);
emp_time$# BEGIN
emp_time$#     IF TG_OP = 'INSERT' THEN
emp_time$#         astr := NEW.id::TEXT;
emp_time$#         mstr := 'Add new user ';
emp_time$#         retstr := mstr || astr;
emp_time$#         INSERT INTO logs (text, added) VALUES (retstr, NOW());
emp_time$#         RETURN NEW;
emp_time$#     ELSIF TG_OP = 'UPDATE' THEN
emp_time$#         astr := NEW.id::TEXT;
emp_time$#         mstr := 'Update user ';
emp_time$#         retstr := mstr || astr;
emp_time$#         INSERT INTO logs (text, added) VALUES (retstr, NOW());
emp_time$#         RETURN NEW;
emp_time$#     ELSIF TG_OP = 'DELETE' THEN
emp_time$#         astr := OLD.id::TEXT;
emp_time$#         mstr := 'Remove user ';
emp_time$#         retstr := mstr || astr;
emp_time$#         INSERT INTO logs (text, added) VALUES (retstr, NOW());
emp_time$#         RETURN OLD;
emp_time$#     END IF;
emp_time$# END;
emp_time$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

- Создание триггера

```
emp_time=# CREATE TRIGGER t_employee
emp_time-# AFTER INSERT OR UPDATE OR DELETE ON employee
emp_time-# FOR EACH ROW
emp_time-# EXECUTE FUNCTION add_to_log();
CREATE TRIGGER
```

- Добавление и обновление данных

```
emp_time=# INSERT INTO employee (username) VALUES ('Новый сотрудник');
INSERT 0 1
emp_time=# UPDATE employee SET username = 'Обновленное имя' WHERE id = 1;
UPDATE 1
```

- Удаление данных

```
emp_time=# DELETE FROM employee WHERE id = 1;
DELETE 1
```

- Содержание таблицы логов

```
emp_time=# SELECT * FROM logs;
      text       |           added
-----------------+----------------------------
 Add new user 2  | 2024-03-04 23:05:18.641733
 Update user 1   | 2024-03-04 23:05:26.641447
 Remove user 1   | 2024-03-04 23:07:20.091143
(3 строки)
```