

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: Даньшин С. А.

Факультет: ИКТ

Группа: K3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

Ход работы:.....	4
Вывод	11

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 10. БД «Автовокзал»

Задание 4. Создать хранимые процедуры:

- ☐ Продажи билета.
- ☐ Возврата билета.
- ☐ Добавления нового рейса.

Задание 5. Создать необходимые триггеры.

Ход работы:

1. Продажи билета

```
CREATE or replace PROCEDURE TicketSale (  
    IN p_passenger_id INT,  
    IN p_trip_id INT,  
    IN p_seat_number INT,  
    IN p_start_station_id INT,  
    IN p_end_station_id INT,  
    IN p_is_online_sale BOOLEAN  
)  
LANGUAGE plpgsql  
AS $$  
begin  
    RAISE NOTICE 'Input Parameters: %, %, %, %, %, %',  
p_passenger_id, p_trip_id, p_seat_number, p_start_station_id,  
p_end_station_id, p_is_online_sale;  
    INSERT INTO tickets (passenger_id, trip_id, seat_number,  
status, start_station_id, end_station_id, is_online_sale, sold_at)  
    VALUES (p_passenger_id, p_trip_id, p_seat_number, 'PAID',  
p_start_station_id, p_end_station_id, p_is_online_sale, NOW());  
END;  
$$;
```

Вызов с учетом функции на добавление поездок:

```
DO $$  
DECLARE  
    v_trip_id INT;  
BEGIN  
  
    SELECT id INTO v_trip_id  
    FROM trips  
    WHERE route_id=1  
    AND start_time='2023-01-01 12:00:00'  
    LIMIT 1;  
  
    IF v_trip_id IS NOT NULL THEN  
        CALL TicketSale(1, v_trip_id, 5, 1, 2, TRUE);  
    ELSE  
        RAISE NOTICE 'Ticket not found based on the criteria.';  
    END IF;  
END $$;
```

Проверка:

```
select * from tickets where seat_number=5;
```

id	passenger_id	trip_id	seat_number	status	start_station_id	end_station_id
16	1	1 000 006	5	PAID	1	2

2. Возврата билета

```
CREATE OR REPLACE PROCEDURE RefundTicket (  
    IN p_ticket_id INT  
)  
LANGUAGE plpgsql  
AS $$  
BEGIN  
    UPDATE tickets  
    SET  
        status = 'CANCELED',  
        is_online_sale = FALSE  
    WHERE  
        id = p_ticket_id;  
  
END;  
$$;
```

Вызов с поиском билета по номеру путешествия и номеру места:

```
DO $$  
DECLARE  
    v_ticket_id INT;  
BEGIN  
    -- Step 1: Select id based on criteria  
    SELECT id INTO v_ticket_id  
    FROM tickets  
    WHERE trip_id = 123  
    AND seat_number = 5  
    LIMIT 1;  
  
    -- Step 2: Call the RefundTicket procedure with the selected  
id  
    IF v_ticket_id IS NOT NULL THEN  
        CALL RefundTicket(v_ticket_id);  
        RAISE NOTICE 'Refunded Ticket ID: %', v_ticket_id;  
    ELSE  
        RAISE NOTICE 'Ticket not found based on the criteria.';  
    END IF;  
END $$;
```

Проверка:

```
select * from tickets where seat_number=5;
```

id	passenger_id	trip_id	seat_number	status	start_station_id	end_station_id
16	1	1 000 006	5	PAID	1	2

3. Добавления нового рейса

```
CREATE OR REPLACE PROCEDURE AddNewTrip (  
    IN p_route_id INT,  
    IN p_bus_id INT,  
    IN p_start_time TIMESTAMP,  
    IN p_end_time TIMESTAMP,  
    IN p_status tripstatus,  
    IN p_price INT  
)  
LANGUAGE plpgsql  
AS $$  
BEGIN  
    INSERT INTO trips (route_id, bus_id, start_time, end_time,  
status, price)  
    VALUES (p_route_id, p_bus_id, p_start_time, p_end_time,  
p_status, p_price);  
END;  
$$;
```

Вызов:

```
CALL AddNewTrip(1, 2, '2023-01-01 12:00:00', '2023-01-01 16:00:00',  
'PLANNED', 200);
```

Проверка:

```
select * from trips where route_id=1 and status='PLANNED' and bus_id=2  
and start_time='2023-01-01 12:00:00';
```

	id	route_id	bus_id	start_time	end_time	status	price
1	1 000 006	1	2	2023-01-01 12:00:00.000	2023-01-01 16:00:00.000	PLANNED	200

4. Модифицировать триггер на проверку корректности входа и выхода сотрудника

(имеющиеся проблемы: может быть отрицательное время работы, человек зашел/вышел в будущем)

```
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.punch_time > now()
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 проверки при вставке билета, что место свободно:

```
CREATE OR REPLACE FUNCTION check_seat_availability()
RETURNS TRIGGER AS $$
DECLARE
    v_seat_count INT;
BEGIN
    SELECT COUNT(*)
    INTO v_seat_count
    FROM tickets
    WHERE trip_id = NEW.trip_id
        AND seat_number = NEW.seat_number
        AND status = 'PAID';

    IF v_seat_count > 0 THEN
        RAISE EXCEPTION 'Seat % for Trip % is not available.',
NEW.seat_number, NEW.trip_id;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER check_seat_availability_trigger
BEFORE INSERT ON tickets
FOR EACH ROW EXECUTE FUNCTION check_seat_availability();
```

Проверка:

```
INSERT INTO tickets (passenger_id, trip_id, seat_number, status,
start_station_id, end_station_id, is_online_sale, sold_at)
VALUES (1, 1000006, 5, 'PAID', 1, 2, true, NOW());
```



SQL Error [P0001]: ERROR:
Seat 5 for Trip 1000006 is not
available.
Где: PL/pgSQL function
check_seat_availability() line
13 at RAISE

Сведения >>



Показать панель

Вывод

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