

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение  
высшего образования  
«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

**Отчет по лабораторной работе № 5**  
**«Процедуры, функции, триггеры в PostgreSQL»**  
**по дисциплине:**  
**«Проектирование и реализация баз данных»**

**Выполнила:**  
студентка II курса ИКТ  
группы К3240  
Никифорова Кюннэй Васильевна

**Проверила:**  
*Говорова Марина Михайловна*

Санкт-Петербург  
2022

**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. 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.

**Наименование БД:** «Служба заказа такси»

**Задание 4.** Создать хранимые процедуры:

- Для вывода данных о пассажирах, которые заказывали такси в заданном, как параметр, временном интервале.
- Вывести сведения о том, куда был доставлен пассажир по заданному номеру телефона пассажира.
- Для вычисления суммарного дохода таксопарка за истекший месяц.

**Задание 5.** Создать необходимые триггеры.

## Выполнение:

### I. Хранимые процедуры

- Для вывода данных о пассажирах, которые заказывали такси в заданном, как параметр, временном интервале.

#### Создание функции:

```
1 CREATE OR REPLACE FUNCTION pass_info(begin_interval date, end_interval date)
2 RETURNS TABLE(id int, phone_number varchar(12))
3 AS
4 $$
5 BEGIN
6     RETURN QUERY
7     SELECT passenger.id_passenger, passenger.passenger_phone_number
8     FROM passenger, taxi_call
9     WHERE passenger.id_passenger = taxi_call.id_passenger
10    AND taxi_call.call_date BETWEEN begin_interval AND end_interval;
11 END;
12 $$ LANGUAGE plpgsql;
```

```
taxi_db=# CREATE OR REPLACE FUNCTION pass_info(begin_interval date, end_interval date)
taxi_db=# RETURNS TABLE(id int, phone_number varchar(12))
taxi_db=# AS
taxi_db=# $$
taxi_db$# BEGIN
taxi_db$# RETURN QUERY
taxi_db$# SELECT passenger.id_passenger, passenger.passenger_phone_number
taxi_db$# FROM passenger, taxi_call
taxi_db$# WHERE passenger.id_passenger = taxi_call.id_passenger
taxi_db$# AND taxi_call.call_date BETWEEN begin_interval AND end_interval;
taxi_db$# END;
taxi_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

#### Вызов функции:

```
SELECT * FROM pass_info('2022-02-01','2022-03-31')
```

```
taxi_db=# SELECT * FROM pass_info('2022-02-01','2022-03-31');
 id | phone_number
----+-----
 16 | +79869338925
 13 | +79342045795
(2 строки)
```

- Вывести сведения о том, куда был доставлен пассажир по заданному номеру телефона пассажира.

Создание функции:

```

1 CREATE OR REPLACE FUNCTION pass_destination(phone varchar(12))
2 RETURNS TABLE(pass_phone_number varchar(12), where_to varchar(40))
3 AS
4 $$
5 BEGIN
6     RETURN QUERY
7     SELECT passenger.passenger_phone_number, taxi_call.where_to FROM passenger
8     JOIN taxi_call ON passenger.id_passenger = taxi_call.id_passenger
9     WHERE passenger.passenger_phone_number = phone;
10 END;
11 $$ LANGUAGE plpgsql;

```

```

taxi_db=# CREATE OR REPLACE FUNCTION pass_destination(phone varchar(12))
taxi_db=# RETURNS TABLE(pass_phone_number varchar(12), where_to varchar(40))
taxi_db=# AS
taxi_db=# $$
taxi_db$# BEGIN
taxi_db$# RETURN QUERY
taxi_db$# SELECT passenger.passenger_phone_number, taxi_call.where_to FROM passenger
taxi_db$# JOIN taxi_call ON passenger.id_passenger = taxi_call.id_passenger
taxi_db$# WHERE passenger.passenger_phone_number = phone;
taxi_db$# END;
taxi_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION

```

Вызов функции:

```
SELECT * FROM pass_destination('+79342045795')
```

```

taxi_db=# SELECT * FROM pass_destination('+79342045795');
 pass_phone_number |      where_to
-----+-----
+79342045795      | Nevsky Avenue 6
(1 строка)

```

- Для вычисления суммарного дохода таксопарка за истекший месяц.

Создание функции:

```
1 CREATE OR REPLACE FUNCTION income()
2 RETURNS TABLE(total_income |int)
3 AS
4 $$
5 BEGIN
6     RETURN QUERY
7     SELECT SUM(taxi_call.payment)
8     FROM taxi_call
9     WHERE EXTRACT(month FROM taxi_call.call_date) = EXTRACT(month FROM current_date) - 1;
10 END;
11 $$ LANGUAGE plpgsql;
```

```
taxi_db=# CREATE OR REPLACE FUNCTION income()
taxi_db=# RETURNS TABLE(total_income bigint)
taxi_db=# AS
taxi_db=# $$
taxi_db$# BEGIN
taxi_db$# RETURN QUERY
taxi_db$# SELECT SUM(taxi_call.payment)
taxi_db$# FROM taxi_call
taxi_db$# WHERE EXTRACT(month FROM taxi_call.call_date) = EXTRACT(month FROM current_date) - 1;
taxi_db$# END;
taxi_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

Вызов функции:

```
SELECT * FROM total_income();
```

```
taxi_db=# SELECT * FROM income();
total_income
-----
          719
(1 строка)
```

## II. Необходимые триггеры

- 1) Создание таблицы *logs*
- 2) Создание триггерной функции

```
1 CREATE OR REPLACE FUNCTION add_to_log() RETURNS TRIGGER AS $$
2 DECLARE
3     mstr varchar(30);
4     astr varchar(100);
5     retstr varchar(254);
6 BEGIN
7     IF TG_OP = 'INSERT' THEN
8         astr = NEW;
9         mstr := 'Add new';
10        retstr := mstr || astr;
11        INSERT INTO public.logs(text,added,table_name)
12        values (retstr,NOW(), TG_TABLE_NAME);
13        RETURN NEW;
14    ELSIF TG_OP = 'UPDATE' THEN
15        astr = NEW;
16        mstr := 'Update';
17        retstr := mstr || astr;
18        INSERT INTO public.logs(text,added,table_name)
19        values (retstr,NOW(), TG_TABLE_NAME);
20        RETURN NEW;
21    ELSIF TG_OP = 'DELETE' THEN
22        astr = OLD;
23        mstr := 'Remove';
24        retstr := mstr || astr;
25        INSERT INTO public.logs(text,added,table_name)
26        values (retstr,NOW(), TG_TABLE_NAME);
27        RETURN OLD;
28    END IF;
29 END;
30 $$ LANGUAGE plpgsql;
```

```

taxi_db=# CREATE OR REPLACE FUNCTION add_to_log() RETURNS TRIGGER AS $$
taxi_db$# DECLARE
taxi_db$# mstr varchar(30);
taxi_db$# astr varchar(100);
taxi_db$# retstr varchar(254);
taxi_db$# BEGIN
taxi_db$# IF TG_OP = 'INSERT' THEN
taxi_db$# astr = NEW;
taxi_db$# mstr := 'Add new';
taxi_db$# retstr := mstr || astr;
taxi_db$# INSERT INTO public.logs(text,added,table_name)
taxi_db$# values (retstr,NOW(), TG_TABLE_NAME);
taxi_db$# RETURN NEW;
taxi_db$# ELSIF TG_OP = 'UPDATE' THEN
taxi_db$# astr = NEW;
taxi_db$# mstr := 'Update';
taxi_db$# retstr := mstr || astr;
taxi_db$# INSERT INTO public.logs(text,added,table_name)
taxi_db$# values (retstr,NOW(), TG_TABLE_NAME);
taxi_db$# RETURN NEW;
taxi_db$# ELSIF TG_OP = 'DELETE' THEN
taxi_db$# astr = OLD;
taxi_db$# mstr := 'Remove';
taxi_db$# retstr := mstr || astr;
taxi_db$# INSERT INTO public.logs(text,added,table_name)
taxi_db$# values (retstr,NOW(), TG_TABLE_NAME);
taxi_db$# RETURN OLD;
taxi_db$# END IF;
taxi_db$# END;
taxi_db$# $$ LANGUAGE plpgsql;
CREATE FUNCTION

```

### 3) Создание триггера

```

1 CREATE TRIGGER t_driver AFTER INSERT OR UPDATE OR DELETE ON
2 public.driver FOR EACH ROW EXECUTE PROCEDURE add_to_log ();

```

```

taxi_db=# CREATE TRIGGER t_driver AFTER INSERT OR UPDATE OR DELETE ON
taxi_db=# public.driver FOR EACH ROW EXECUTE PROCEDURE add_to_log ();
CREATE TRIGGER

```

### 4) Проверка

```

1 INSERT INTO public.driver (id_driver, driver_name, driver_address,
2 driver_phone_number, driver_category, driver_passport) OVERRIDING SYSTEM VALUE VALUES
3 (9, 'Kovalenko ALbert', 'Kosygina 22', '+79542657928', 'C', '9811345622');
4
5 UPDATE public.driver set driver_phone_number = '+79023865489' where id_driver = 9;
6
7 DELETE FROM public.driver where id_driver = 9;

```

```

taxi_db=# INSERT INTO public.driver (id_driver, driver_name, driver_address,
taxi_db=# driver_phone_number, driver_category, driver_passport) OVERRIDING SYSTEM VALUE VALUES
taxi_db=# (9, 'Kovalenko ALbert', 'Kosygina 22', '+79542657928', 'C', '9811345622');
INSERT 0 1
taxi_db=#
taxi_db=# UPDATE public.driver set driver_phone_number = '+79023865489' where id_driver = 9;
UPDATE 1
taxi_db=#
taxi_db=# DELETE FROM public.driver where id_driver = 9;
DELETE 1

```

```
SELECT * FROM public.logs;
```

```

taxi_db=# SELECT * FROM public.logs;

```

text	added	table_name
Add new(9,"Kovalenko ALbert","Kosygina 22",+79542657928,C,9811345622)	2022-05-05 03:21:38.224953	driver
Update(9,"Kovalenko ALbert","Kosygina 22",+79023865489,C,9811345622)	2022-05-05 03:21:38.233451	driver
Remove(9,"Kovalenko ALbert","Kosygina 22",+79023865489,C,9811345622)	2022-05-05 03:22:01.46636	driver

(3 строки)

## Выводы:

В результате выполненной работы:

- Изучены и созданы процедуры/функции;
- Создан триггер для логирования событий вставки, удаления, редактирования данных.