Санкт-Петербургский Политехнический Университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**ОТЧЕТ**

**по лабораторной работе**

**«Хранимые процудуры»**

По дисциплине «Базы данных»

Работу выполнил студент

Группа 3530901/60202

Курякин Д. А.

Работу принял преподаватель

Мяснов А.В.

Санкт-Петербург

2020

**Содержание**

[**1.** **Цель работы** 2](#_Toc38823307)

[**2.** **Программа работы** 2](#_Toc38823308)

[**3.** **Стандартные запросы** 2](#_Toc38823309)

[**3.1.**  **Задание 1** 2](#_Toc38823310)

[**3.2.**  **Задание 2** 4](#_Toc38823311)

[**Вывод** 5](#_Toc38823312)

**1. Цель работы**

Познакомиться с обработкой данных на стороне сервера с помощью хранимых процедур.

**2. Программа работы**

1. Изучение языка PL/pg-SQL.

2. Создание хранимых процедур на стороне сервера.

3. Выкладывание скриптов в GitLab.

4. Демонстрация результатов работы преподавателю.

**3. Стандартные запросы**

**3.1. Задание 1**

По заданному водителю вывести всю информацию:

* нарушения
* полисы
* ДТП

Из-за того, что при выводе всей информации таблица стала очень большой, то было принято решение написать 4 функции по каждая из которых выводила свой тип информации: нарушения, полисы, ДТП где водитель был пострадавшим, ДТП где водитель был виновным. CREATE OR REPLACE FUNCTION *get\_driver\_data\_dtp\_guilty*(id\_driver\_ integer)  
RETURNS table(id\_driver integer, dtp\_id integer, car integer, car\_injured integer, details\_name varchar)  
AS $$  
 SELECT driver\_license.id as id\_driver, dtp.id AS dtp\_id, c1.id AS car, c2.id AS car\_injured, det.name AS details\_name  
 FROM driver\_license  
 JOIN people ON driver\_license.people\_id = people.id  
 JOIN car c1 ON people.id = c1.people\_id  
 JOIN guilty ON c1.id = guilty.guilty\_id  
 JOIN dtp ON guilty.guilty = dtp.guilty  
 JOIN injured ON dtp.injured = injured.injured  
 JOIN car c2 ON injured.injured\_id = c2.id  
 JOIN damage\_car dc on dtp.damage\_car = dc.damage\_car  
 JOIN damage dam on dc.damage = dam.damage  
 JOIN details det on dam.damage\_id = det.id  
 WHERE driver\_license.id = id\_driver\_  
  
$$  
LANGUAGE sql;  
  
select *\** from *get\_driver\_data\_dtp\_guilty*(5);  
  
CREATE OR REPLACE FUNCTION *get\_driver\_data\_dtp\_injured*(id\_driver\_ integer)  
RETURNS table(id\_driver integer, dtp\_id integer, car integer, car\_guilty integer, details\_name varchar)  
AS $$  
 SELECT driver\_license.id as id\_driver, dtp.id AS dtp\_id, c1.id AS car, c2.id AS car\_guilty, det.name AS details\_name  
 FROM driver\_license  
 JOIN people ON driver\_license.people\_id = people.id  
 JOIN car c1 ON people.id = c1.people\_id  
 JOIN injured ON c1.id = injured.injured\_id  
 JOIN dtp ON injured.injured = dtp.injured  
 JOIN guilty ON dtp.guilty = guilty.guilty  
 JOIN car c2 ON guilty.guilty\_id = c2.id  
 JOIN damage\_car dc on dtp.damage\_car = dc.damage\_car  
 JOIN damage dam on dc.damage = dam.damage  
 JOIN details det on dam.damage\_id = det.id  
 WHERE driver\_license.id = id\_driver\_  
$$  
LANGUAGE sql;  
  
select *\** from *get\_driver\_data\_dtp\_injured*(5);  
  
CREATE OR REPLACE FUNCTION *get\_driver\_data\_fine*(id\_driver\_ integer)  
RETURNS table(id\_driver integer, fine\_id integer, fine\_police\_certificate integer, fine\_data\_and\_time TIMESTAMP, fine\_id\_violation integer, violation\_title text, violation\_punishment text)  
AS $$  
 SELECT driver\_license.id as id\_driver, fine.id AS fine\_id,  
 fine.police\_certificate AS fine\_police\_certificate, fine.data\_and\_time AS fine\_data\_and\_time, fine.id\_violation AS fine\_id\_violation, violation.title AS violation\_title, violation.punishment AS violation\_punishment  
 FROM driver\_license  
 JOIN fine ON driver\_license.id = fine.driver\_license  
 JOIN violation ON fine.id\_violation = violation.id  
 WHERE driver\_license.id = id\_driver\_  
  
$$  
LANGUAGE sql;  
  
select *\** from *get\_driver\_data\_fine*(5);  
  
CREATE OR REPLACE FUNCTION *get\_driver\_data\_osago*(id\_driver\_ integer)  
RETURNS table(id\_driver integer,  
 osago\_id integer,  
 osago\_data\_and\_time\_of\_issue TIMESTAMP,  
 osago\_end\_date\_and\_time TIMESTAMP,  
 osago\_car varchar(12)  
 )  
AS $$  
 SELECT driver\_license.id as id\_driver,  
 osago.id AS osago\_id,  
 osago.data\_and\_time\_of\_issue AS osago\_data\_and\_time\_of\_issue,  
 osago.end\_date\_and\_time AS osago\_end\_date\_and\_time,  
 osago.car AS osago\_car  
 FROM driver\_license  
 JOIN people ON driver\_license.people\_id = people.id  
 JOIN car ON people.id = car.people\_id  
 JOIN osago ON car.registration\_plate = osago.car  
 WHERE driver\_license.id = id\_driver\_  
  
$$  
LANGUAGE sql;  
  
select *\** from *get\_driver\_data\_osago*(5);

**3.2. Задание 2**

По имеющемуся водительскому удостоверению добавить его копию с перечнем категорий, если срок действия уже истек на данный момент.

Создаем временную таблицы tmp и заполняем ее. Сравниваем дату окончания водительского удостоверения с сегодняшней датой. Если время действия водительского удостоверения истекло, то копируем водительское удостоверение и создаем новый уникальный ключ для категорий, так же копируем все категории с новым ключом. Если время действия истекло то выходим из процедуры.

drop procedure if exists driver\_copy;  
create procedure driver\_copy(id\_driver\_ integer)  
as $$  
 DECLARE rand\_categ bigint;  
 newdate timestamp;  
 i bigint;  
 max\_rand integer;  
 min\_rand integer;  
begin  
  
 max\_rand = 100000000;  
 min\_rand = 1000000;  
 drop table if exists tmp;  
  
 create temporary table tmp  
 (  
 "id" serial,  
 "number" integer,  
 "categories" integer,  
 "data\_and\_time\_of\_issue" TIMESTAMP,  
 "end\_date\_and\_time" TIMESTAMP,  
 "unit\_gipdd" varchar(128),  
 "people\_id" integer  
 );  
 INSERT INTO tmp (select \* from driver\_license);  
  
 newdate = statement\_timestamp();  
 if exists(select \* from tmp where tmp.id = id\_driver\_ and tmp.end\_date\_and\_time < newdate) then  
 i = min\_rand;  
 loop  
 rand\_categ = (SELECT floor(random() \* max\_rand + min\_rand));  
 if not exists(select \* from tmp where categories = rand\_categ) then  
 INSERT INTO driver\_license (number, categories, data\_and\_time\_of\_issue, end\_date\_and\_time, unit\_gipdd, people\_id)  
 select tmp.number, rand\_categ, newdate, newdate + '10 YEAR', tmp.unit\_gipdd, tmp.people\_id  
 from tmp where id = id\_driver\_  
 limit 1;  
 INSERT INTO categories(categories, id\_categories)  
 select rand\_categ, categories.id\_categories from categories where categories.categories = (  
 select categories  
 from tmp where id = id\_driver\_  
 limit 1  
 );  
 exit;  
 end if;  
 i = i + 1;  
 if i > max\_rand then  
 exit;  
 end if;  
 end loop;  
 end if;  
end;  
$$  
 language plpgsql;  
  
call driver\_copy(1)

**Вывод**

В процессе выполнения данной лабораторной работы были изучены возможности языка PL/pgSQL. В частности, был получен опыт написания функций, которые выводили структурированную информацию с множества таблиц, был получен опыт написания процедур, вычисляемых нужные данные и создание новых строк в таблицах при нужных условиях.