МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования  
**«Вятский государственный университет»**

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

**«Пользовательские функции»**

Отчёт по лабораторной работе №3 дисциплины

«Базы данных»

Выполнил студент группы ПИб-21\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Вологжанина А.А./

Проверил преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Клюкин В. Л./

Киров 2022

**Цели лабораторной работы:**

1) Познакомиться с созданием пользовательских функций и триггеров в PostgreSQL;   
2) Освоить работу с составными типами данных и массивами;   
3) Изучить основы работы с процедурным языком PL/pgSQL.

**Задание на лабораторную работу:**   
При выполнении работы нужно использовать БД, созданную в лабораторных

работах №1 и №2. Нужно выполнить следующие шаги.

1. Для любой таблицы создать функцию save\_имя\_таблицы, которая

принимает на вход параметры, соответствующие её столбцам, и, если

переданное значение первичного ключа равно null, выполняет запрос insert,

иначе – запрос update для соответствующей строки. Функция должна вернуть

значение первичного ключа вставленной или изменённой строки.

2. Для любой таблицы, на которую имеются внешние ключи, создать

функцию delete\_имя\_таблицы, принимающую на вход значение первичного

ключа строки и ничего не возвращающую. Если на удаляемую строку

существуют ссылки, то функция должна поднимать ошибку «Невозможно

выполнить удаление, так как есть внешние ссылки».

3. Для таблицы, содержащей столбец с числовыми значениям, создать

функцию, которая принимает на вход число – минимальное значение – и

возвращает setof имя\_таблицы – множество строк, в которых значение числа

больше или равно переданному аргументу.

4. Создать составной тип, содержащий не менее 2-3 полей, по крайней

мере одно из которых должно быть числовым. Создать функцию, которая

принимает массив объектов этого типа и минимальное значение для

указанного поля. Функция должна возвращать массив отфильтрованных по

переданному значению объектов.

5. Для любой таблицы создать таблицу log\_имя\_таблицы, которая будет

содержать лог изменений по любому выбранному столбцу этой таблицы.

Для этого нужны столбцы:

- первичный ключ;

- внешний ключ на выбранную таблицу;

- дата изменения строки;

- старое значение столбца;

- новое значение столбца.

Реализовать заполнение таблицы с логом с помощью триггеров на

вставку/изменение строк.

6. Реализовать любую функцию на свой выбор, использующую для

получения результата динамически формируемый запрос.

**Ход работы:**Задание 1. Функция save\_subject.

CREATE OR REPLACE FUNCTION save\_subject (

\_id BIGINT,

name\_subject VARCHAR(50),

num\_of\_hours INT

)

RETURNS BIGINT

AS $$

DECLARE

used\_id BIGINT;

BEGIN

IF \_id IS NULL THEN

INSERT INTO subject (name\_subject, hours)

VALUES (name\_subject, hours)

RETURNING id /\* Конструкция позволяет вернуть id нового элемента \*/

INTO \_id; /\* id нового элемента записывается в переменную used\_id \*/

ELSE

UPDATE subject SET

name\_subject = name\_subject,

hours = hours

WHERE id = \_id;

\_id := \_id; /\* Нам уже известен id, поэтому просто присвоим его \*/

END IF;

RETURN \_id;

END;

$$ LANGUAGE plpgsql;

Таблица до изменений:

Изображение выглядит как стол

Автоматически созданное описание

Таблица после изменений:  
Изображение выглядит как стол

Автоматически созданное описание

Задание 2. Функция delete\_subject.

CREATE OR REPLACE FUNCTION delete\_subject (

\_id BIGINT

)

RETURNS VOID

AS $$

BEGIN

DELETE FROM subject WHERE id = \_id; /\* Просто пытаемся удалить строку \*/

EXCEPTION

WHEN foreign\_key\_violation THEN /\* Перехват нужного нам исключения \*/

RAISE EXCEPTION 'Невозможно выполнить удаление, так как есть внешние

ссылки.';

END;

$$ LANGUAGE plpgsql;

Таблица до изменений:

Изображение выглядит как стол

Автоматически созданное описание  
Таблица после изменений:

Изображение выглядит как стол

Автоматически созданное описание

Задание 3. Функция фильтрации по числовому значению.

CREATE OR REPLACE FUNCTION filter\_subject\_by\_hours (

min\_val BIGINT

)

RETURNS SETOF subject

AS $$

BEGIN

RETURN QUERY (SELECT \* FROM subject WHERE hours >= min\_val);

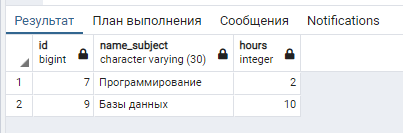
END;

$$ LANGUAGE plpgsql;

Таблица до изменений:

Изображение выглядит как стол

Автоматически созданное описание  
Таблица после изменений:



Задание 4. Функция фильтрации массива объектов

CREATE TYPE t\_subject AS (

id BIGINT,

name\_subject VARCHAR(50),

hours INTEGER

);

CREATE OR REPLACE FUNCTION filter\_array\_of\_subjects (

arr t\_subject[],

filter\_var INTEGER

)

RETURNS t\_subject[]

AS $$

BEGIN

RETURN ARRAY( /\* Преобразуем выборку в массив \*/

SELECT (id, name\_subject, hours)::t\_subject /\* Создаем таблицу из элементов

массива \*/

FROM unnest(arr)

WHERE hours >= filter\_var

);

END;

$$ LANGUAGE plpgsql;

Таблица до изменений:  
Изображение выглядит как стол

Автоматически созданное описание

Таблица после изменений:

Изображение выглядит как текст

Автоматически созданное описание

Задание 5. Таблица log\_subject.

CREATE TABLE log\_subject (

id BIGSERIAL PRIMARY KEY,

subject\_id BIGINT REFERENCES subject(id),

change\_datetime TIMESTAMP DEFAULT NOW(),

old\_value INT DEFAULT NULL,

new\_value INT DEFAULT NULL

);

CREATE OR REPLACE FUNCTION trigger\_func()

RETURNS TRIGGER

AS $$

DECLARE

old\_val INT;

BEGIN

/\* Определяем "старое" значение \*/

IF (TG\_OP = 'UPDATE') THEN

old\_val := OLD.hours;

ELSIF (TG\_OP = 'INSERT') THEN

old\_val := NULL;

end if;

/\* Производим запрос \*/

INSERT INTO log\_subject

(subject\_id, old\_value, new\_value)

VALUES

(NEW.id, old\_val, NEW.hours);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER commit\_subject\_change

AFTER UPDATE OR INSERT

ON subject

FOR EACH ROW

EXECUTE PROCEDURE trigger\_func();

Таблица до изменений:  
Изображение выглядит как стол

Автоматически созданное описание

Таблица после изменений:

Изображение выглядит как стол

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Задание 6. Создание динамически формируемого запроса.

CREATE OR REPLACE FUNCTION get\_value\_by\_id (

tableName VARCHAR,

columnName VARCHAR,

id BIGINT

)

RETURNS TEXT

AS $$

DECLARE

result TEXT;

BEGIN

EXECUTE 'SELECT '|| columnName ||' FROM '|| tableName ||' WHERE id = $1' USING id

INTO result;

RETURN result;

END;

$$ LANGUAGE plpgsql;

Таблица до изменений:  
Изображение выглядит как стол

Автоматически созданное описание

Таблица после изменений:

SELECT get\_value\_by\_id('subject', 'name\_subject', 9);

Изображение выглядит как текст

Автоматически созданное описание

**Вывод:**

В ходе выполнения данной лабораторной работы были ознакомлены с созданием пользовательских функций и триггеров в PostgreSQL, с работой с составными типами данных и массивами, а также изучены основы работы с процедурным языком PL/pgSQL