Министерство образования и науки Российской Федерации Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

**Основы DМL-запросов в PostgreSQL**

Лабораторная работа №3

по курсу «Базы данных»

Выполнил студент группы ИВТб-31 /Категов А.Д/ Проверил преподаватель /Клюкин В.Л./

Киров 2024

**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. 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**

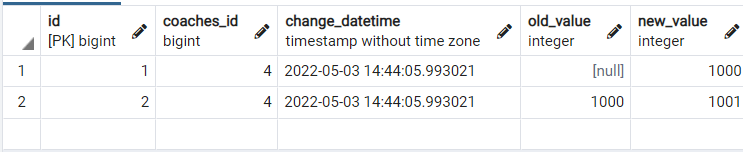


Рисунок 1 - Таблица с логом изменений в таблице

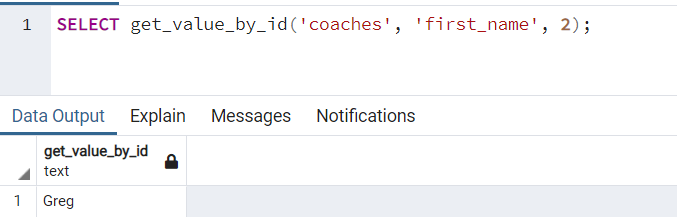


Рисунок 2 – Таблица динамически сформированного запроса

**Листинг программы**

CREATE OR REPLACE FUNCTION save\_vehicle (

\_id BIGINT,

\_model VARCHAR(30),

\_license\_plate VARCHAR(30),

\_kind vehicle\_type

)

RETURNS BIGINT

AS $$

DECLARE

used\_id BIGINT;

BEGIN

IF \_id IS NULL THEN

INSERT INTO vehicle (model, license\_plate, kind)

VALUES (\_model, \_license\_plate, \_kind)

RETURNING id /\* Вернуть id нового элемента \*/

INTO used\_id; /\* В переменную used\_id \*/

ELSE

UPDATE vehicle SET

model = \_model,

license\_plate = \_license\_plate,

kind = \_kind

WHERE id = \_id;

used\_id := \_id; /\* Присвоим известный id \*/

END IF;

RETURN used\_id;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION delete\_vehicle (

\_id BIGINT

)

RETURNS VOID

AS $$

BEGIN

DELETE FROM vehicle 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;

CREATE OR REPLACE FUNCTION filter\_instructor\_by\_price (

min\_price BIGINT

)

RETURNS SETOF instructor

AS $$

BEGIN

RETURN QUERY (SELECT \* FROM instructor WHERE price >= min\_price);

END;

$$ LANGUAGE plpgsql;

CREATE TYPE t\_exam AS (

id BIGINT,

mark int2,

id\_team bigint,

id\_inspector bigint

);

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

arr t\_exam[],

filter\_var integer

)

RETURNS t\_exam[]

AS $$

BEGIN

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

SELECT (id, mark, id\_team, id\_inspector)::t\_exam

FROM unnest(arr)

WHERE mark >= filter\_var

);

END;

$$ LANGUAGE plpgsql;

/\*CREATE TABLE log\_examinee (

id BIGSERIAL PRIMARY KEY,

examinee\_id BIGINT REFERENCES examinee(id),

change\_datetime TIMESTAMP DEFAULT NOW(),

old\_value bigint DEFAULT NULL,

new\_value bigint DEFAULT NULL

);\*/

CREATE OR REPLACE FUNCTION trigger\_func()

RETURNS TRIGGER

AS $$

DECLARE

old\_val bigint;

BEGIN

IF (TG\_OP = 'UPDATE') THEN /\* Определяем "старое" значение \*/

old\_val := OLD.id\_vehicle;

ELSIF (TG\_OP = 'INSERT') THEN

old\_val := NULL;

end if;

INSERT INTO log\_examinee

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

VALUES

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

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER commit\_examinee\_change

AFTER UPDATE OR INSERT

ON examinee

FOR EACH ROW

EXECUTE PROCEDURE trigger\_func();

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;

**Вывод:** в ходе данной лабораторной работы работы были изучены основы написания пользовательских функций на процедурном языке PL/pgSQL. Освоена работа с составными типами данных и массивами, созданием триггеров. В процессе работы были написаны функции и заполнены таблицы, созданные в первой лабораторной работе. На основании всего сказанного выше можно сделать вывод о том, что цель достигнута, необходимые знания получены, задание выполнено верно, что доказывается полученными результатами.