Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

# Лабораторная работа№ 3

«Процедуры, функции, триггеры в PostrgreSQL»

Выполнил: Галиновский Роман Андреевич

Группа: K3240 Преподаватель: 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.

**Вариант 2**

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.

Указание. Работа выполняется в консоли SQL Shell (psql).

## ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ:

### **Название БД**

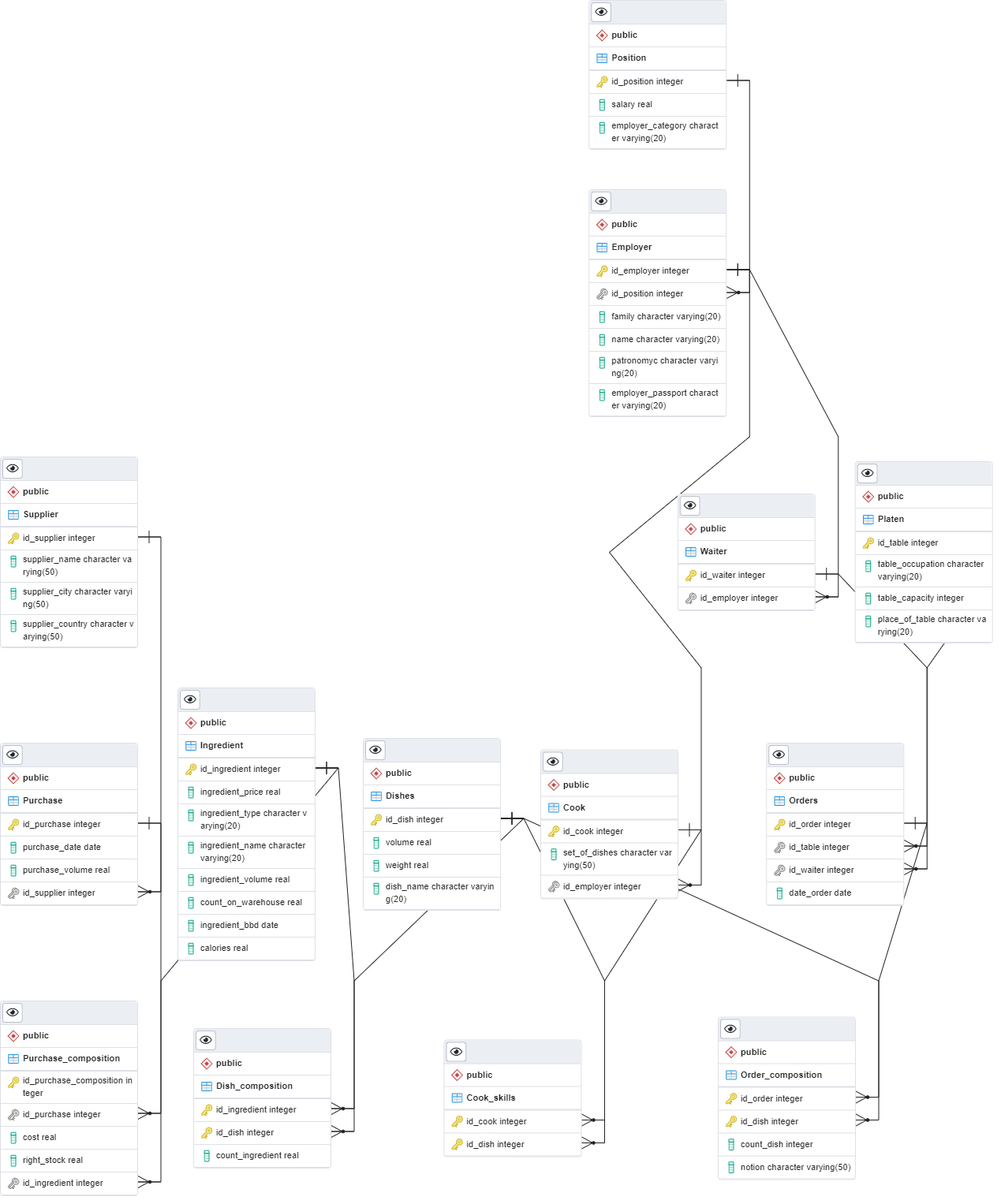
**Вариант 13. «Ресторан»**

Описание предметной области: Сотрудники ресторана – повара и официанты. За каждым официантом закреплены определенные столы. Каждый повар готовит определенный набор блюд. Запас продуктов на складе не должен быть ниже заданного значения. Цена заказа складывается из стоимости ингредиентов и наценки, которая составляет 40% стоимости ингредиентов. БД должна содержать следующий минимальный набор сведений: ФИО сотрудника. Паспортные данные сотрудника. Категория сотрудника. Должность сотрудника. Оклад сотрудника. Наименование ингредиента. Код ингредиента. Дата закупки. Объем закупки. Количество продукта на складе. Необходимый запас продукта. Срок годности. Цена ингредиента. Поставщик. Наименование блюда. Код блюда. Объем ингредиента. Номер стола. Дата заказа. Код заказа. Количество. Название блюда. Ингредиенты, входящие в блюдо. Тип ингредиента.

### Состав реквизитов сущностей:

* + 1. **Ингредиент (**ID ингредиента, цена ингредиента, тип ингредиента, название ингредиента, объём ингредиента, количество на складе, срок годности, калорийность)
    2. **Поставщик** (ID поставщика, имя поставщика, город поставщика, страна поставщика)
    3. **Стол** (ID стола, размещение стола, занятость стола, вместимость стола)
    4. **Должность** (ID должности, оклад, категория сотрудника)
    5. **Закупки** (ID закупки, ID поставщика, дата закупки, объём закупки)
    6. **Состав закупки** (ID состава закупки, ID закупки, ID ингредиента, стоймость, нужный запас)
    7. **Сотрудники** (ID сотрудника, ID должности, Фамилия, имя, отчество, паспорт сотрудника, *категория, зарплата*)
    8. **Официант** (ID Официанта, ID сотрудника)
    9. **Заказы** (ID заказа, ID стола, ID Официанта, Дата заказа)
    10. **Состав Заказа(**ID заказа, ID блюда, Количество блюда, примечание)
    11. **Повара** (ID Повара, ID сотрудника, набор блюд)
    12. **Умение повара**(ID Повара, ID Блюда)
    13. **Блюдо** (ID блюда, объём, вес, название блюда)
    14. **Состав Блюда** (ID ингредиента, ID блюда, количество ингредиентов)

***Схема базы данных:***



**Выполнение:**

1. *Создание хранимых процедур:*

*1.1.* Вывести сведения о заказах заданного официанта на заданную дату:

CREATE OR REPLACE FUNCTION orders\_for\_waiters (order\_date date, waiter\_id integer)

RETURNS TABLE (family VARCHAR, name VARCHAR,

order\_id integer)

AS $$

SELECT public."Employer".family, public."Employer".name,

public."Orders".id\_order

FROM public."Waiter", public."Employer", public."Orders"

WHERE order\_date = public."Orders".date\_order

AND public."Waiter".id\_employer = public."Employer".id\_employer

AND public."Waiter".id\_waiter = waiter\_id;

$$

LANGUAGE SQL;

Изображение выглядит как текст

Автоматически созданное описание

Результат:

При проверке SELECT \* FROM orders\_for\_waiters(‘2022-06-10’, 111132);

Изображение выглядит как текст

Автоматически созданное описание

*1.2* Выполнить расчет стоимости заданного заказа:

*CREATE OR REPLACE FUNCTION price\_of\_current\_order (order\_id integer)*

*RETURNS INTEGER*

*AS $$*

*SELECT (SUM(public."Ingredient".ingredient\_price) \* public."Order\_composition".count\_dish) AS RES*

*FROM public."Ingredient", public."Order\_composition", public."Dish\_composition",public."Dishes"*

*WHERE public."Ingredient".id\_ingredient = public."Dish\_composition".id\_ingredient*

*AND public."Dish\_composition".id\_dish = public."Dishes".id\_dish*

*AND public."Dishes".id\_dish = public."Order\_composition".id\_dish*

*AND order\_id = public."Order\_composition".id\_order*

*GROUP BY public."Order\_composition".count\_dish*

*$$*

*LANGUAGE SQL;*

Изображение выглядит как текст

Автоматически созданное описание

Результат:

При проверке SELECT price\_of\_current\_order(111702);

SELECT price\_of\_current\_order(111165);

Изображение выглядит как текст

Автоматически созданное описание

*1.3.* Повышения оклада заданного сотрудника на 30 % при повышении его категории:

CREATE OR REPLACE FUNCTION salary\_increase (employer\_id integer)

RETURNS TABLE(family VARCHAR, name VARCHAR, salary real, category integer)

AS $$

UPDATE public."Employer"

SET salary = salary \* 1.3,

category = category+1

WHERE id\_employer IN(SELECT id\_employer FROM public."Employer"

WHERE employer\_id = public."Employer".id\_employer);

SELECT public."Employer".family, public."Employer".name, public."Employer".salary, public."Employer".category

FROM public."Employer"

WHERE employer\_id = public."Employer".id\_employer

$$

LANGUAGE SQL;

*p.s в запросе было сказано сделать исключительно процедуру по UPDATE сотрудника, но я решил помимо самого UPDATE категории и ЗП, сделать так, чтоб функция возвращала таблицу, где измененные данные непосредственно будут видны.*

*Изображение выглядит как текст

Автоматически созданное описание*

Результат:

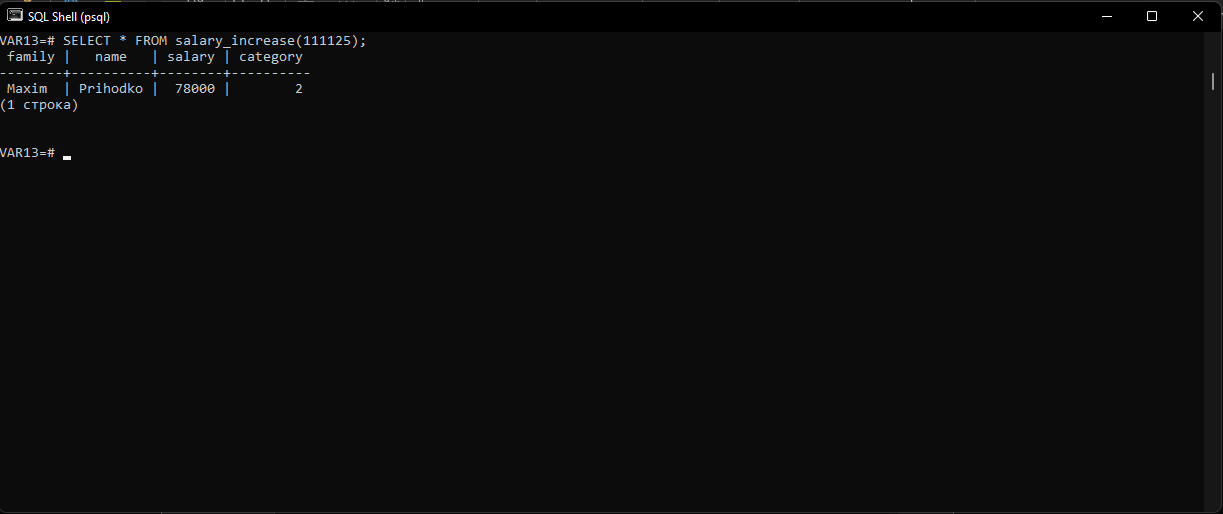
До:

Изображение выглядит как текст

Автоматически созданное описание

После:

При проверке SELECT \* FROM salary\_increase(111125);



Изображение выглядит как текст, внутренний, темный, снимок экрана

Автоматически созданное описание

1. *Создание триггеров:  
   2.1.* Создадим универсальный триггер для логирования действий добавления, обновления и удаления данных. Предварительно создадим таблицу «logs» для этого:  
     
   Изображение выглядит как текст

   Автоматически созданное описание  
   Создадим триггер и сделаем запросы:  
     
   *Изображение выглядит как текст

   Автоматически созданное описание  
   Результаты:  
   Изображение выглядит как текст

   Автоматически созданное описание*

**Выводы:**

В ходе выполнения лабораторной работы я узнал принцип создания хранимых процедур и триггеров.