**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №5 «Процедуры, функции, триггеры в PostgreSQL»

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Бархатова Н.А.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**Оглавление**

[Цель работы 3](#_Toc153141044)

[Практическое задание 3](#_Toc153141045)

[Выполнение **Ошибка! Закладка не определена.**](#_Toc153141046)

[Создайте хранимые процедуры/функции 3](#_Toc153141047)

[1. Вывести сведения о заказах заданного официанта на заданную дату. 3](#_Toc153141048)

[2. Выполнить расчет стоимости заданного заказа. 4](#_Toc153141049)

[3. Повышения оклада заданного сотрудника на 30 % при повышении его категории. 5](#_Toc153141050)

[Триггеры 7](#_Toc153141051)

[Вывод 9](#_Toc153141052)

# Цель работы

Овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

# Практическое задание

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. Создать авторский триггер по варианту индивидуального задания.

## Создайте хранимые процедуры/функции

## Вывести сведения о заказах заданного официанта на заданную дату.

**CREATE** **OR** **REPLACE** **FUNCTION** get\_orders\_by\_waiter\_and\_date(

p\_waiter\_id INTEGER,

p\_date DATE

)

**RETURNS** **TABLE** (

order\_id INTEGER,

order\_date **TIMESTAMP**,

menu\_item\_id INTEGER,

menu\_item\_name VARCHAR,

item\_count INTEGER

)

**AS** $$

**BEGIN**

**RETURN** QUERY

**SELECT**

o.id **AS** order\_id,

o.date\_time **AS** order\_date,

oi.menu\_item\_id,

mi.name **AS** menu\_item\_name,

oi.**count** **AS** item\_count

**FROM**

**schema**.**order** o

**JOIN**

**schema**.order\_composition oi **ON** o.id = oi.order\_id

**JOIN**

**schema**.menu\_item mi **ON** oi.menu\_item\_id = mi.id

**WHERE**

o.employee\_id = p\_waiter\_id

**AND** DATE\_TRUNC ('day', o.date\_time) = DATE\_TRUNC('day', p\_date);

**END**;

$$ **LANGUAGE** plpgsql;

Изображение выглядит как текст, снимок экрана, меню, дизайн

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

## Выполнить расчет стоимости заданного заказа.

**CREATE** **OR** **REPLACE** **FUNCTION** calculate\_order\_cost(p\_order\_id INTEGER)

**RETURNS** INTEGER

**AS** $$

**DECLARE**

total\_cost INTEGER:= **0**;

**BEGIN**

**SELECT** **SUM**(mip.price \* oc.**count**)

**INTO** total\_cost

**FROM**

**schema**.order\_composition oc

**JOIN**

**schema**.menu\_item\_price mip **ON** oc.menu\_item\_id = mip.menu\_item\_id

**WHERE**

oc.order\_id = p\_order\_id;

**RETURN** total\_cost;

**END**;

$$ **LANGUAGE** plpgsql;

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

## Повышения оклада заданного сотрудника на 30 % при повышении его категории.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**CREATE** **OR** **REPLACE** **PROCEDURE** increase\_salary\_if\_category\_higher(employee\_id INTEGER, new\_category INTEGER)

**AS** $$

**DECLARE**

current\_salary INTEGER;

current\_category INTEGER;

**BEGIN**

**SELECT** jt.salary, jt.category **INTO** current\_salary, current\_category

**FROM** **schema**.job\_title jt

**JOIN** **schema**.employee e **ON** e.job\_title\_id = jt.id

**WHERE** e.personnel\_number = employee\_id;

IF new\_category > current\_category **AND** new\_category <= **6** **THEN**

**UPDATE** **schema**.job\_title jt

**SET** salary = current\_salary \* POWER(**1**.**3**, new\_category - current\_category), category = new\_category

**WHERE** jt.id = (

**SELECT** job\_title\_id

**FROM** **schema**.employee

**WHERE** personnel\_number = employee\_id

);

**END** IF;

**END**;

$$ **LANGUAGE** plpgsql;

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

## Триггеры

Триггер на создание записи в таблицу menu\_item\_price для нового блюда с автоматической генераций цены блюда через стоимость ингредиентов.

**CREATE** **OR** **REPLACE** **FUNCTION** calculate\_menu\_item\_price\_update()

**RETURNS** **TRIGGER** **AS** $$

**DECLARE**

total\_ingredient\_cost **INTEGER**;

**BEGIN**

**SELECT** **COALESCE** (**SUM** (pc.ingredient\_price \* mic.ingredient\_value), 0)

**INTO** total\_ingredient\_cost

**FROM** schema.menu\_item\_composition mic

**JOIN** schema.purchase\_composition pc **ON** mic.ingredient\_id = pc.ingredient\_id

**WHERE** mic.menu\_item\_id = **NEW**.menu\_item\_id;

**UPDATE** schema.menu\_item\_price **SET** price = total\_ingredient\_cost \* 8 **WHERE** menu\_item\_id = **NEW**.menu\_item\_id;

**RETURN** **NEW**;

**END**;

$$ **LANGUAGE** plpgsql;

**CREATE** **TRIGGER** calculate\_menu\_item\_price\_trigger\_update

**BEFORE** **UPDATE** **ON** schema.menu\_item\_composition

**FOR** EACH **ROW**

**EXECUTE** **FUNCTION** calculate\_menu\_item\_price\_update();

**CREATE** **OR** **REPLACE** **FUNCTION** calculate\_menu\_item\_price\_insert()

**RETURNS** **TRIGGER** **AS** $$

**DECLARE**

total\_ingredient\_cost **INTEGER**;

**BEGIN**

**SELECT** **COALESCE** (**SUM**(pc.ingredient\_price \* **NEW**.ingredient\_value), 0)

**INTO** total\_ingredient\_cost

**FROM** schema.menu\_item\_composition mic

**JOIN** schema.purchase\_composition pc **ON** mic.ingredient\_id = pc.ingredient\_id

**WHERE** mic.menu\_item\_id = **NEW**.menu\_item\_id;

**INSERT** **INTO** schema.menu\_item\_price (menu\_item\_id, price)

**VALUES** (**NEW**.menu\_item\_id, total\_ingredient\_cost \* 8)

**ON** CONFLICT (menu\_item\_id) DO NOTHING;

**RETURN** **NEW**;

**END**;

$$ **LANGUAGE** plpgsql;

**CREATE** **TRIGGER** calculate\_menu\_item\_price\_trigger\_insert

AFTER **INSERT** **ON** schema.menu\_item\_composition

**FOR** EACH **ROW**

**EXECUTE** **FUNCTION** calculate\_menu\_item\_price\_insert();

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

# Вывод

В ходе выполнения лабораторной работы были успешно освоены практические навыки создания и использования процедур, функций и триггеров в базе данных PostgreSQL. Работа велась в компьютерном классе с использованием СУБД PostgreSQL и SQL Shell (psql). Полученные навыки предоставляют возможность эффективного управления данными в базе, автоматизации рутинных операций и улучшению общей производительности системы. Это важный шаг в обучении и практике работы с базами данных, который может быть полезен как в академическом, так и в профессиональном контексте. Таким образом, цель работы достигнута, а полученные знания и опыт позволят легко адаптироваться к решению разнообразных задач в области баз данных.