ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

Дисциплина:
«Проектирование и реализация баз данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

«процедуры, функции, триггеры в PostgreSQL»

Выполнил:
студент группы K33391
Черкес Артур Викторович

(подпись)

Проверил(а):
Говорова Марина Михайловна

(отметка о выполнении)

(подпись)

Санкт-Петербург 2023
г.

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. 2.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 13. БД «Ресторан»

Описание предметной области: Необходимо создать систему для обслуживания заказов клиентов в ресторане.

Сотрудники ресторана – повара и официанты.

За каждым официантом закреплены определенные столы за смену. Клиенты могут бронировать столы заранее.

Каждый повар может готовить определенный набор блюд.

Официант принимает заказ от стола и передает его на кухню. Шефповар распределяет блюда для приготовления между поварами. В одном заказе может быть несколько одинаковых или разных 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: Табельный номер сотрудника. 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. Калорийность (на 100г продукта). Поставщик. Наименование блюда. Код блюда. Объем ингредиента. Номер стола. Дата заказа. Код заказа. Количество. Название блюда. Ингредиенты, входящие в блюдо. Тип ингредиента.

Задание 1.1 (ЛР 1 БД). Выполните инфологическое моделирование базы данных системы. (Ограничения задать самостоятельно.)

Задание 1.2. Создайте логическую модель БД, используя ИЛМ (задание 1.1). Используйте необходимые средства поддержки целостности данных в СУБД.

Задание 2. Создать запросы:

- Вывести данные официанта, принявшего заказы на максимальную сумму за истекший месяц.
- Рассчитать премию каждого официанта за последние 10 дней (5% от стоимости каждого заказа).
- Подсчитать, сколько ингредиентов содержит каждое блюдо.
- Вывести название блюда, содержащее максимальное число ингредиентов.
- Какой повар может приготовить максимальное число видов блюд?
- Сколько закреплено столов за каждым из официантов?
- Какой из ингредиентов используется в максимальном количестве блюд?

Задание 3. Создать представление:

- для расчета стоимости ингредиентов для заданного блюда;
- количество приготовленных блюд по каждому блюду за определенную дату.

Задание 4. Создать хранимые процедуры:

- Вывести сведения о заказах заданного официанта на заданную дату.
- Выполнить расчет стоимости заданного заказа.
- Повышения оклада заданного сотрудника на при повышении его категории.

Задание 5. Создать необходимые триггеры.

Выполнение:
Создать хранимые процедуры:

Вывести сведения о заказах заданного официанта на заданную дату.

```
Restaurant=# CREATE OR REPLACE FUNCTION GetWaiterOrdersOnDate(    waiter_id bigint,    target_date date)RETURNS TABLE (    order_id bigi
nt,    order_date date,    status character varying(20),    payment_state character varying(20),    table_id bigint)AS $$BEGIN    RETURN
 QUERY    SELECT        o.id AS order_id,        o.date AS order_date,        o.status,        o.payment_state,        o.table_id    FRO
M        restaurant.orders AS o    WHERE        o.staff_personnel_id = waiter_id        AND o.date = target_date;END;$$ LANGUAGE plpgsql
;
CREATE FUNCTION
```

```
Restaurant=# SELECT * FROM GetWaiterOrdersOnDate(4, '2023-10-02');
 order_id | order_date | status  | payment_state | table_id
----------+------------+---------+---------------+----------
       19 | 2023-10-02 | Активен | Оплачен       |        4
(1 row)
```

Выполнить расчет стоимости заданного заказа.

```
Restaurant=# CREATE OR REPLACE FUNCTION calculate_order_cost(order_id bigint) RETURNS bigint AS $$DECLARE    total_cost bigint := 0;BEGIN    SEL
ECT SUM(d.price * op.dishes_amount)    INTO total_cost    FROM restaurant.orders_precessings op    JOIN restaurant.dishes d ON op.dishes_id = d.
id    WHERE op.orders_id = order_id;    RETURN total_cost;END;$$ LANGUAGE plpgsql;
CREATE FUNCTION
Restaurant=# SELECT calculate_order_cost(19);
 calculate_order_cost
----------------------
                  800
(1 row)
```

Повышения оклада заданного сотрудника на 30 % при повышении его категории

```
Restaurant=# CREATE OR REPLACE FUNCTION increase_salary_on_promotion(personnel_id_param bigint) RETURNS void AS $$DECLARE    current_sala
ry bigint;    new_salary bigint;BEGIN    SELECT salary INTO current_salary FROM restaurant.staff WHERE personnel_id = personnel_id_param;
    UPDATE restaurant.staff SET category = 'Новая категория' WHERE personnel_id = personnel_id_param;    new_salary := current_salary * 1
.3;    UPDATE restaurant.staff SET salary = new_salary WHERE personnel_id = personnel_id_param;END;$$ LANGUAGE plpgsql;
CREATE FUNCTION
Restaurant=# SELECT increase_salary_on_promotion(1);
 increase_salary_on_promotion
------------------------------

(1 row)
```

Триггеры

Записываем логи

```
Restaurant=# CREATE OR REPLACE FUNCTION log_order_changes()
Restaurant-# RETURNS TRIGGER AS $$
Restaurant$# BEGIN
Restaurant$#     IF TG_OP = 'INSERT' THEN
Restaurant$#         INSERT INTO restaurant.order_logs (action, order_id, date_changed)
Restaurant$#         VALUES ('INSERT', NEW.id, now());
Restaurant$#     ELSIF TG_OP = 'UPDATE' THEN
Restaurant$#         INSERT INTO restaurant.order_logs (action, order_id, date_changed)
Restaurant$#         VALUES ('UPDATE', OLD.id, now());
Restaurant$#     ELSIF TG_OP = 'DELETE' THEN
Restaurant$#         INSERT INTO restaurant.order_logs (action, order_id, date_changed)
Restaurant$#         VALUES ('DELETE', OLD.id, now());
Restaurant$#     END IF;
Restaurant$#     RETURN NEW;
Restaurant$# END;
Restaurant$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
Restaurant=# CREATE TRIGGER orders_log_trigger
Restaurant-# AFTER INSERT OR UPDATE OR DELETE ON restaurant.orders
Restaurant-# FOR EACH ROW
Restaurant-# EXECUTE FUNCTION log_order_changes();
CREATE TRIGGER
```

```
Restaurant=# INSERT INTO restaurant.orders (date, status, payment_state, table_id, staff_personnel_id)VALUES ('2023-10-10', 'Активен', 'Оплачен',
4, 4);
INSERT 0 1
```

```
Restaurant=# UPDATE restaurant.orders SET status = 'Активен' WHERE id = 20;
UPDATE 1
```

```
Restaurant=# DELETE FROM restaurant.orders WHERE id = 21;
DELETE 1
```

Query | Query History

```
1  SELECT * FROM restaurant.order_logs
2
```

Data Output | Messages | Notifications

	action character varying (50)	order_id bigint	date_changed date
1	INSERT	24	2023-10-04
2	UPDATE	20	2023-10-04
3	DELETE	21	2023-10-04