

Київський національний університет ім. Тараса Шевченка

Кафедра мережевих та інтернет технологій

## **Лабораторна робота № 4**

### **Дисципліна: бази даних та інформаційні системи**

**Тема:** Розширення можливостей PostgreSQL: користувацькі типи, функції та тригери

Виконав: Студент групи МІТ-31

Пугач Назар

**Мета:** Закріпити знання з розширюваності PostgreSQL. Навчитися створювати користувацькі типи даних. Реалізувати власну користувацьку функцію або агрегат. Створити тригери для логування змін у базі даних. Автоматично оновлювати пов'язані таблиці чи заповнювати значення. Оновити діаграму бази даних відповідно до виконаних завдань. Перевірити коректність роботи реалізованих об'єктів через виконання тестових SQL-запитів.

### **Хід роботи**

**Завдання 1:** Створення користувацького типу даних.

```
CREATE TYPE payment_status AS ENUM ('pending', 'completed', 'failed');  
ALTER TABLE payments  
ALTER COLUMN status TYPE payment_status  
USING status::payment_status;
```

**Завдання 2:** Створення користувацької функції або агрегату.

```
CREATE FUNCTION total_payments_analysis(user_id_param INT)  
RETURNS TABLE (  
    total_amount DECIMAL,  
    completed_amount DECIMAL,  
    pending_amount DECIMAL  
) AS $$  
BEGIN  
    RETURN QUERY  
    SELECT  
        COALESCE(SUM(amount), 0) AS total_amount,
```

```

        COALESCE(SUM(CASE WHEN status = 'completed' THEN amount
ELSE 0 END), 0) AS completed_amount,

        COALESCE(SUM(CASE WHEN status = 'pending' THEN amount ELSE
0 END), 0) AS pending_amount

FROM payments

WHERE user_id = user_id_param;

END;

$$ LANGUAGE plpgsql;

```




	total_amount  numeric	completed_amount  numeric	pending_amount  numeric
1	210.75	210.75	0

Рисунок 1 – Результат роботи функції.

**Завдання 3:** Створення тригерів для логування змін та автоматичного оновлення пов'язаних таблиць.

```

CREATE TABLE payment_log (
    log_id SERIAL PRIMARY KEY,
    payment_id INT,
    operation VARCHAR(10),
    changed_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE OR REPLACE FUNCTION log_payment_changes() RETURNS
TRIGGER AS $$

BEGIN

    INSERT INTO payment_log (payment_id, operation)
    VALUES (NEW.id, TG_OP);

    RETURN NEW;

END;

$$ LANGUAGE plpgsql;

```

```
CREATE TRIGGER track_payment_changes
AFTER INSERT OR UPDATE OR DELETE ON payments
FOR EACH ROW
EXECUTE FUNCTION log_payment_changes();
```

#### Завдання 4: Оновлення діаграми бази даних.

Змінився тип даних в таблиці платежів, та додалась нова таблиця для логування.

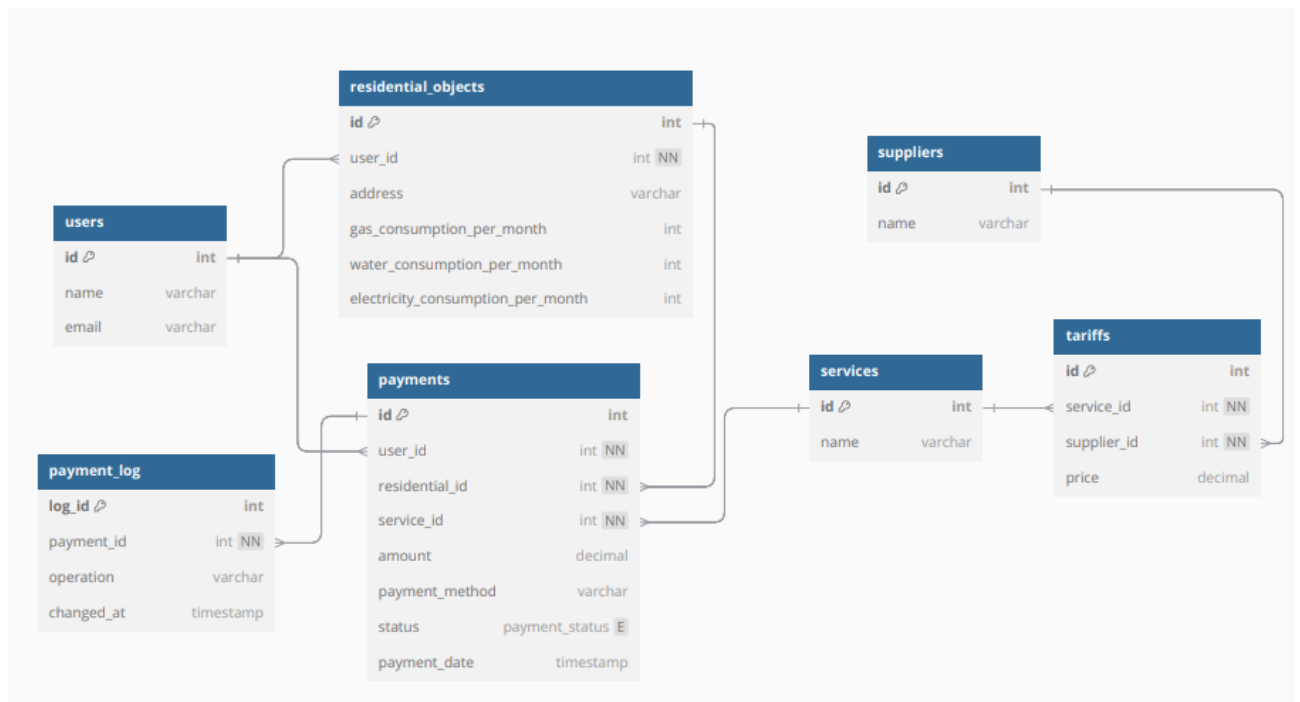


Рисунок 2 – Оновлена діаграма.

**Висновок:** Під час виконання лабораторної роботи я закріпив знання з розширюваності PostgreSQL. Навчився створювати користувацькі типи даних. Реалізував власну користувацьку функцію. Створив тригери для логування змін у базі даних. Оновив діаграму бази даних відповідно до виконаних завдань. Перевірив коректність роботи реалізованих об'єктів через виконання тестових SQL-запитів.