Лабораторна робота №4

Виконав студент групи MIT-31 Тимохін Роман Миколайович

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

Мета роботи: 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.

Користувацький тип даних

створюємо ENUM тип для статусу замовлення

```
-- Користувацький тип статусу замовлення

CREATE TYPE order_status AS ENUM ('Очікується', 'Оплачено',
'Скасовано', 'Виконано');

-- Зміна поля у таблиці orders

ALTER TABLE orders

ALTER COLUMN status DROP DEFAULT,

ALTER COLUMN status TYPE order_status USING status::order_status,

ALTER COLUMN status SET DEFAULT 'Очікується';
```

• Це дозволяє жорстко обмежити список допустимих статусів та уникнути помилок введення.

Користувацька функція або агрегат

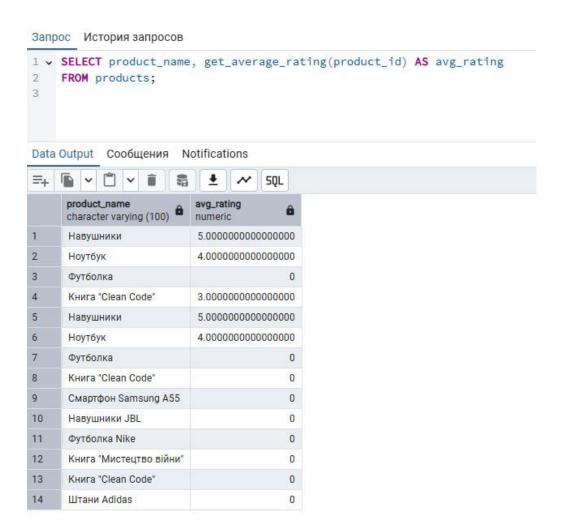
Функція: підрахунок середнього рейтингу товару

```
CREATE OR REPLACE FUNCTION get_average_rating(prod_id INT)
RETURNS NUMERIC AS $$
DECLARE
    avg_rating NUMERIC;
BEGIN
    SELECT AVG(rating) INTO avg_rating
    FROM reviews
    WHERE product_id = prod_id;

RETURN COALESCE(avg_rating, 0);
END;
$$ LANGUAGE plpgsql;
```

Тестовий виклик:

SELECT product_name, get_average_rating(product_id) AS avg_rating FROM products;



Тригери для логування та оновлення

Логування змін у таблиці products

```
-- Створення таблиці логів

CREATE TABLE product_log (
    log_id SERIAL PRIMARY KEY,
    action_type VARCHAR(10),
    product_id INT,
    old_price NUMERIC,
    new_price NUMERIC,
    changed_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Функція тригера:

```
CREATE OR REPLACE FUNCTION log product update()
RETURNS TRIGGER AS $$
BEGIN
    IF TG OP = 'UPDATE' THEN
        INSERT INTO product log(action type, product id, old price,
new price)
        VALUES ('UPDATE', NEW.product id, OLD.price, NEW.price);
    ELSIF TG OP = 'DELETE' THEN
        INSERT INTO product log(action type, product id, old price)
        VALUES ('DELETE', OLD.product id, OLD.price);
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

Тригер:

```
CREATE TRIGGER trg product update
AFTER UPDATE OR DELETE ON products
FOR EACH ROW
EXECUTE FUNCTION log product update();
```

Автоматичне оновлення: сума замовлення

```
-- Додамо стовпець суми
ALTER TABLE orders ADD COLUMN total amount NUMERIC DEFAULT 0;
-- Функція, яка оновлює суму
CREATE OR REPLACE FUNCTION update order total()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE orders
    SET total amount = (
        SELECT SUM(p.price * oi.quantity)
        FROM order items oi
        JOIN products p ON oi.product id = p.product id
        WHERE oi.order id = NEW.order id
```

```
)
WHERE order_id = NEW.order_id;

RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

♦ Тригер:

CREATE TRIGGER trg_update_total

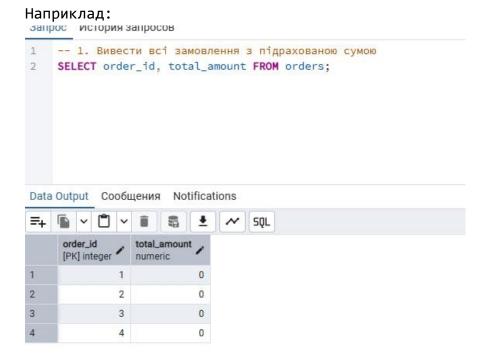
AFTER INSERT OR UPDATE ON order_items

FOR EACH ROW

EXECUTE FUNCTION update_order_total();

Перевірка роботи (Тестові запити)

```
-- 1. Вивести всі замовлення з підрахованою сумою SELECT order_id, total_amount FROM orders;
-- 2. Вивести лог змін по товарах SELECT * FROM product_log ORDER BY changed_at DESC;
-- 3. Змінити ціну товару для тесту UPDATE products SET price = price + 100 WHERE product_id = 1;
-- 4. Перевірити середній рейтинг SELECT product_name, get_average_rating(product_id) FROM products;
```



Оновлена діаграмма буде файлом pdf на github.

Висновок:

У ході виконання лабораторної роботи №4 було поглиблено розуміння принципів побудови розширених структур бази даних на прикладі інформаційної системи інтернет-магазину. Реалізовано користувацький тип даних (ENUM) для зберігання статусів замовлень, що забезпечує цілісність і передбачуваність даних. Створено користувацьку функцію для розрахунку середнього рейтингу товарів, яка підвищує гнучкість у вибірках. Запроваджено механізм логування змін у таблиці товарів за допомогою тригерів, що покращує контроль над змінами. Додано автоматичне оновлення загальної суми замовлення при зміні його вмісту. Оновлена ER-діаграма відображає всі нові сутності та зв'язки. Застосовані підходи підвищують продуктивність, надійність та масштабованість бази даних.