ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №10
«Транзакції в СКБД PostgreSQL»

Виконав: Гуменюк Станіслав

Ст. Групи ПМІ-33с

Оцінка ___

Прийняв:

2024

Транзакція 1: Додавання нового клієнта та оформлення замовлення

- Демонструє використання BEGIN і COMMIT
- Використовує SAVEPOINT двічі
- Показує ROLLBACK TO у разі дублювання email або перевищення суми замовлення

```
-- Транзакція 1: Додавання нового клієнта та оформлення замовлення
BEGIN;

-- Додавання нового клієнта
INSERT INTO person (name, surname, birth_date)
VALUES ('Олена', 'Петренко', '1990-05-15');

SAVEPOINT new_customer;

-- Додавання контактної інформації
INSERT INTO person_contact (person_id, contact_type_id, contact_value)
VALUES
    ((SELECT MAX(id) FROM person), (SELECT id FROM contact_type WHERE name = 'Phone'), '050-123-
    ((SELECT MAX(id) FROM person), (SELECT id FROM contact_type WHERE name = 'Email'), 'olena.pe

-- Перевірка: якщо email вже існує, відкатимося до точки збереження
IF EXISTS (
    SELECT 1 FROM person_contact
    WHERE contact_type_id = (SELECT id FROM contact_type WHERE name = 'Email')
    AND contact_value = 'olena.petrenko@email.com'
    HAVING COUNT(*) > 1
) THEN
    ROLLBACK TO new_customer;
    RAISE NOTICE 'Email вже існує. Відкат до створення клієнта.';
ELSE
    -- Додавання клієнта
    INSERT INTO customer (person_id, card_number, discount)
    VALUES ((SELECT MAX(id) FROM person), 2001, 0.05);

    SAVEPOINT customer_added;
```

```
    -- Створення замовлення
    INSERT INTO customer_order (operation_time, supermarket_id, customer_id)
    VALUES (CURRENT_DATE, 1, (SELECT MAX(person_id) FROM customer));

    -- Додавання деталей замовлення
    INSERT INTO order_details (customer_order_id, product_id, price, price_with_discount, produc
    VALUES
        ((SELECT MAX(id) FROM customer_order), 1, 0.50, 0.475, 5),
        ((SELECT MAX(id) FROM customer_order), 4, 2.50, 2.375, 2);

    -- Перевірка: якщо загальна сума замовлення перевищує 1000, відкатимося до точки збереження
    IF (SELECT SUM(price_with_discount * product_amount) FROM order_details WHERE customer_order
        ROLLBACK TO customer_added;
        RAISE NOTICE 'Сума замовлення перевищує ліміт. Відкат до створення клієнта.';
    ELSE
        COMMIT;
        RAISE NOTICE 'Транзакція успішно завершена.';
    END IF;
END IF;
```

Транзакція 2: Оновлення цін на продукти та додавання нової категорії

- Демонструє використання BEGIN і COMMIT
- Використовує SAVEPOINT після оновлення цін
- Показує ROLLBACK TO, якщо ціна нового продукту занадто низька порівняно з середньою ціною в категорії

```sql
-- Транзакція 2: Оновлення цін на продукти та додавання нової категорії
BEGIN;

-- Оновлення цін на всі продукти (підвищення на 10%)
UPDATE product SET price = price * 1.1;

SAVEPOINT prices_updated;

-- Додавання нової категорії продуктів
INSERT INTO product_category (name) VALUES ('Органічні продукти');

-- Додавання нового продукту до нової категорії
INSERT INTO product_title (title, product_category_id)
VALUES ('Органічні яблука', (SELECT id FROM product_category WHERE name = 'Органічні продукти'))

INSERT INTO product (product_title_id, manufacturer_id, price, comment)
VALUES (
    (SELECT id FROM product_title WHERE title = 'Органічні яблука'),
    (SELECT id FROM manufacturer WHERE name = 'FreshFarms'),
    1.50,
    'Свіжі органічні яблука'
);

-- Перевірка: якщо ціна нового продукту менше середньої ціни в категорії, відкатимося до оновлен
IF (
    SELECT AVG(price) FROM product
    JOIN product_title ON product.product_title_id = product_title.id
    WHERE product_title.product_category_id = (SELECT id FROM product_category WHERE name = 'Орг
) > 1.50 THEN
    ROLLBACK TO prices_updated;
    RAISE NOTICE 'Ціна нового продукту занадто низька. Відкат до оновлення цін.';
ELSE
    COMMIT;
    RAISE NOTICE 'Транзакція успішно завершена.';
END IF;
```

Демонстрація роботи паралельних транзакцій на прикладі рівня ізоляції READ COMMITED

```
-- Сесія 1
BEGIN;
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

-- Сесія 2
BEGIN;
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

-- Сесія 1: Оновлення ціни продукту
UPDATE product SET price = 2.00 WHERE id = 1;

-- Сесія 2: Спроба прочитати ціну продукту
SELECT price FROM product WHERE id = 1;
-- Результат: 1.50 (стара ціна, тому що транзакція 1 ще не зафіксована)

-- Сесія 1: Фіксація змін
COMMIT;

-- Сесія 2: Повторне читання ціни продукту
SELECT price FROM product WHERE id = 1;
-- Результат: 2.00 (нова ціна, тому що транзакція 1 вже зафіксована)

-- Сесія 2: Спроба оновити той самий продукт
UPDATE product SET price = 2.50 WHERE id = 1;
-- Ця операція буде успішною, оскільки немає конфлікту

-- Сесія 2: Фіксація змін
COMMIT;

-- Будь-яка сесія: Перевірка фінальної ціни
SELECT price FROM product WHERE id = 1;
-- Результат: 2.50
```