SQL на практике: создаем таблицу товаров и взаимодействуем с ней. Часть 2

Грачев Д.Г.

Триггеры

Триггер — специальная хранимая процедура, которая автоматически выполняется при определенных событиях в таблице.

- `INSERT` — при добавлении записи
- `UPDATE` — при изменении записи
- `DELETE` — при удалении записи

Плюсы:

1. **Автоматизация** — не нужно помнить об обновлении связанных данных
2. **Целостность данных** — гарантия корректности счетчиков и связей
3. **Аудит** — автоматическое логирование изменений
4. **Бизнес-правила** — принудительное выполнение правил на уровне БД

Минусы:

1. **Скрытая логика** — сложнее отлаживать
2. **Производительность** — дополнительные операции
3. **Каскадные эффекты** — могут вызывать другие триггеры

Схема

```sql
-- Категории продуктов
CREATE TABLE categories (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    product_count INT DEFAULT 0  -- количество продуктов в категории
);
```

```sql
-- Продукты
CREATE TABLE products (
    id SERIAL PRIMARY KEY,
    name VARCHAR(200) NOT NULL,
    category_id INT REFERENCES categories(id) ON DELETE SET NULL,
    price DECIMAL(10,2) NOT NULL,
    stock_quantity INT DEFAULT 0,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Без триггера

```sql
-- Ручное обновление (легко забыть!)
INSERT INTO products (name, category_id, price)
VALUES ('iPhone 15', 1, 999.99);

UPDATE categories
SET product_count = product_count + 1
  WHERE id = 1;
```

Функции для триггера

```
CREATE OR REPLACE FUNCTION update_category_product_count()
RETURNS TRIGGER AS $$
BEGIN
    IF TG_OP = 'INSERT' THEN
        UPDATE categories
        SET product_count = product_count + 1
        WHERE id = NEW.category_id;
    END IF;
    RETURN NULL; -- Для AFTER триггера
END;
  $$ LANGUAGE plpgsql;
```

Создание триггера

```
CREATE TRIGGER trg_update_product_count

AFTER INSERT ON products

FOR EACH ROW

  EXECUTE FUNCTION update_category_product_count();
```

Демонстрация работы

```sql
-- 1. Добавляем категорию
INSERT INTO categories (name) VALUES ('Электроника');
-- 2. Добавляем продукты (триггер сработает автоматически)
INSERT INTO products (name, category_id, price)
VALUES
    ('Ноутбук', 1, 1500),
    ('Смартфон', 1, 800);
```

Практическое задание

Добавить поддержку операций DELETE и UPDATE для функции **update_category_product_count**. Вызвать триггер и проверить что отрабатывает корректно.

Итого

Подходит:

1. Четко документировать триггеры
2. Использовать для поддержания целостности
3. Тестировать на разных сценариях

Не подходит:

1. Создавать сложные цепочки триггеров
2. Использовать для бизнес-логики (лучше в приложении)
3. Забывать про производительность

Ресурсы

1. https://postgrespro.ru/docs/postgresql/current/plpgsql-trigger - офф документация на русском
2. https://habr.com/ru/companies/otus/articles/857396/ - хорошая статья на хабре