Laboratorium 6

Bazy Danych 2

mgr. inż. Aleksander Wojtowicz

Procedury wyzwalane, *znane również jako triggery*, to specjalne rodzaje procedur składowanych, które są automatycznie wykonywane *(wyzwalane)* przez serwer bazy danych w odpowiedzi na określone zdarzenia, takie jak wstawienie, aktualizacja lub usunięcie rekordów w tabeli.

Służą one do zapewnienia integralności danych, automatyzacji zadań, egzekwowania reguł biznesowych oraz rejestrowania zmian w bazie danych.

Zastosowanie procedur wyzwalanych:

- **Automatyzacja zadań:** Procedury wyzwalane mogą automatycznie wykonywać operacje.

- **Zapewnienie integralności danych:** Mogą egzekwować reguły biznesowe i ograniczenia, które nie są bezpośrednio obsługiwane przez standardowe mechanizmy bazy danych.
  Np. aktualizacja danych w powiązanych tabelach po wstawieniu nowego rekordu.

- **Rejestrowanie zmian:** Triggery mogą być używane do śledzenia zmian w danych, tworząc logi operacji.

- **Kontrola dostępu:** Mogą ograniczać dostęp do danych na poziomie rekordów.

Składnia dla `CREATE TRIGGER`

```
CREATE [OR REPLACE] TRIGGER nazwa_wyzwalacza
{BEFORE | AFTER | INSTEAD OF}
{INSERT | UPDATE [OF kolumna] | DELETE}
ON nazwa_tabeli
[REFERENCING OLD AS stara | NEW AS nowa]
[FOR EACH ROW]
[WHEN (warunek)]
DECLARE
  -- Deklaracje zmiennych
BEGIN
  -- Główny kod wyzwalacza
END;
```

- `Nazwa_wyzwalacza`: To po prostu identyfikator wyzwalacza.
- `BEFORE`, `AFTER`, `INSTEAD OF`: Określają, kiedy wyzwalacz ma zostać uruchomiony *(przed, po lub zamiast operacji).*
- `INSERT`, `UPDATE`, `DELETE`: To zdarzenia DML, które aktywują wyzwalacz *(dodawanie, aktualizacja, usuwanie danych).*
- `ON nazwa_tabeli`: Wskazuje tabelę, na której wyzwalacz jest zdefiniowany.
- `REFERENCING`: Pozwala na odniesienie do starych i nowych wartości rekordów.
- `FOR EACH ROW`: Oznacza, że wyzwalacz uruchamia się dla każdego wiersza, a nie tylko raz dla całej instrukcji.
- `WHEN (warunek)`: Pozwala na określenie warunku, który musi być spełniony, aby wyzwalacz został uruchomiony.

Różnice między `BEFORE`, `AFTER`, `INSTEAD OF`:

- `BEFORE`: Wyzwalacz uruchamiany przed wykonaniem operacji DML, co pozwala na weryfikację lub zmianę danych przed ich zapisaniem.
  Np.: Możemy użyć wyzwalacza BEFORE, aby upewnić się, że wprowadzane dane spełniają określone kryteria (np. nieujemne liczby).

- `AFTER`: Wyzwalacz uruchamiany po wykonaniu operacji DML, co pozwala na reakcję na zmiany już zapisane w bazie danych.
  Przykład: Po dodaniu nowego zamówienia, wyzwalacz AFTER może automatycznie zaktualizować stan magazynowy.

- `INSTEAD OF`: Ten rodzaj wyzwalacza jest używany głównie dla widoków. Zastępuje standardową operację DML, umożliwiając niestandardowe działanie.
  Przykład: Możemy użyć wyzwalacza INSTEAD OF, aby obsłużyć specyficzne przypadki aktualizacji widoku.

Przykłady wyzwalaczy / triggerów:

Wyzwalacz logujący zmiany w tabeli pracownicy:

Do tego będzie potrzebna nam tabela **`LogZmian`**:

```
CREATE TABLE LogZmian (
  id_pracownika NUMBER,
  typ_operacji VARCHAR2(10),
  data_zmiany DATE
);
```

```
CREATE OR REPLACE TRIGGER LogowanieZmian
AFTER INSERT OR UPDATE OR DELETE ON pracownicy
FOR EACH ROW
DECLARE
  typ_operacji VARCHAR2(10);
BEGIN
  IF INSERTING THEN
    typ_operacji := 'INSERT';
  ELSIF UPDATING THEN
    typ_operacji := 'UPDATE';
  ELSIF DELETING THEN
    typ_operacji := 'DELETE';
  END IF;

  INSERT INTO LogZmian (id_pracownika, typ_operacji,
data_zmiany)
  VALUES (:NEW.id_pracownika, typ_operacji, SYSDATE);
END;
```

- `Nazwa_wyzwalacza:` LogowanieZmian
- `AFTER:` uruchamiany po operacji DML
- `INSERT, UPDATE, DELETE:` Wszystkie trzy
  (rejestruje zmiany po wstawieniu, aktualizacji lub usunięciu rekordu)
- `ON pracownicy:` Wskazuje tabelę, na której wyzwalacz jest zdefiniowany
- `FOR EACH ROW:` Oznacza, że wyzwalacz uruchamia się dla każdego wiersza, a nie tylko raz dla całej instrukcji
- `DECLARE:` Rozpoczyna blok deklaracji zmiennych
- `typ_operacji:` Zmienna przechowująca informację o rodzaju operacji *(INSERT, UPDATE lub DELETE)*
- `:NEW.id_pracownika:` Odniesienie do nowej wartości kolumny id_pracownika w wierszu, która spowodowała wyzwalacz
- `SYSDATE:` Aktualna data i godzina

Wyzwalacz sprawdzający warunki biznesowe przed wstawieniem danych do tabeli:

```
CREATE OR REPLACE TRIGGER
SprawdzanieWarunkowBiznesowych
BEFORE INSERT ON pracownicy
FOR EACH ROW
BEGIN
  IF :NEW.pensja > 100000 THEN
    RAISE_APPLICATION_ERROR(-20002, 'Pensja przekracza
maksymalny limit dla nowego pracownika.');
  END IF;
END;
```

- `Nazwa_wyzwalacza:` SprawdzanieWarunkowBiznesowych
- `BEFORE:` Wyzwalacz uruchamiany jest przed wstawieniem nowego rekordu do tabeli.
- `ON pracownicy:` Wskazuje tabelę, na której wyzwalacz jest zdefiniowany.
- `FOR EACH ROW:` Oznacza, że wyzwalacz uruchamia się dla każdego wiersza, a nie tylko raz dla całej instrukcji.
- `IF :NEW.pensja > 100000 THEN:` Sprawdza, czy nowa pensja przekracza określony limit *(100 000)*.

Wyzwalacz synchronizujący dane między tabelami `pracownicy` i `zamowienia`:

Potrzebna będzie nam do tego nowa tabela:

```
CREATE TABLE zamowienia (
    id_zamowienia NUMBER PRIMARY KEY,
    id_pracownika NUMBER,
    kwota_zamowienia NUMBER
);

-- Dodawanie danych do tabeli zamówienia
INSERT INTO zamowienia (id_zamowienia, id_pracownika,
kwota_zamowienia) VALUES (1, 1, 1000);
INSERT INTO zamowienia (id_zamowienia, id_pracownika,
kwota_zamowienia) VALUES (2, 2, 1500);
INSERT INTO zamowienia (id_zamowienia, id_pracownika,
kwota_zamowienia) VALUES (3, 3, 800);
```

Wyzwalacz synchronizujący dane między tabelami pracownicy i zamowienia:

```
CREATE OR REPLACE TRIGGER SynchronizacjaDanych
AFTER UPDATE OF pensja ON pracownicy
FOR EACH ROW
BEGIN
  UPDATE zamowienia
  SET kwota_zamowienia = kwota_zamowienia * :NEW.pensja
/ :OLD.pensja
  WHERE id_pracownika = :NEW.id_pracownika;
END;
```

Wyzwalacz **`SynchronizacjaDanych`** zostanie uruchomiony po każdej aktualizacji kolumny **„pensja”** w tabeli **„pracownicy”**.

Dla każdego wiersza, wyzwalacz oblicza nową wartość **`kwota_zamowienia`** w tabeli **„zamowienia”** na podstawie stosunku nowej i starej pensji pracownika.

Typowe problemy związane z kodem w PL/SQL:

- **Złożoność kodu:** Używaj instrukcji warunkowych i pętli w sposób, który nie komplikuje zrozumienia przepływu programu.

- **Dekompozycja:** Podziel długie procedury na mniejsze funkcje i procedury, które są łatwiejsze do zrozumienia i testowania.

- **Optymalizacja zapytań SQL:** Analizuj zapytania za pomocą narzędzi takich jak `EXPLAIN PLAN`, aby znaleźć i usunąć wąskie gardła wydajności.

- **Indeksy:** Stosuj indeksy w bazie danych tam, gdzie to możliwe, aby przyspieszyć operacje wyszukiwania i sortowania.

- **Zarządzanie błędami:** Implementuj obsługę wyjątków, aby prawidłowo reagować na błędy i unikać ich ukrywania.

Najlepsze praktyki:

- **Standardy kodowania:** Stosuj spójne standardy nazewnictwa i formatowania, aby ułatwić współpracę i utrzymanie kodu.

- **Zarządzanie transakcjami:** Używaj transakcji do zapewnienia integralności danych, stosując odpowiednio polecenia COMMIT i ROLLBACK.

- **Testowanie:** Przeprowadzaj regularne testy jednostkowe i integracyjne, aby wykrywać i naprawiać błędy wcześnie.

- **Dokumentacja:** Dokładnie dokumentuj kod, w tym komentarze i zewnętrzne dokumenty, aby ułatwić zrozumienie i utrzymanie systemu.

Zadania:

1. Napisz wyzwalacz, który przed każdą aktualizacją pensji pracownika sprawdzi, czy nowa pensja nie przekracza 10-krotności średniej pensji wszystkich pracowników. Jeśli tak, wyzwalacz powinien zablokować aktualizację i zgłosić błąd.

2. Stwórz drugą tabelę projekty, gdzie id_pracownika jest kluczem obcym. Utwórz wyzwalacz, który po zmianie nazwiska pracownika w tabeli pracownicy zaktualizuje również jego nazwisko w tabeli projekty.

3. Stwórz wyzwalacz, który po każdej zmianie pensji pracownika obliczy różnicę między nową a starą pensją i zapisze ją wraz z id_pracownika do tabeli historia_pensji.

4. Utwórz wyzwalacz, który nie pozwoli dodać nowego pracownika do tabeli pracownicy, jeśli liczba pracowników przekroczyłaby 10.

5. Dodaj do tabeli pracownicy zawiera kolumnę datę urodzenia data_urodzenia. Napisz wyzwalacz, który przed dodaniem nowego pracownika sprawdzi, czy nie przekracza on 65 lat.