Oracle PL/SQL

Paweł Rajba

pawel@ii.uni.wroc.pl
http://www.kursy24.eu/

Zawartość modułu 5

- Wprowadzenie
- Tworzenie i wykonywanie procedur i funkcji
- Instrukcja RETURN
- Parametry procedur i funkcji oraz ich przesyłanie
- Metadane programu
- Pakiety PL/SQL, pakiety wielokrotnego użycia
- Zbiory wyników
- Przeciążanie deklaracji

 Copyright © Paweł Rajba

Wprowadzenie

- Mamy do dyspozycji
  - Procedury
  - Funkcje
  - Pakiety
- Cechy
  - Unikalne nazwy
  - Składowane w bazie danych
  - Dostępne są metadane (ze słownika danych)

 Copyright © Paweł Rajba

Tworzenie procedur

- Procedurę tworzymy poprzez instrukcję:
  - CREATE [OR REPLACE] PROCEDURE nazwa
    [(param1 {IN|OUT|IN OUT} typ DEFAULT expr, ...
     (paramN {IN|OUT|IN OUT} typ DEFAULT expr]
    {IS|AS}
    [deklaracje zmiennych]
    BEGIN
        ...
    [EXCEPTION
        ...]
    END [nazwa procedury]

 Copyright © Paweł Rajba

Tworzenie funkcji

- Funkcję tworzymy poprzez instrukcję:
  - CREATE [OR REPLACE] FUNCTION nazwa
    [(param1 {IN|OUT|IN OUT} typ DEFAULT expr, ...
     (paramN {IN|OUT|IN OUT} typ DEFAULT expr]
    RETURN typ_danych
    {IS|AS}
    [deklaracje zmiennych]
    BEGIN
        ...
    [EXCEPTION ...]
    END [nazwa funkcji]

 Copyright © Paweł Rajba

Instrukcja RETURN

- Służy zwracania wartości
- Zatrzymuje wykonywania podprogramu
- Stosujemy
  - RETURN; – procedury
  - RETURN wyrażenie; – funkcje

 Copyright © Paweł Rajba

Wykonywanie procedur lub funkcji

- Wykonywanie jest na jeden z trzech sposobów:
  - Jako instrukcję bloku PL/SQL
    - Aplikacja apex,inne kawałki kodu
  - Używając EXECUTE
    - SQL*Plus
  - Używając CALL
    - SQL*Plus
- Przykład: procedury-funkcje-start.txt

 Copyright © Paweł Rajba

Parametry procedur i funkcji

- Przekazywanie parametrów
  - Pozycyjne, Imienne, Mieszane
  - Typy %TYPE, %ROWTYPE
- Deklarowanie trybów
  - IN (domyślnie), OUT, IN OUT
- Parametry domyślne
- Przykład: procedury-funkcje-parametry.txt

 Copyright © Paweł Rajba

Kilka uwag

- Procedury i funkcje są przechowywane w postaci skompilowanej
  - Kod źródłowy jest zapisywany w słownikach
- Klauzula OR REPLACE powoduje nadpisanie istniejącej wersji
- Przed i po operacji tworzenia procedury lub funkcji jest wykonywany COMMIT
- Funkcje nie powinny mieć parametrów typu OUT i IN OUT (do tego są procedury)

 Copyright © Paweł Rajba

Metadane podprogramu

- Mamy cztery perspektywy
  - USER_OBJECTS, ALL_OBJECTS
    - Info o wersji skompilowanej
  - USER_SOURCE, ALL_SOURCE
    - Kody źródłowe
- Odpalamy zapytania
  - select object_name, object_type, status from user_objects where object_name='SP_ZNAKI';
  - select text from user_source where name='SP_ZNAKI';

 Copyright © Paweł Rajba

Pakiety PL/SQL

- Pakiet składa się z dwóch części
  - specyfikacji
    - określa publiczne procedury i funkcje
    - zawiera tylko nagłówki
  - treści
    - zawiera treść metod publicznych
    - może zawierać także procedury i funkcje prywatne
- Pakiety wprowadzają modularność

 Copyright © Paweł Rajba

Deklaracja pakietu

- Część specyfikacji
  - CREATE [OR REPLACE] PACKAGE nazwa_pakietu {IS|AS} [PRAGMA SERIALLY_REUSABLE;]
    [type_definition]
    [constant_declaration]
    [variable_declaration]
    [exception_declaration]
    [cursor_declaration]
    [procedure_header]
    [function_header]
    END [nazwa_pakietu]

 Copyright © Paweł Rajba

Deklaracja pakietu

- Część treści
  - CREATE [OR REPLACE] PACKAGE BODY nazwa
    {IS|AS} [PRAGMA SERIALLY_REUSABLE;]
    [private_declarations]
    [cursor_body]
    [public_procedures_implementation]
    [public_functions_implementation]
      [BEGIN
          ... // instrukcje]
    END [nazwa_pakietu]

 Copyright © Paweł Rajba

Pakiety PL/SQL

- Kilka uwag
  - Nazwa specyfikacji i treści pakietu muszą mieć taką samą nazwę
  - Sekcja zaczynająca się do BEGIN to sekcja inicjalizacyjna (pakietu nie zaczynamy od BEGIN)
  - Deklaracje publiczne mogą być w dowolnej kolejności
    - odwołania mogą być tylko do już zadeklarowanych składn.
  - Najpierw kompilujemy specyfikację, potem treść
  - Specyfikacja może istnieć bez treści (wtedy nie może być deklaracji procedur i funkcji)

 Copyright © Paweł Rajba

Pakiety PL/SQL

- Odwołania do obiektów
  - pakiet.składowa
- Obiekty prywatne dostępne są tylko w ramach definiowanego pakietu
- Tworzenie instancji i inicjacja pakietu
  - Przy pierwszym odwołaniu do składowej pakietu, jest on ładowany do pamięci i pozostaje tam na czas sesji
  - Wtedy też wykonywana jest sekcja inicjalizacyjna
- Przykład: pakiety.txt

 Copyright © Paweł Rajba

Zwracanie zbiorów wyników

- Może mieć miejsce
  - przy zwracaniu wyniku przez funkcję
  - przy przekazaniu parametru typu OUT procedury
- Może być na dwa sposoby
  - Za pomocą kusorów
  - Za pomocą funkcji tabelowych
    - ten punkt umówimy później przy okazji kolekcji

 Copyright © Paweł Rajba

Zwracanie zbiorów wyników

- Wykorzystanie kursorów może być na 2 sposoby
  - Poprzez użycie słabego kursora SYS_REFCURSOR w typie zwrotnym funkcji lub w typie danych parametru OUT procedury
  - Definiujemy typ REF CURSOR w specyfikacji pakietu i następnie wykorzystujemy j.w.
    - w specyfikacji może być tylko deklaracja typu (nie może być deklaracji zmiennych)
- Przykład: zbiory-wynikow.txt

 Copyright © Paweł Rajba

Funkcje w instrukcjach SQL

- Funkcje możemy wykorzystywać w:
  - instrukcji SELECT (list kolumn, klazule WHERE, HAVING, GROUP BY, ORDER BY)
  - instrukcji INSERT (klauzula VALUES)
  - instrukcji UPDATE (klauzula SET)
- Odwołanie
  - [[schemat.]pakiet.]funkcja(p1,...,pN)
- Przykład: funkcje-instrukcje.txt

 Copyright © Paweł Rajba

Przesyłanie parametrów

- Odbywa się na sposoby
  - przez wartość
  - przez odwołanie (wskaźnik)
- Zachowanie domyślne
  - IN – przez odwołanie
  - OUT, IN OUT – przez wartość
    - W przypadku wyjątku kopia (być może zmieniona) jest odrzucana i pierwotna wartość zachowana
    - Można to domyślne zachowanie zmienić używając wskazówki dla kompilatora NOCOPY – wtedy przekazywanie jest przez odwołanie (szybsze rozwiązanie)

 Copyright © Paweł Rajba

Przesyłanie parametrów

- Wskazówka NOCOPY
  - Deklaracja parametru
    - param {OUT | IN OUT} NOCOPY typ_danych
  - Jest to tylko wskazówka, może zostać przez kompilator zignorowana
  - Parametr nie może elementem tabeli indeksowej (ale może być tabelą indeksową)
  - Używanie NOCOPY zwiększa szybkość przy przekazywaniu dużych obiektów (np. kolekcji)
- Przykład: parametry-przesylanie.txt

 Copyright © Paweł Rajba

Przeciążanie deklaracji

- Co to jest przeciążanie, każdy wie
- Przeciążać możemy procedury i funkcje w ramach pakietu
- Warunki przeciążania
  - różne ilości parametrów
  - różne typy parametrów (int i number w tym kontekście nie są różne)
- Przykład: procedury-funkcje-przeciazanie.txt

 Copyright © Paweł Rajba

Pakiety wielokrotnego użycia

- Normalnie, uruchomienie pakietu jest przechowywane w pamięci sesji
- Wraz ze wzrostem liczby użytkowników, wzrasta zużycie pamięci sesji
- Oznaczenie pakietu jako serially_reusable
  - Powoduje wykonanie pakietu tylko na czas wywołania bazodanowego
  - Stan wykonywalny pakietu dostaje niewielkę pulę pamięci, która po ukończeniu wywołania bazodanowego zostanie zwolniona

 Copyright © Paweł Rajba

Pakiety wielokrotnego użycia

- Składnia
  - PRAGMA SERIALLY_REUSABLE
- Kilka uwag
  - Deklaracja pragmy musi być zarówno w specyfikacji, jak i w treści
  - Po każdym wykonaniu, zmienne są resetowane, a kursory zamykane
  - Pakiet wielokrotnego użycia nie może zawierać procedur (funkcji), z któych korzystają wyzwalacze
- Przykład: pakiety-wielokrotne.txt

 Copyright © Paweł Rajba