Pachetele în PL/SQL

Pachetele reprezintă una dintre cele mai puternice structuri puse la dispoziție de PL/SQL pentru organizarea, modularizarea și securizarea codului. Un pachet grupează logic subprograme, tipuri de date, constante și variabile într-o singură unitate, facilitând dezvoltarea de aplicații complexe și bine structurate.

Prin utilizarea pachetelor, logica unui sistem devine mai ușor de întreținut, iar comunicarea dintre componente se realizează într-un mod coerent și eficient. Pachetele permit ascunderea implementării interne și expunerea doar a interfeței publice, ceea ce protejează integritatea codului și permite modificări fără a afecta modulele dependente.

Ce este un pachet în PL/SQL

Un pachet este un obiect PL/SQL compus din două părți:

- specificația pachetului (PACKAGE SPEC) – definește interfața publică și ce elemente sunt vizibile în afara pachetului.
- corpul pachetului (PACKAGE BODY) – conține implementarea logică a procedurilor și funcțiilor declarate în specificație, precum și elemente private.

Pachetele sunt stocate în baza de date ca obiecte permanente, iar Oracle le gestionează automat compilarea, încărcarea în memorie și menținerea stării pentru variabilele globale conținute în acestea.

Când se utilizează pachetele

Pachetele sunt recomandate în diverse situații:

- organizarea codului complex: Atunci când aplicația conține multe proceduri, funcții, tipuri de date personalizate sau constante, pachetele permit gruparea lor într-un modul cu scop clar definit.
- ascunderea implementării interne: Specificația pachetului expune doar elementele publice, în timp ce implementarea poate fi modificată oricând fără a afecta aplicațiile care utilizează pachetul.
- reutilizarea codului: Odată creat, un pachet este accesibil în întreaga bază de date și poate fi folosit de orice utilizator care are privilegii corespunzătoare.
- creșterea performanței: când este apelat un subprogram dintr-un pachet, Oracle încarcă întregul pachet în memorie, reducând costurile de acces la cod.

Structura unui pachet

Un pachet are două componente:

1. Specificația pachetului

Definirea interfeței publice: ce este vizibil în afară. Conține:

- declarații de tipuri de date (RECORD, TABLE, VARRAY)
- constante
- variabile globale
- prototipuri de proceduri și funcții
- excepții declarate

Nu conține cod executabil, cu excepția valorilor atribuite variabilelor în momentul declarării.

```
CREATE OR REPLACE PACKAGE p_angajati IS
  -- constante
  c_bonus CONSTANT NUMBER := 500;

  -- tipuri de date
  TYPE rec_angajat IS RECORD (
```

```
    id     NUMBER,
    nume   VARCHAR2(100),
    salariu NUMBER
  );

  -- subprograme publice
  PROCEDURE mareste_salariu(p_id NUMBER, p_procent NUMBER);
  FUNCTION salariu_total(p_id NUMBER) RETURN NUMBER;
END p_angajati;
```

2. Corpul pachetului

Corpul este definirea implementării unui pachet. Conține:

- codul subprogramelor
- variabile sau subprograme private
- blocuri de inițializare ale pachetului

Corpul pachetului este obligatoriu doar dacă specificația definește cel puțin o procedură sau funcție.

```
CREATE OR REPLACE PACKAGE BODY p_angajati IS

  -- variabilă privată
  v_modificari NUMBER := 0;

  -- procedură publică
  PROCEDURE mareste_salariu(p_id NUMBER, p_procent NUMBER) IS
  BEGIN
    UPDATE angajati
    SET salariu = salariu * (1 + p_procent / 100)
    WHERE id = p_id;

    v_modificari := v_modificari + 1;
  END;

  -- funcție publică
  FUNCTION salariu_total(p_id NUMBER) RETURN NUMBER IS
    v_sal NUMBER;
```

```
  BEGIN
    SELECT salariu INTO v_sal
    FROM angajati
    WHERE id = p_id;

    RETURN v_sal + c_bonus;
  END;

  -- bloc de inițializare al pachetului
  BEGIN
    DBMS_OUTPUT.PUT_LINE('Pachetul p_angajati a fost încărcat.');
  END;

END p_angajati;
```

Blocul de inițializare este executat o singură dată la prima apelare a oricărui subprogram al pachetului.

Pachete predefinite în PL/SQL

Oracle furnizează numeroase pachete standard care oferă funcții utilitare, administrare, control al sesiunii și raportare.

Cele mai importante:

1. DBMS_OUTPUT: Permite afișarea de mesaje în bufferul sesiunii.

```
DBMS_OUTPUT.PUT_LINE('Mesaj');
```

2. DBMS_RANDOM: Generează numere sau șiruri aleatoare.

3. DBMS_UTILITY: Oferă funcții diverse (convertirea numelor de obiecte, generare hash, execuție LRU etc.)

4. DBMS_SQL: Permite execuția dinamică avansată a instrucțiunilor SQL.

5. DBMS_SCHEDULER / DBMS_JOB: Administrarea job-urilor automate.

6. UTL_FILE: Operații cu fișiere text pe server.

7. UTL_MAIL / UTL_SMTP: Trimitere emailuri din PL/SQL.

8. DBMS_TRANSACTION, DBMS_LOCK: Control asupra tranzacțiilor și blocărilor.

Aceste pachete sunt esențiale pentru dezvoltarea aplicațiilor Oracle la nivel profesional.

Exerciții

a) Să se rețină despre fiecare proiect: începerea, sfârșitul, deadline-ul, numele, suma salarillor angajaților, numele și data angajării pentru cel mai bine plătit și cel mai prost plătit angajat din cadrul său. Apoi să se afișeze aceste informații sortate după mai multe criterii: alfabetic după numele angajaților, după durata proiectului, după diferența de salarii dintre cei 2 angajați, după distanța de la deadline la incheierea proiectului, după procentajul din salariul total pe care îl au cei 2 angajați, toate acestea putând fi crescător sau descrescător. Să se scrie toate aceste funcții într-un singur pachet.
b) Să se construiască în memoria programului o matrice care să rețină câți angajați dintr-un anumit departament au un anumit job. Spre exemplu, vom avea pe coloane departamentele și pe linii joburile, iar dacă sunt 5 angajați din al patrulea departament care au al doilea job, în tabel pe poziția (2,4), sau (1,3) cu indexare de la 0, va aparea valoarea 5. Să se folosească parcurgerea acestei matrici pentru a determina: combinația de job și departament cu cei mai mulți angajați, numărul de combinații care au 0 angajați și jobul ocupat de angajații din cele mai multe departamente.

a) For every project in the database, store its name, beginning date, end date, deadline, sum of salaries for employees and for both the best and worst paid employees, their names and hire_dates. Sort these projects by a few criteria: names of the employees, project duration, distance between project end date and deadline, percentage of the 2 employee's salaries in the entire project, all of these having the option to either be in increasing or decreasing order. Write all of these functions in a single package.
b) Build a matrix in the program's memory that stores how many employees in a certain department have a certain job. For example, if we store departments on the columns and jobs on the lines, and if we have 5 employees from the 4^{th} department that all have the 2^{nd} job, in our table we will have the value 5 at position (2,4), or (1,3) if indexed by 0. Use this matrix to determine the combination of job and department that has the most employees, the number of combinations that have 0 employees, and the job occupied by employees in the most departments.