Sistemi baza podataka

Slavica Aleksić

slavica@uns.ac.rs

Prikaz vrednosti izraza

- PL/SQL na nivou DBMS-a i SQL*Plus-a – kombinacija:
  - SET SERVEROUTPUT ON i
  - DBMS_OUTPUT.PUT_LINE (message)

Primeri predaje vrednosti izraza

```
DECLARE
        V_A NUMBER;
        S_A NUMBER := '10';
BEGIN
    V_A := S_A * 6/1.5;
    DBMS_OUTPUT.PUT_LINE('Stampa vrednosti za V_A');
    DBMS_OUTPUT.PUT_LINE('Vrednost za V_A je: ' ||
    V_A);
    DBMS_OUTPUT.PUT_LINE('Vrednost za V_A je: ' ||
    TO_CHAR(V_A));
END;
```

Ugrađivanje blokova i opseg delovanja promenljivih

```
DECLARE
-- opseg delovanja x – do kraja spoljnjeg bloka
  x  BINARY_INTEGER;
BEGIN
  DECLARE
-- opseg delovanja y – do kraja unutrašnjeg bloka
    y  PLS_INTEGER;
  BEGIN
      y := x;
  END;
END;
```

Ugrađivanje blokova i opseg delovanja promenljivih

- **NAPOMENA:** Naziv lokalno deklarisane konstrukcije ima prioritet, u odnosu na naziv globalno deklarisane konstrukcije

Ugrađivanje blokova i opseg delovanja promenljivih

```
DECLARE
  x  BINARY_INTEGER;     -- vidljivost: u spoljnjem bloku
BEGIN
  DECLARE
    x  VARCHAR2(20);
          -- vidljivost: samo u unutrašnjem bloku
    y  PLS_INTEGER;
          -- vidljivost: samo u unutrašnjem bloku
  BEGIN
     y := TO_NUMBER(x, '$99,990.00');
  END;
END;
```

Upotreba SQL naredbi u PL/SQL-u

- Dva načina upotrebe:
  - direktni
  - posredni, putem PL/SQL kursora

Direktni način upotrebe SELECT naredbe

```
SELECT select_list
INTO {variable[, variable]...
                | record_variable}
FROM table
[WHERE condition]
...
```

Direktni način upotrebe SELECT naredbe

- SELECT naredba mora da vrati **JEDAN I SAMO JEDAN** red
- U protivnom, dolazi do pokretanja odgovarajućih izuzetaka
- Klauzula INTO obezbeđuje memorisanje vrednosti preuzete (selektovane) torke
- U izrazima, upotrebljenim u okviru naredbe SELECT, moguće je referenciranje na PL/SQL i bind (host) promenljive
- NAPOMENA: važno je poštovati konvencije imenovanja promenljivih, kolona tabela i samih tabela

Primeri direktne upotrebe naredbe SELECT

```
DECLARE
   v_Count NUMBER(3);
BEGIN
   SELECT COUNT(*)
   INTO   v_Count
   FROM   Projekat;
   DBMS_OUTPUT.PUT_LINE(v_Count);
END;
```

Primeri direktne upotrebe naredbe SELECT

```
DECLARE
   V_Spr Projekat.Spr%TYPE := 10;
   V_Nap Projekat.Nap%TYPE;
   V_Nar Projekat.Nar%TYPE;
BEGIN
   SELECT Spr, Nap, Nar
   INTO   V_Spr, V_Nap, V_Nar
   FROM   Projekat
   WHERE Spr = V_Spr;
   DBMS_OUTPUT.PUT_LINE(v_Spr);
   DBMS_OUTPUT.PUT_LINE(v_Nap);
   DBMS_OUTPUT.PUT_LINE(v_Nar);
END;
```

Implicitni SQL kursor

- Sve SQL naredbe se parsiraju i izvršavaju u okviru kursorskih područja
- DML naredbama, koje se izvršavaju u PL/SQL bloku, dodeljuju se kursorska područja (kursori), čiji je programski naziv SQL
  - Implicitni SQL kursor
- Moguće je ispitivanje statusa implicitnog SQL kursora, nakon svake izvršene DML naredbe

Implicitni SQL kursor

- Funkcije ispitivanja statusa implicitnog SQL kursora
  - SQL%FOUND
    - TRUE, ako je bar jedan red bio predmet poslednje DML operacije, inače FALSE
  - SQL%NOTFOUND
    - TRUE, ako ni jedan red nije bio predmet poslednje DML operacije, inače FALSE
  - SQL%ROWCOUNT
    - broj redova, koji su bili predmet poslednje DML operacije
  - SQL%ISOPEN
    - uvek ima vrednost FALSE.
    - Upravljanje (otvaranje i zatvaranje) implicitnim kursorima je uvek automatsko. Neposredno nakon svake DML operacije, SQL kursorsko područje se automatski zatvori.

Primer

```
BEGIN
  UPDATE Projekat
  SET Nap = ''
  WHERE 1=2;
  DBMS_OUTPUT.PUT_LINE('Jedan update sa
   WHERE USLOVOM 1=2');
  DBMS_OUTPUT.PUT_LINE(sql%rowcount || '
   zapisa');
END;
```

DML naredbe

- Normalna upotreba naredbi INSERT, UPDATE i DELETE

Primeri upotrebe DML naredbi

```
ACCEPT D_Prz PROMPT 'Unesite prezime: '
ACCEPT D_Ime PROMPT 'Unesite ime: '

BEGIN
   INSERT INTO Radnik (Mbr, Prz, Ime, God)
   VALUES (SEQ_Mbr.NEXTVAL, '&D_Prz', '&D_Ime',
   SYSDATE);
   IF SQL%FOUND THEN
         DBMS_OUTPUT.PUT_LINE('Dodata nova torka u
   tabelu Radnik.');
   ELSE
         DBMS_OUTPUT.PUT_LINE('Unos torke u tabelu
   Radnik nije uspeo.');
   END IF;
END;
```

Primeri upotrebe DML naredbi

```
DECLARE
  v_Mbr radnik.mbr%TYPE := 203;
  broj_del NUMBER;
BEGIN
  DELETE FROM radnik
  WHERE  mbr = v_Mbr;
  broj_del := SQL%ROWCOUNT;
  DBMS_OUTPUT.PUT_LINE('Obrisano je: '||
   broj_del || ' radnika');
END;
```

Naredbe za upravljanje tokom izvođenja programa

- Naredba selekcije
- Naredbe iteracije

Naredba selekcije

```
IF logički_izraz THEN
  blok_izvršnih naredbi;
[ELSIF logički_izraz THEN
  blok_izvršnih naredbi;
]...
[
ELSE
  blok_izvršnih naredbi;
]
END IF;
```

Naredbe iteracije

- Bezuslovna (beskonačna) iteracija / LOOP

```
LOOP
   blok_izvršnih_naredbi;
END LOOP;
```

- Uslovna iteracija, s testom uslova na početku / WHILE LOOP

```
WHILE logički_izraz LOOP
  blok_izvršnih_naredbi;
END LOOP;
```

Naredbe iteracije

- Brojačka iteracija / FOR LOOP

  FOR brojač IN [REVERSE]
  donja_granica..gornja_granica LOOP

  blok_izvršnih_naredbi;

  END LOOP;

  NAPOMENA: Brojačku promenljivu *brojač* nije potrebno deklarisati.

  Korak brojača je uvek 1.

Izlazak iz petlje / EXIT

- EXIT [labela] [WHEN logički_izraz]

- EXIT se, najčešće, koristi u kombinaciji s bezuslovnom pteljom LOOP ... END LOOP
  - Obezbeđenje formiranja uslovne petlje, s mogućnošću testa uslova petlje na bilo kojoj poziciji u petlji

```
<<labela>>
LOOP
   ...
   EXIT [labela] [WHEN logički_izraz]
   ...
END LOOP;
```

Primeri upotrebe konstrukcija za upravljanje tokom izvođenja programa

```
BEGIN
   FOR i IN REVERSE 1..3 LOOP
        DBMS_OUTPUT.PUT_LINE('Vrednost brojaca i je: '
   || TO_CHAR(i));
   END LOOP;
END;
```

```
BEGIN
   FOR i IN 1..3 LOOP
        DBMS_OUTPUT.PUT_LINE('Vrednost brojaca i je: '
   || TO_CHAR(i));
   END LOOP;
END;
```

Primeri upotrebe konstrukcija za upravljanje tokom izvođenja programa

```
DECLARE
   i NUMBER(1) := 1;
BEGIN
   WHILE i <= 3 LOOP
        DBMS_OUTPUT.PUT_LINE('Vrednost
   brojaca i je: ' || TO_CHAR(i));
        i := i + 1;
   END LOOP;
END;
```

Primeri upotrebe konstrukcija za upravljanje tokom izvođenja programa

```
DECLARE
   i NUMBER(1) := 1;
BEGIN
   LOOP
        EXIT WHEN i > 3;
        DBMS_OUTPUT.PUT_LINE('Vrednost
   brojaca i je: ' || TO_CHAR(i));
        i := i + 1;
   END LOOP;
END;
```

Primeri upotrebe konstrukcija za upravljanje tokom izvođenja programa

```
DECLARE
   i NUMBER(1) := 0;
BEGIN
   LOOP
        i := i + 1;
        DBMS_OUTPUT.PUT_LINE('Vrednost
   brojaca i je: ' || TO_CHAR(i));
        EXIT WHEN i >= 3;
   END LOOP;
END;
```

Primeri upotrebe konstrukcija za upravljanje tokom izvođenja programa

```
ACCEPT N PROMPT 'N: '
BEGIN
   FOR i IN 1..&N LOOP
        IF MOD(i, 2) = 0 THEN
                DBMS_OUTPUT.PUT_LINE(i || ' je paran
   broj.');
        ELSIF MOD(i, 2) = 1 THEN
                DBMS_OUTPUT.PUT_LINE(i || ' je neparan
   broj.');
        ELSE
                DBMS_OUTPUT.PUT_LINE('Nemoguc
   slucaj.');
        END IF;
   END LOOP;
END;
```

Zadatak

Napisati PL/SQL blok koji će:

- interaktivno prihvatiti vrednosti za Mbr, Prz, Ime, Sef, Plt i God,
- dodati novu torku u tabelu Radnik, s prethodno preuzetim podacima i
- angažovati novododatog radnika na projektu sa Spr = 10 i 5 sati rada.

Rešenje

```
BEGIN
   INSERT INTO radnik (Mbr, Prz, Ime, Plt, God)
   VALUES (&&Mbr, '&&Prz', '&&Ime', &&Plt,
   '&&God');
   INSERT INTO radproj (Mbr, Spr, Brc)
   VALUES (&&Mbr, 10, 5);
   COMMIT;
END;
```

Zadatak

Napisati PL/SQL blok koji će:

- izbrisati angažovanje prethodno dodatog radnika na projektu sa šifrom 10 i obavestiti porukom korisnika da li je brisanje uspešno obavljeno,
- izbrisati prethodno dodatog radnika iz evidencije i obavestiti porukom korisnika da li je brisanje uspešno obavljeno,
- sačuvati vrednost za Mbr izbrisanog radnika u lokalnoj promenljivoj pod nazivom *Del_Mbr*

Rešenje

```
ACCEPT v_Mbr PROMPT 'MBR = '

DECLARE
    Del_Mbr radnik.Mbr%TYPE;
BEGIN
    DELETE FROM radproj
    WHERE Mbr = &v_Mbr AND Spr = 10;
    IF SQL%FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Brisanje rada na projektu uspesno obavljeno.');
    ELSE
            DBMS_OUTPUT.PUT_LINE('Brisanje rada na projektu nije uspesno
    obavljeno.');
    END IF;

    DELETE FROM radnik
    WHERE Mbr = &v_Mbr ;
    IF SQL%FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Brisanje radnika uspesno obavljeno.');
    ELSE
            DBMS_OUTPUT.PUT_LINE('Brisanje radnika nije uspesno obavljeno.');
    END IF;
    Del_Mbr := &v_Mbr;
END;
```

Zadatak

Kreirati tabelu Spisak_zarada, korišćenjem SQL komande:

CREATE TABLE Spisak_zarada (Mbr NUMBER(3),

Plt NUMBER(10, 2), Evri VARCHAR2(10),
CONSTRAINT Sz_PK PRIMARY KEY (Mbr))

Napisati PL/SQL blok koji će:

za svaku torku iz tabele Radnik, za koju je matični broj u intervalu od 10 do 100, izuzimajući radnika s matičnim brojem 90, preneti u tabelu Spisak_zarada matični broj, iznos plate, i inicijalizovati polje Evri sa vrednošću plate u evrima. Ukoliko radnik već postoji u tabeli izvršiti izmenu vrednosti obeležja Plt i Evri. Kurs evra treba da zadaje korisnik iz okruženja.

Rešenje

```
ACCEPT E PROMPT 'Kurs evra je: '
DECLARE
     v_Plt Spisak_zarada.Plt%TYPE;
       broj NUMBER :=0;
BEGIN
     FOR i IN 1..10 LOOP
              IF i != 9 THEN
                   SELECT Plt INTO v_Plt FROM Radnik
                   WHERE Mbr = 10*i;
                     SELECT COUNT(*) INTO broj FROM Spisak_zarada
                     WHERE Mbr = 10*i;
                     IF broj = 0 THEN
                       INSERT INTO Spisak_zarada (Mbr, Plt, Evri)
                       VALUES (10*i, v_Plt, v_Plt/&E );
                     ELSE
                       UPDATE Spisak_zarada
                       SET Plt = v_Plt,
                       Evri = v_Plt/&E
                       WHERE Mbr = 10*i;
                    END IF;
              END IF;
     END LOOP;
END;
```

Zadatak

Napisati PL/SQL blok koji će:

Proveravati ima li radnika sa platom manjom od zadate. Ako ima povećati premiju za 20% svakom radniku koji ima takvu platu. Ukoliko radnik nema uopste premiju dodeliti mu premiju od 5000. Ako svi radnici imaju platu veću od zadate ispisati poruku o tome.