PL/SQL

Aret Leetsaar

Sissejuhatuseks

- Procedural Language/SQL on laiendus SQL keelele, mis on mõeldud relatsiooniliste andmebaaside kasutamiseks.
- PL/SQL sisaldab mitmeid võimalusi programmeerimis-keeltelt, mis disainiti aastatel 1970 kuni 1980, näiteks plokkstruktuuride ja protseduuride kasutamine ning kõikvõimalikud manipulatsioonid keeruliste andmetüüpidega. Mis teeb PL/SQL-ist võimsa tööriista transaktsioonide haldamiseks.
- Keelt on võimalik kompileerida masinkoodi tasemele ja transleerida P-koodiks. Lisaks on võimalik kasutada teistes keeltes kirjutatud teeke pakettide vahendusel.

Sissejuhatuseks(1)

- PL/SQL-i on võimalik kasutada nii serveri poolel (salvestatud protseduurid ja funktsioonid, andmebaasi trigerid, paketid) kui ka rakenduse poolel (rakenduses trigeritena või funktsioonide ja protseduuridena).

- Eeliseks PL/SQL kasutamisel on see, et andmebaasiga suhtlemisel saadetakse andmeid tervikliku plokina ja ei tekitata asjatut võrguliiklust iga pisema SQL lause jaoks eraldi. Aruannete ja rakenduste koostamisel võimaldab PL/SQL lisada funktsionaalsust ja on võimas tööriist kui tähtis on kiire tulemuse saavutamine arendustöödes.

Süntaksist

- Translaator on tõstutundetu
- Kõik käsud lõpevad semikooloniga
- Ühe realine kommentaar algab -- ja lõigu kommentaar tähistatakse sarnaselt C keelele /* kommentaar */
- Programmi plokki alustab BEGIN ja lõppeb END

Näide (anonüümse ploki struktuur)

```
DECLARE -- soovituslik
        i_vanus         NUMBER;
        s_nimi          VARCHAR2 (150):='Jaan Jalgratas';
        d_synd          DATE;

BEGIN           -- kohustuslik
        SELECT eesnimi||' '||perenimi nimi, synni_kp
        INTO s_nimi, d_synd
        FROM kliendid WHERE id=1;

        i_vanus := sysdate – d_synd;

        dbms_output.put_line(s_nimi||' on '||i_vanus);
EXCEPTION     -- soovituslik
        WHEN no_data_found THEN
              null;
END;        -- kohustuslik
```

Näide (protseduur)

```
-- protseduur a+b

CREATE OR REPLACE PROCEDURE liida
        ( p_a IN NUMBER, p_b IN NUMBER)
IS
        s    NUMBER;

BEGIN
        s :=      p_a+p_b;
        dbms_output.put_line('Summa: '||s);
END liida;
/
```

Näide (funktsioon)

```
-- funktsioon a+b

CREATE OR REPLACE FUNCTION liida
        ( p_a IN NUMBER, p_b IN NUMBER)
RETURN NUMBER
IS
        s    NUMBER;

BEGIN
        s :=      p_a+p_b;
        RETURN s;
END liida;
/
```

Andmetüübid

- Skalaarsed – enamasti samad mis veeru tüübiks Oracle andmebaasis, aga toetatakse ka booleani
- Komposiitsed – võimaldab kirjeid koostada ja nendega manipuleerida
- Viidatüübid – viidatakse teatud tüüpi andmetele
- LOB (large objects) – andmetüüp suurema hulga binaarse info säilitamiseks (pildid näiteks)

Peamised skalaarsed andmetüübid

- CHAR [(maksimaalne pikkus)]
- VARCHAR2 [(maksimaalne pikkus)]
- LONG
- LONG RAW
- NUMBER [(täpsus, skaala)]
- BINARY_INTEGER
- PLS_INTEGER
- BOOLEAN
- DATE
- TIMESTAMP

Muutujate deklareerimine

```
DECLARE

nimi      VARCHAR2 (100);
maksud    NUMBER (9,2) := 0;
aegub     DATE := SYSDATE+14;
boonus    CONSTANT NUMBER(3,2) := 8.25;
kaup      kaubad.nimetus%TYPE;
```

Näide (FOR tsükkel kursoriga)

```
-- Tsükkel üle tabeli kasutajad
FOR kursor IN (
        SELECT id, nimi, vanus FROM kasutajad
) LOOP

  dbms_output.put_line('Kasutaja '||kursor.nimi||' ID: ' || kursor.id);

END LOOP;
```

Operaatorid

**	Astendamine
+, -	Arvu märk
*, /	Korrutamine, jagamine
+, -, ||	Liitmine, lahutamine, konkatenatsioon
=, <, >, <=, >=, <>, !=, ~=, ^=, IS NULL, LIKE, BETWEEN, IN	Võrdulused
NOT AND OR	Loogilised operaatorid

Programmi täitmine

Programmi täitmine toimub ülevalt alla ja järjest täidetakse kõiki ridu vastavalt hargnemistele kuni vea ilmnemiseni, mis suunab täitmise esimesse kõige lähema taseme erindisse (kui alamprogrammis pole veatöötlust, siis on võimalik viga püüda peaprogrammis).

Alamprogrammidele saab parameetreid anda kaasa ainult lugemiseks või ka muutmiseks (vastavalt siis IN ja OUT alamprogrammi deklareerimisel).

Funktsioonid ja protseduurid võivad paikneda eraldi asuvatena või on koondatud ühtsesse paketti, mis koosneb kerest ja deklaratiivsest osast. Paketi päises deklareeritud andmetüüpe, funktsioone ja protseduure saab kasutada erinevate rakenduste poolt vastavalt õigustele.

Hargnemiseks vajalik

- IF-THEN-ELSE
- CASE
- GOTO
- LOOP tsükkel
- FOR tsükkel
- CURSOR FOR tsükkel
- WHILE tsükkel
- REPEAT UNTIL tsükkel
- EXIT

IF-THEN-ELSE

```
IF tingimus THEN
        {...programmi plokk...}
ELSIF condition THEN
        {...programmi plokk...}
ELSE
        {...programmi plokk...}
END IF;
```

CASE

```
CASE :valiku.muutuja
WHEN 'variant 1' THEN
        -- täidetakse, kui muutuja väärtuseks on variant 1

WHEN 'variant 2' THEN
        -- täidetakse, kui muutuja väärtuseks on variant 2

ELSE
        -- täidetakse ülejäänud juhtudel

END CASE;
```

FOR tsükkel, tsükkel, WHILE tsükkel

```
FOR i IN [REVERSE]  min_nr..max_nr
LOOP
    {…korduv plokk…}
END LOOP;

LOOP
    {…korduv plokk…}
    EXIT WHEN lopu_tingimus;
END LOOP;

WHILE tingimus
LOOP
    {…korduv plokk…}
END LOOP;
```

Sisend/Väljund

Andmebaas peamiselt, aga lisaks on veel võimalused kasutada sisendi/väljundi jaoks vastavaid pakette, mis võimaldavad manipuleerida failidega, kasutada erinevaid interneti protokolle, suhelda kasutajaga interaktiivselt.

Keskkond

- PL/SQL on masinast sõltumatu
- Optimiseerida on võimalik pragmasid kasutades
- Ressursside hõivamine ja vabastamine käib automaatselt

Operatsioonikeskkond