




LABORATOR 2

Recapitulare
Introdurre PL/SQL



Vizualizari (Views)

- Tabel virtual obtinut prin aplicarea operatiei “select” asupra unor tabele existente
- Nu sunt create fizic pe disc
- Sunt inregistrate in tabelul “user_views”
- Sintaxa:
 - *CREATE VIEW name_of_view AS SELECT columns FROM table [WHERE some conditions...]*

Vizualizari (Views)

Rulati scriptul database_deploy_script.sql

1. Creati o vizualizare care sa aduca studentii cu bursa > 1350.
2. Sa se actualizeze tabelul students astfel: bursa studentului cu id 1 sa aiba valoarea 1400. Analog actualizati bursa la valoarea 1200.
3. Aceleasi actualizari pe vizualizarea creata anterior.
4. Incercati sa inserati urmatoarele randuri in vizualizare. Observati si comentati.

```
insert into best values(1998,'123AB1','f1','l1',3,'B1',200,sysdate, 'aaa@gmail.com',  
sysdate, sysdate);  
insert into best values(1999,'123AB2','f2','l2',2,'B2',1400,sysdate, 'abc@gmail.com',  
sysdate, sysdate);
```
5. Stergeti randurile inserate si modificati view-ul incat inserarea sa fie permisa doar pentru inregistrarile care vor putea fi afisate din view. Inserati din nou inregistrarile anterioare.
6. Creati o vizualizare care sa contina coloanele fname, lname, scholarship. Inserati o noua inregistrare prin intermediul vizualizarii.

Variabile de substitutie

- sunt utile in crearea de comenzi/script-uri dinamice
- se pot folosi pentru stocarea temporara de valori, transmiterea de valori intre comenzi *SQL* etc.
- Pot fi create prin:
 - comanda *DEFINE*(*DEFINE* variabila = valoare;)
 - Prefixarea cu & (indica existenta unei variabile intr-o comanda *SQL*, daca variabila nu exista, atunci ea este creata)
 - Prefixarea cu && (indica existenta unei variabile intr-o comanda *SQL*, daca variabila nu exista, atunci *ea este creata*). Deosebirea fata de & este ca, daca se foloseste &&, atunci referirea ulterioara cu & sau && nu mai cere ca utilizatorul sa introduca de fiecare data valoarea variabilei. Este folosita valoarea data la prima referire.
- Variabilele de substitutie pot fi eliminate cu ajutorul comenzii *UNDEF[INE]*
- Variabilele de tip *DATE* sau *CHAR* trebuie sa fie incluse intre apostrofuri in comanda *SELECT*
- inlocuiesc/substituie in cadrul comenzii *SQL* variabila respectiva cu sirul de caractere introdus de utilizator
- o variabila ramane pana la eliminarea ei cu o comanda *UNDEF* sau pana la terminarea sesiunii *SQL* respective

Comanda	Descriere
<i>ACC[EPT]</i> variabila [tip] [<i>PROMPT text</i>]	Citește o linie de intrare și o stochează într-o variabilă utilizator.
<i>PAU[SE]</i> [text]	Afișează o linie vidă, urmată de o linie conținând text, apoi așteaptă ca utilizatorul să apese tasta <i>return</i> . De asemenea, această comandă poate lista două linii vide, urmate de așteptarea răspunsului din partea utilizatorului.
<i>PROMPT</i> [text]	Afișează mesajul specificat sau o linie vidă pe ecranul utilizatorului.

DEFINE variabila = valoare	Creaza o variabila utilizator cu valoarea de tip sir de caracter precizata.
DEFINE variabila	Afiseaza variabila, valoarea ei si tipul de data al acesteia.
DEFINE	Afiseaza toate variabilele existente in sesiunea curenta, impreuna cu valorile si tipurile lor de date.

- Sa se afiseze codul, numele, salariul si codul departamentului din care face parte pentru un angajat al carui cod este introdus de utilizator de la tastatura.

I. SELECT employee_id, last_name, salary, department_id
FROM employees WHERE employee_id = &p_cod;

II. DEFINE p_cod;
SELECT employee_id, last_name, salary, department_id
FROM employees WHERE employee_id = &p_cod;
UNDEFINE p_cod;

III. DEFINE p_cod=100;
SELECT employee_id, last_name, salary, department_id
FROM employees WHERE employee_id = &&p_cod;
UNDEFINE p_cod;

IV. ACCEPT p_cod PROMPT "cod= ";
SELECT employee_id, last_name, salary, department_id
FROM employees WHERE employee_id = &p_cod;

Tabele temporare

- Stocheaza date numai pe durata unei tranzactii sau a intregii sesiuni
- Nu li se alocă spatiu la creare decat dacă s-a folosit clauza “AS subcerere”, altfel spatial este alocat la primul “insert”
- Comenzile LDD pot fi efectuate asupra unui table temporar doar dacă nu există nicio sesiune atasată acestuia (o sesiune este atasată unui table dacă efectuează o operație INSERT asupra acestuia)

```
CREATE GLOBAL TEMPORARY TABLE [schema.]nume_tabel  
  ( {nume_coloană_1 tip_date [DEFAULT expresie]  
    [constr_coloană [constr_coloană] ... ]  
    | constr_tabel_sau_view }  
    [, {nume_coloană_2 ... } ] )  
[ON COMMIT {DELETE | PRESERVE} ROWS;
```

Secvente

- Obiect ce permite generarea de nr. intregi unice
- Independente de tabele -> pot fi folosite pentru mai multe tabele
- Informatii -> dictionarul datelor: *USER_SEQUENCES*, *ALL_ SEQUENCES*, *DBA_ SEQUENCES*
- La definirea unei secvențe se pot specifica:
 - numele secvenței
 - diferența dintre 2 numere generate succesiv, implicit fiind 1 (INCREMENT BY);
 - numărul initial, implicit fiind 1 (START WITH);
 - valoarea maximă, implicit fiind 1027 pentru o secvență ascendentă și -1 pentru una descendentă;
 - valoarea minimă;
 - dacă secvența ciclează după ce atinge limita; (CYCLE)
 - câte numere să încarce în *cache server*, implicit fiind încărcate 20 de numere (CACHE).
- Pseudocoloanele ***NEXTVAL*** și ***CURRVAL*** permit lucrul efectiv cu secvențele.


```
CREATE SEQUENCE nume_secv  
[INCREMENT BY n]  
[START WITH n]  
[{MAXVALUE n | NOMAXVALUE}]  
[{MINVALUE n | NOMINVALUE}]  
[{CYCLE | NOCYCLE}]  
[{CACHE n | NOCACHE}]
```

- Creați o secvență pentru generarea codurilor de departamente, `SEQ_DEPT_PNU`. Secvența va începe de la 200, va crește cu 10 de fiecare dată și va avea valoarea maximă 10000, nu va cicla și nu va încărca nici un număr înainte de cerere.
- Să se selecteze informații despre secvențele utilizatorului curent (nume, valoare minimă, maximă, de incrementare, ultimul număr generat).
- În ce instrucțiuni se pot / nu se pot utiliza pseudocoloanele?

Obs: Pseudocoloanele se pot utiliza în:

- lista *SELECT* a comenzilor ce nu fac parte din subcereri;
- lista *SELECT* a unei cereri ce apare într un *INSERT*;
- clauza *VALUES* a comenzii *INSERT*;
- clauza *SET* a comenzii *UPDATE*.

Obs : Pseudocoloanele nu se pot utiliza:

- în lista *SELECT* a unei vizualizări;
- într-o comanda *SELECT* ce conține *DISTINCT*, *GROUP BY*, *HAVING* sau *ORDER BY*;
- într-o subcerere în comenzile *SELECT*, *UPDATE*, *DELETE*
- în clauza *DEFAULT* a comenzilor *CREATE TABLE* sau *ALTER TABLE*.

Indecsi

- Obiect al unei scheme utilizat pentru imbunatatirea performantei unui anumit tip de cereri asupra unui tabel
- evită scanarea completă a unui tabel la efectuarea unei cereri
- reduc operațiile de citire/scriere de pe disc utilizând o cale mai rapidă de acces la date și anume pointeri la liniile tabelului care corespund unor anumite valori ale unei chei (coloane)
- sunt independenți de tabelele pe care le indexează, în sensul că dacă sunt șterși nu afectează conținutul tabelelor sau comportamentul altor indecși
- sunt menținuți și utilizați automat de către server-ul Oracle
- la ștergerea unui tabel, sunt șterși și indecșii asociați acestuia
- Indecșii pot fi creați :
 - *automat: odată cu definirea unei constrangeri PRIMARY KEY sau UNIQUE*
 - *manual: cu ajutorul comenzii CREATE INDEX*

- Se creează un index atunci când:
 - O coloană conține un domeniu larg de valori;
 - O coloană conține nu număr mare de valori null;
 - Una sau mai multe coloane sunt folosite des în clauza WHERE sau în condiții de join în programele de aplicații
 - Tabelul este mare și de obicei cererile obțin mai puțin de 2%-4% din liniile tabelului.
- USER_INDEXES, ALL_INDEXES, ALL_IND_COLUMNS

```
CREATE [UNIQUE] INDEX index_name  
    ON table_name (column1, column2, ... column_n)  
    [ COMPUTE STATISTICS ];
```

Introducere PL/SQL

- PL/SQL este un limbaj cu structura de bloc, adică programele sunt compuse din blocuri care pot fi complet separate sau încuibărite unul în altul.
- Un program poate cuprinde unul sau mai multe blocuri. Un bloc poate fi anonim sau neanonim.
- Blocurile **anonime** sunt blocuri PL/SQL fără nume, care sunt construite dinamic și sunt executate o singură dată. Acest tip de bloc nu are argumente și nu returnează un rezultat.
- Blocurile **neanonime** sunt fie blocuri având un nume (etichetate), care sunt construite static sau dinamic și sunt executate o singură dată, fie subprograme, pachete sau declanșatori..

Structura unui bloc *PL/SQL* este compusă din trei secțiuni distincte:

Blocul *PL/SQL* are următoarea structură generală:

[<<*nume_bloc*>>]

[DECLARE

instrucțiuni de declarare]

BEGIN

instrucțiuni executabile (SQL sau PL/SQL)

[EXCEPTION

tratarea erorilor]

END [*nume_bloc*];

Dacă blocul *PL/SQL* este executat fără erori, va apărea mesajul:

PL/SQL procedure successfully completed

Compatibilitate SQL

- Din punctul de vedere al compatibilității PL/SQL versus SQL există următoarele reguli de bază:
- PL/SQL furnizează toate comenzile LMD ale lui SQL, comanda SELECT cu clauza INTO, comenzile LCD, funcțiile, pseudo-coloanele și operatorii SQL
- PL/SQL nu include comenzile LDD.
- Majoritatea funcțiilor SQL sunt disponibile în PL/SQL.
- Există funcții noi, specifice PL/SQL, cum sunt funcțiile SQLCODE și SQLERRM.
- Există funcții SQL care nu sunt disponibile în instrucțiuni procedurale (de exemplu, DECODE, NULLIF, funcțiile grup), dar care sunt disponibile în instrucțiunile SQL dintr-un bloc PL/SQL. SQL nu poate folosi funcții sau attribute specifice PL/SQL.
- ! Funcțiile grup trebuie folosite cu atenție, deoarece instrucțiunea SELECT ... INTO nu poate conține clauza GROUP BY.

Exercitii

1. Se consideră următorul bloc PL/SQL:

```
<<bloc>>
DECLARE
    v_cantitate NUMBER(3) := 300;
    v_mesaj     VARCHAR2(255) := 'Produs 1';
BEGIN
    <<subbloc>>
    DECLARE
        v_cantitate NUMBER(3) := 1;
        v_mesaj     VARCHAR2(255) := 'Produs 2';
        v_locatie   VARCHAR2(50) := 'Europa';
    BEGIN
        v_cantitate := v_cantitate + 1;
        v_locatie   := v_locatie || 'de est';
    END;
    v_cantitate := v_cantitate + 1;
    v_mesaj := v_mesaj || ' se afla in stoc';
    v_locatie := v_locatie || 'de est';
END;
/
```

Evaluati:

- valoarea variabilei v_cantitate în subbloc;
- valoarea variabilei v_locatie la poziția în subbloc ;
- valoarea variabilei v_cantitate în blocul principal ;
- valoarea variabilei v_mesaj în blocul principal ;
- valoarea variabilei v_locatie în blocul principal.

Exercitii

1. Creați un bloc anonim care să afișeze un mesaj pe ecran (mai multe variante).
2. Sa se creeze un bloc care sa calculeze si sa afiseze suma a 2 nr. citite de la tastatura.
3. Creati si executati un bloc PL/SQL care cere de la tastatura 2 numere (prin variabile de substitutie SQL* Plus). Calculati valoarea functiei. Rezultatul va fi retinut intr-o variabila PL/SQL si va fi tiparit pe ecran.

$$f(x, y) = \begin{cases} \frac{x}{y} + y, & \text{daca } y \neq 0 \\ x^2, & \text{daca } y = 0 \end{cases}$$

4. Să se calculeze suma salariilor pentru un job al cărui cod este introdus de utilizator. Căutarea se va face case-insensitive.