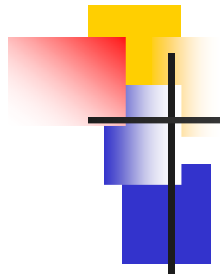


# CURS 10

## PL/SQL III



# Structuri de control

---

- Reprezintă cea mai importantă extensie față de SQL introdusă în PL/SQL
- Tipuri de structuri de control
  - conditional (IF-THEN, IF-THEN-ELSE, CASE...)
  - iterativ (FOR-LOOP, WHILE-LOOP...)
  - secvențial (GOTO)



# Structuri condiționale - IF

- Sintaxa generală:  
**IF condiții THEN**  
    **Instrucțiuni**  
**[ELSIF condiții THEN**  
    **Instrucțiuni]**  
**[ELSE**  
    **Instrucțiuni]**  
**END IF;**
- *condiții* – expresie evaluată la o valoare booleană (TRUE, FALSE) sau NULL
  - Dacă expresia condiției din instrucțiunea IF este NULL se sare automat la execuția instrucțiunilor din clauza ELSE
- *instrucțiuni*- una sau mai multe instrucțiuni PL/SQL sau SQL



# Structuri condiționale- IF-THEN-ELSE

```
DECLARE
    sold_cont NUMBER(11,2);
    cont CONSTANT NUMBER(4) := 3;
    suma_debit CONSTANT NUMBER(5,2) := 500.00;
BEGIN
    SELECT sold INTO sold_cont
    FROM conturi
    WHERE id_cont = cont FOR UPDATE OF sold;
    IF
        sold_cont >= suma_debit THEN
        UPDATE conturi SET sold = sold - suma_debit WHERE id_cont = cont;
    ELSE
        INSERT INTO temp
        VALUES (cont, sold_cont, 'Fonduri insuficiente');
    END IF;
COMMIT;
END;
```



# Structuri condiționale - CASE

- Sintaxa generală

**CASE selector**

**WHEN expresie1 THEN rez1**

**WHEN expresie2 THEN rez2**

**...**

**WHEN expresieN THEN rezN**

**[ELSE rezN+1]**

**END;**

- O expresie CASE selectează și întoarce un anumit rezultat
- Selecția unui anumit rezultat din mai multe posibile se bazează pe valorile la care sunt evaluate expresiile asociate cu selectorul, specificate în clauza WHEN



## Structuri condiționale – CASE II

- O variantă posibilă în PL/SQL este cea în care în clauza WHEN sunt considerate condiții de căutare

**CASE**

**WHEN cond\_c1 THEN rez1**

**WHEN cond\_c2 THEN rez2**

**...**

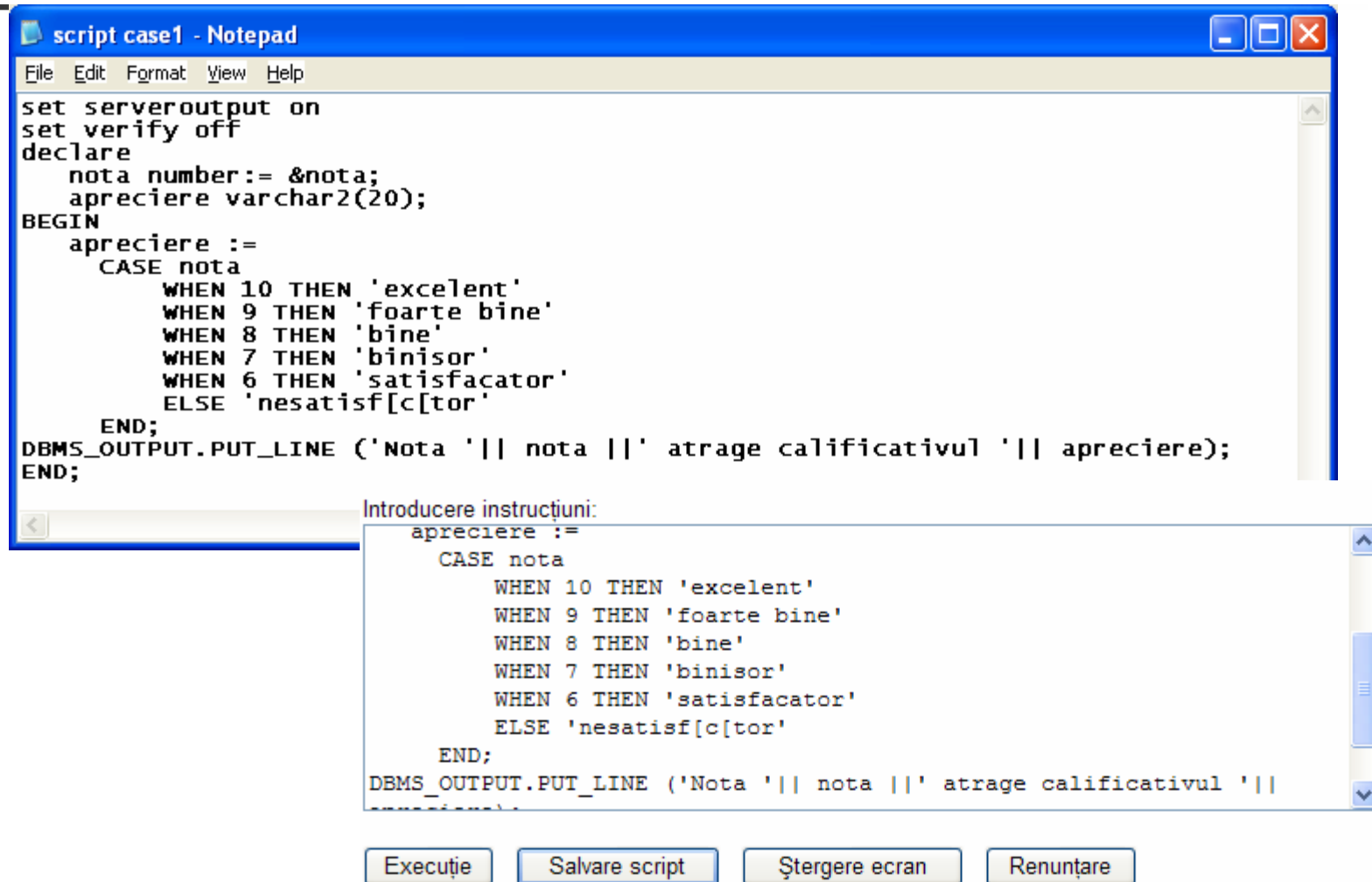
**WHEN cond\_cN THEN rezN**

**[ELSE rezN+1]**

**END;**

- În acest caz expresia CASE nu are selector

# Structuri condiționale - CASE



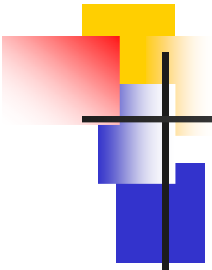
The screenshot shows a Notepad window titled "script case1 - Notepad" with a menu bar (File, Edit, Format, View, Help). The main text area contains the following PL/SQL code:

```
set serveroutput on
set verify off
declare
    nota number := &nota;
    apreciere varchar2(20);
BEGIN
    apreciere :=
        CASE nota
            WHEN 10 THEN 'excelent'
            WHEN 9 THEN 'foarte bine'
            WHEN 8 THEN 'bine'
            WHEN 7 THEN 'binisor'
            WHEN 6 THEN 'satisfacator'
            ELSE 'nesatisfacator'
        END;
    DBMS_OUTPUT.PUT_LINE ('Nota ' || nota || ' atrage calificativul ' || apreciere);
END;
```

Below the main text area, there is a dialog box titled "Introducere instrucțiuni:" containing the same code block, specifically the CASE statement and the DBMS\_OUTPUT.PUT\_LINE statement. The dialog box has a scroll bar on the right.

At the bottom of the Notepad window, there are four buttons: "Execuție", "Salvare script", "Ștergere ecran", and "Renunțare".

Nota 10 atrage calificativul excelent  
Procedură PL/SQL încheiată cu succes.



script case2 - Notepad

File Edit Format View Help

```
set serveroutput on
set verify off
declare
    nota number:= &nota;
    apreciere varchar2(20);
BEGIN
    apreciere :=
        CASE
            WHEN nota=10 or nota=9 THEN 'excelent'
            WHEN nota=8 or nota=7 THEN 'bine'
            WHEN nota=6 or nota=5 THEN 'satisfacator'
            ELSE 'nesatisfacator'
        END;
    DBMS_OUTPUT.PUT_LINE ('Nota '|| nota ||' atrage calificativul '|| apreciere);
END;
```

Introducere instrucțiuni:

```
apreciere :=
CASE
    WHEN nota=10 or nota=9 THEN 'excelent'
    WHEN nota=8 or nota=7 THEN 'bine'
    WHEN nota=6 or nota=5 THEN 'satisfacator'
    ELSE 'nesatisfacator'
END;
DBMS_OUTPUT.PUT_LINE ('Nota '|| nota ||' atrage calificativul '||
apreciere);
END;
```

Execuție

Salvare script

Ștergere ecran

Renunțare

Nota 8 atrage calificativul bine  
Procedură PL/SQL încheiată cu succes.





# Instrucțiunea CASE

- Dacă în clauza THEN respectiv ELSE se includ instrucțiuni PL/SQL atunci avem o instrucțiune CASE

Expresia CASE	Instrucțiunea CASE
<ul style="list-style-type: none"><li>-evaluează condițiile și întoarce o valoare</li><li>-este finalizată cu END;</li></ul>	<ul style="list-style-type: none"><li>-evaluează condițiile si execută diferite acțiuni</li><li>-poate fi un întreg bloc PL/SQL</li><li>-este finalizat cu END CASE</li></ul>

script instr case - WordPad

File Edit View Insert Format Help

set serveroutput on  
DECLARE  
deptid NUMBER;  
deptname VARCHAR2(20);  
emp\_tot NUMBER;  
man\_id NUMBER:=108;  
BEGIN  
CASE man\_id  
WHEN 108 THEN  
SELECT department\_id, department\_name  
INTO deptid, deptname  
FROM departments  
WHERE department\_id=108;  
SELECT count(\*) INTO emp\_tot  
FROM employees  
WHERE department\_id=deptid;  
WHEN 200 THEN  
SELECT department\_id, department\_name  
INTO deptid, deptname  
FROM departments  
WHERE department\_id=200;  
SELECT count(\*) INTO emp\_tot  
FROM employees  
WHERE department\_id=deptid;  
END CASE;  
DBMS\_OUTPUT.put\_line ('Lucrezi la departamentul ' || deptname || ' in care sunt in total ' || emp\_tot || ' angajati');  
END;

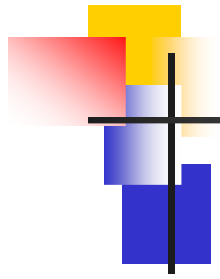
Introducere instructiuni:

```
set serveroutput on
DECLARE
  deptid NUMBER;
  deptname VARCHAR2(20);
  emp_tot NUMBER;
  man_id NUMBER:=10;
BEGIN
  CASE man_id
    WHEN 10 THEN
      SELECT department_id, department_name
```

Execuție Salvare script Ștergere ecran Renunțare

Lucrezi la departamentul Administration in care sunt in total 1 angajati  
Procedură PL/SQL încheiată cu succes.

For Help, press F1



# Structuri iterative - LOOP

- Permite execuția repetată a unei secvențe de instrucțiuni.
  - aceasta secvență se plasează între cuvintele cheie LOOP și END LOOP
- Exista trei tipuri de bucle:
  - de bază (basic loop)
  - bucle FOR - acțiuni iterative pe baza unui contor
  - bucle WHILE – acțiuni iterative pe baza unei condiții



# Bucle de bază

---

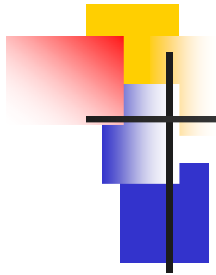
- Sintaxa generală

LOOP

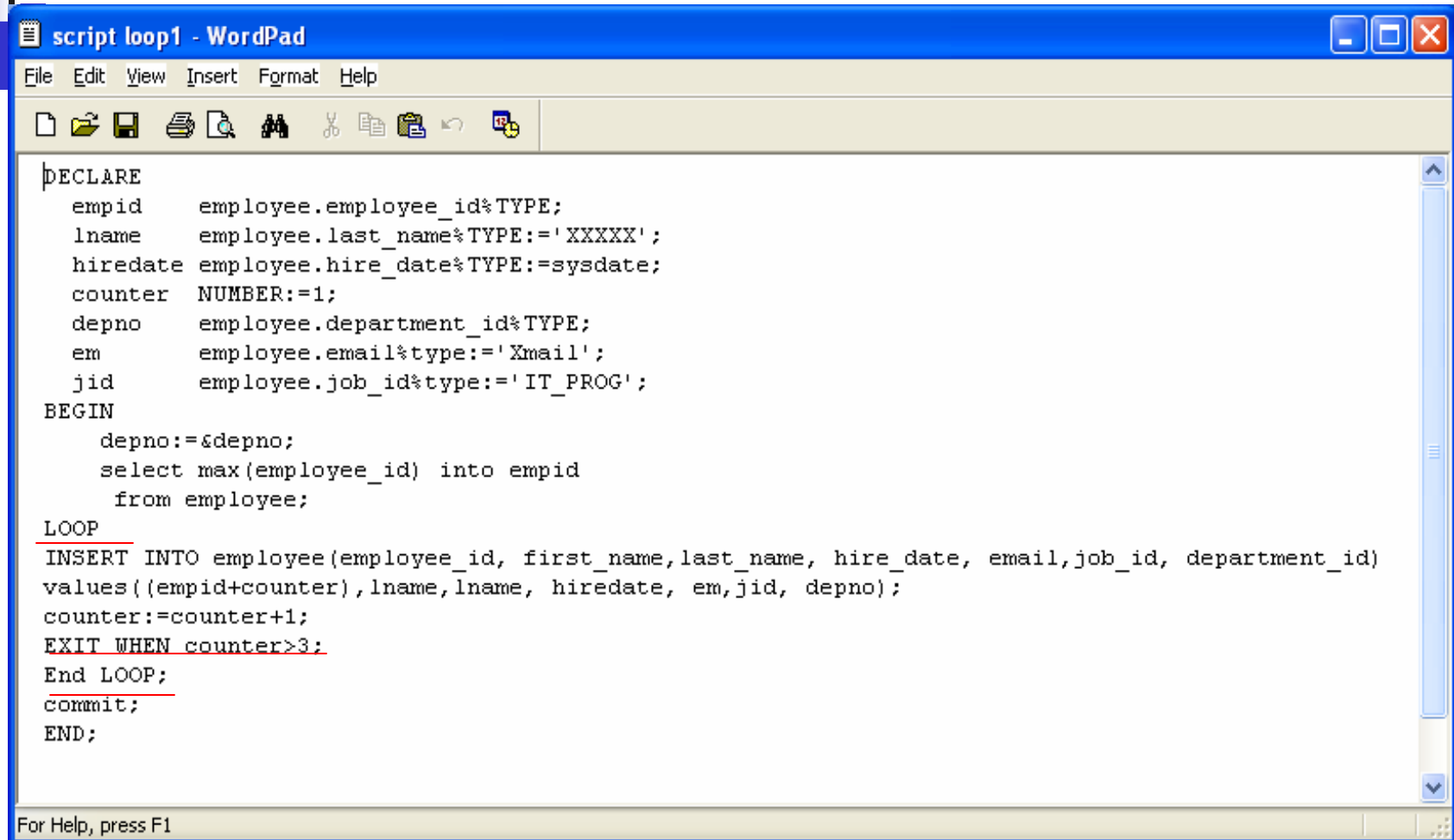
-- secvența de instrucțiuni

EXIT [WHEN conditie];

END LOOP;



- Instrucțiunea EXIT-WHEN permite completarea unei bucle daca prelucrarea devine imposibilă sau nedorita.
- La întâlnirea instrucțiunii EXIT se evaluează condiția din clauza WHEN.
  - Daca condiția este adevărată se iese din buclă si se trece la instrucțiunea următoare.



```
script loop1 - WordPad
File Edit View Insert Format Help

DECLARE
  empid      employee.employee_id%TYPE;
  lname      employee.last_name%TYPE:='XXXXX';
  hiredate   employee.hire_date%TYPE:=sysdate;
  counter    NUMBER:=1;
  depno      employee.department_id%TYPE;
  em         employee.email%type:='Xmail';
  jid        employee.job_id%type:='IT_PROG';
BEGIN
  depno:=&depno;
  select max(employee_id) into empid
    from employee;
LOOP
  INSERT INTO employee(employee_id, first_name, last_name, hire_date, email, job_id, department_id)
  values((empid+counter), lname, lname, hiredate, em, jid, depno);
  counter:=counter+1;
  EXIT WHEN counter>3;
End LOOP;
commit;
END;

For Help, press F1
```

Introducere instrucțiuni:

```

depno:=&depno;
select max(employee_id) into empid
from employee;
LOOP
INSERT INTO employee(employee_id, first_name,last_name, hire_date,
email,job_id, department_id)
values((empid+counter),lname,lname, hiredate, em,jid, depno);
counter:=counter+1;
EXIT WHEN counter>3;
End LOOP;

```

Execuție

Salvare script

Ștergere ecran

Renunțare

vechi 10: depno:=&depno;

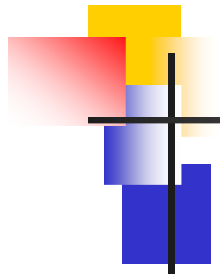
nou 10: depno:=10;

Procedură PL/SQL încheiată cu succes.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID
203	Susan	Mavris	SMAVRIS	515.123.7777	07-lun-1994 12:00:00 AM	HR_REP
204	Hermann	Baer	HBAER	515.123.8888	07-lun-1994 12:00:00 AM	PR_REP
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-lun-1994 12:00:00 AM	AC_MGR
206	William	Gietz	WGIEZT	515.123.8181	07-lun-1994 12:00:00 AM	AC_ACC...
207	XXXXX	XXXXX	Xmail		12-Dec-2009 08:58:18 ...	IT_PROG
208	XXXXX	XXXXX	Xmail		12-Dec-2009 08:58:18 ...	IT_PROG
209	XXXXX	XXXXX	Xmail		12-Dec-2009 08:58:18 ...	IT_PROG

Execute time (s): 0.015 Rows returned: 108

Apply Revert Show SQL Close Help



# Bucle FOR

- Permit specificarea unui domeniu de numere întregi, apoi execută o secvență de instrucțiuni, câte o dată pentru fiecare întreg din gama
  - Modalitate de a scurta testul pentru numărul de iterații
  - NU se declară contorul (este declarat implicit)
- Sintaxa generală:

```
FOR contor IN [REVERSE]  
    lim_inferioară .. lim_superioara LOOP  
    instrucțiuni  
    ...  
END LOOP;
```



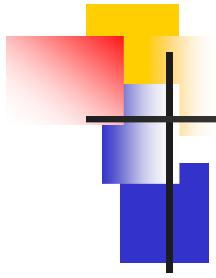
# Exemplu

script FOR loop - Notepad

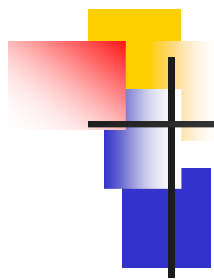
```
File Edit Format View Help
DECLARE
  empid      employee.employee_id%TYPE;
  lname      employee.last_name%TYPE:= 'YYYYY';
  hiredate   employee.hire_date%TYPE:= sysdate;
  depno      employee.department_id%TYPE;
  em         employee.email%type:= 'Ymail';
  jid        employee.job_id%type:= 'IT_PROG';
BEGIN
  depno:=&depno;
  select max(employee_id) into empid
  from employee;
  for i IN 1..3 LOOP
    INSERT INTO employee(employee_id, first_name, last_name, hire_date, email, job_id, department_id)
    values((empid+i), lname, lname, hiredate, em, jid, depno);
  End LOOP;
  commit;
END;
```

209	XXXXX	XXXXX	Xmail		12-Dec-2009 08:58:18 PM	IT_PROG
210	XXXXX	XXXXX	Xmail		13-Dec-2009 05:14:27 PM	IT_PROG
211	XXXXX	XXXXX	Xmail		13-Dec-2009 05:14:27 PM	IT_PROG
212	XXXXX	XXXXX	Xmail		13-Dec-2009 05:14:27 PM	IT_PROG
213	YYYYY	YYYYY	Ymail		13-Dec-2009 05:16:04 PM	IT_PROG
214	YYYYY	YYYYY	Ymail		13-Dec-2009 05:16:04 PM	IT_PROG
215	YYYYY	YYYYY	Ymail		13-Dec-2009 05:16:04 PM	IT_PROG

Execute time (s): 0.031 Rows returned: 114 Apply Revert Show SQL Close Help



- Contorul trebuie referit exclusiv în interiorul buclei
  - In afara acesteia este nedefinit
- Contorul nu trebuie referit ca țintă pentru o atribuire
- Niciuna din limitele domeniului pentru contor nu trebuie sa fie NULL.



# Bucle WHILE

- Asociază o condiție unei secvențe de instrucțiuni
  - Înaintea fiecărei iterații este evaluată condiția
    - dacă rezultatul întors este TRUE secvența de instrucțiuni este executată
    - Dacă rezultatul este FALSE sau NULL se iese din buclă și se trece la instrucțiunea următoare
- Sintaxa generală:

**WHILE** **conditie** **LOOP**

**Instructiune**

.....

**END LOOP;**

script while loop - Notepad

```

File Edit Format View Help

DECLARE
  empid      employee.employee_id%TYPE;
  lname      employee.last_name%TYPE:= 'ZZZZZ';
  hiredate   employee.hire_date%TYPE:= sysdate;
  depno      employee.department_id%TYPE;
  em         employee.email%type:= 'Zmail';
  jid        employee.job_id%type:= 'IT_PROG';
  counter    NUMBER:=5;
BEGIN
  depno:=&depno;

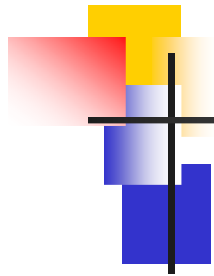
```

Table Editor : "HR"."EMPLOYEE" - hr@ORACLE09

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID
201	Michael	Hartstein	MHARTSTE	515.123.5555	17-Feb-1996 12:00:00 AM	MK_MAN
202	Pat	Fay	PFAY	603.123.6666	17-Aug-1997 12:00:00 AM	MK_REP
203	Susan	Mavris	SMAVRIS	515.123.7777	07-lun-1994 12:00:00 AM	HR_REP
204	Hermann	Baer	HBAER	515.123.8888	07-lun-1994 12:00:00 AM	PR_REP
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-lun-1994 12:00:00 AM	AC_MGR
206	William	Gietz	WGIEZT	515.123.8181	07-lun-1994 12:00:00 AM	AC_ACCOUNT
211	ZZZZZ	ZZZZZ	Zmail		13-Dec-2009 05:34:25 ...	IT_PROG
210	ZZZZZ	ZZZZZ	Zmail		13-Dec-2009 05:34:25 ...	IT_PROG
209	ZZZZZ	ZZZZZ	Zmail		13-Dec-2009 05:34:25 ...	IT_PROG
208	ZZZZZ	ZZZZZ	Zmail		13-Dec-2009 05:34:25 ...	IT_PROG
207	ZZZZZ	ZZZZZ	Zmail		13-Dec-2009 05:34:25 ...	IT_PROG

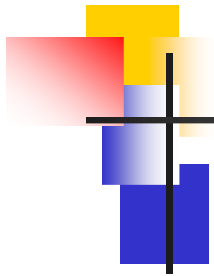
Execute time (s): 0.031 Rows returned: 110

Apply Revert Show SQL Close Help



# Bucle imbricate si etichete

- Pot fi încapsulate bucle pe nivele multiple
- Se utilizează etichete pentru a distinge între blocuri si bucle
  - Eticheta se plasează inaintea primei instructiuni, pe aceeași linie sau pe linii separate.
  - Eticheta este un identificator nedeclarat plasat între delimitatori astfel:  
`<<nume_eticheta>>`
- Ieșirea din bucla exterioară se face cu instrucțiunea EXIT care referă eticheta



# Exemplu

---

```
...
BEGIN
  << bucla ext>>
  LOOP
    Contor:=contor+1;
  EXIT WHEN contor>10;
    <<bucla_int>>
    LOOP
      .....
      EXIT bucla_ext WHEN sum_sal>1000;
      --se iese din ambele bucle
      EXIT WHEN optiune='DA';
      -- se iese din bucla interioara
      .....
    END LOOP bucla_int;
    .....
  END LOOP bucla_ext;
END;
```



# Structuri de control secvențial- GOTO

- Permite saltul necondiționat la o etichetă ce precede o instrucțiune executabilă sau un bloc PL/SQL

```
IF rating > 90 THEN
```

```
    GOTO calc_raise; -- branch to label
```

```
END IF;
```

```
...
```

```
<<calc_raise>>
```

```
IF job_title = 'SALESMAN' THEN -- control resumes here
```

```
    amount := commission * 0.25;
```

```
    ELSE
```

```
        amount := salary * 0.10;
```

```
END IF;
```



# Manipularea excepțiilor

---

- Excepțiile sunt erori PL/SQL apărute în timpul execuției programului
  - La apariția unei astfel de erori execuția blocului se termină
    - Se poate specifica un handler care să execute o serie de acțiuni înaintea finalizării blocului
- Excepțiile pot fi ridicate:
  - Implicit de către serverul Oracle
  - Explicit prin program
- O excepție poate fi manipulată:
  - Prin capturarea sa cu ajutorul unui handler
  - Prin propagarea sa în mediul apelant





# O excepție poate fi manipulată:

- Prin capturarea sa cu ajutorul unui handler
  - Presupune includerea unei secțiuni EXCEPTION în programul PL/SQL pentru a captura excepția
    - Dacă excepția este ridicată în secțiunea executabilă a blocului se prelucrează ramura corespunzătoare excepției din secțiunea EXCEPTION a blocului
    - Dacă excepția este manipulată corect ea nu se va propaga în blocul ce o include sau în mediul apelant
    - Se va considera că blocul PL/SQL este finalizat cu succes
- Prin propagarea sa în mediul apelant
  - Dacă excepția este ridicată în secțiunea executabilă a blocului și nu există un handler corespunzător, blocul se termină cu o eroare și excepția se propagă în mediul apelant



# Tipuri de excepții

Excepții	Descriere	Mod de manipulare
Erori predefinite în serverul Oracle	Aprox. 1/20 din erorile apărute în blocurile PL/SQL	Nu se declara, ele sunt ridicate implicit de serverul Oracle
Erori nedefinite în serverul Oracle	Orice alte erori standard din serverul Oracle	Se declară în secțiunea declarativă și permit serverului Oracle să le ridice implicit
Erori definite de utilizatori	Condiții pe care programatorul le consideră a fi anormale	Se declară în secțiunea declarativă și sunt ridicate explicit



# Capturarea excepțiilor

- Orice eroare poate fi capturată și tratată prin includerea unui handler corespondent în secțiunea de manipulare a excepțiilor din blocul PL/SQL
  - Fiecare handler constă într-o clauză WHEN care specifică numele unei excepții urmat de o succesiune de instrucțiuni ce trebuie executate când este ridicată excepția respectivă
  - În secțiunea de excepții pot fi incluse oricâte handlere
  - Nu sunt admise mai multe handlere pentru aceeași excepție

- 
- Sintaxa generală:

EXCEPTION

WHEN exceptie1 [OR exceptie2...] THEN

Instructiune1;

Instructiune2;

...

[WHEN exceptie3 [OR exceptie4...] THEN

Instructiune1;

Instructiune2;

.....]

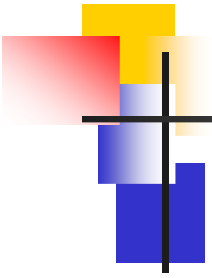
[WHEN OTHERS THEN

Instructiune1;

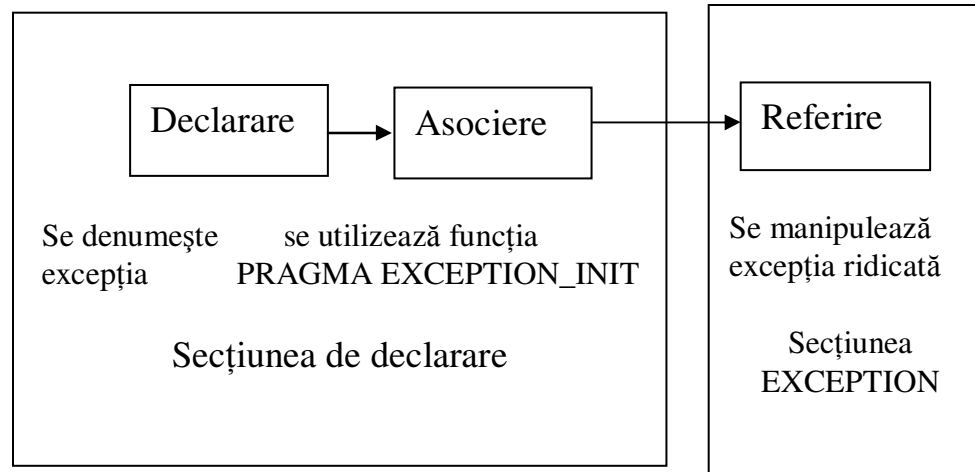
Instructiune2;

...]

# Exceptii generate de erori predefinite în serverul Oracle

- 
- Rutina de manipulare a excepțiilor va folosi numele predefinit
  - Exemple:
    - NO\_DATA\_FOUND
    - TOO\_MANY\_ROWS
    - INVALID\_CURSOR
    - ZERO\_DIVIDE
    - DUP\_VAL\_ON\_INDEX
    - COLLECTION\_IS\_NULL

# Exceptii generate de erori care nu sunt predefinite în serverul Oracle



- Sunt erori standard Oracle
- Pot fi create excepții care vizează aceste erori cu funcția PRAGMA EXCEPTION\_INIT – aceasta indica compilatorului să asocieze un nume de excepție cu un număr de eroare Oracle
  - Este posibilă referirea unei excepții interne prin nume și scrierea unui handler specific acesteia



# Exemplu- captarea erorii cu numărul -01400 'cannot insert NULL'

Set serveroutput on  
DECLARE

```
insert_excep EXCEPTION;  
PRAGMA EXCEPTION_INIT  
insert_excep,-01400 );
```

BEGIN

```
Insert into departments  
(department_id, department_name) values (280,null);
```

EXCEPTION

```
WHEN insert_excep THEN  
DBMS_OUTPUT.put_line ('operatiune esuata');  
DBMS_OUTPUT.put_line (SQLERRM);
```

END;

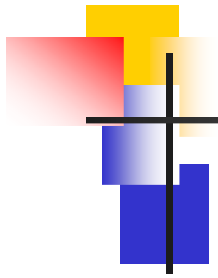


## Funcții utile pentru captarea erorilor

---

- `SQLCODE` – întoarce o valoare numerică pentru codul erorii
- `SQLERRM` – întoarce mesajul asociat cu numărul erorii



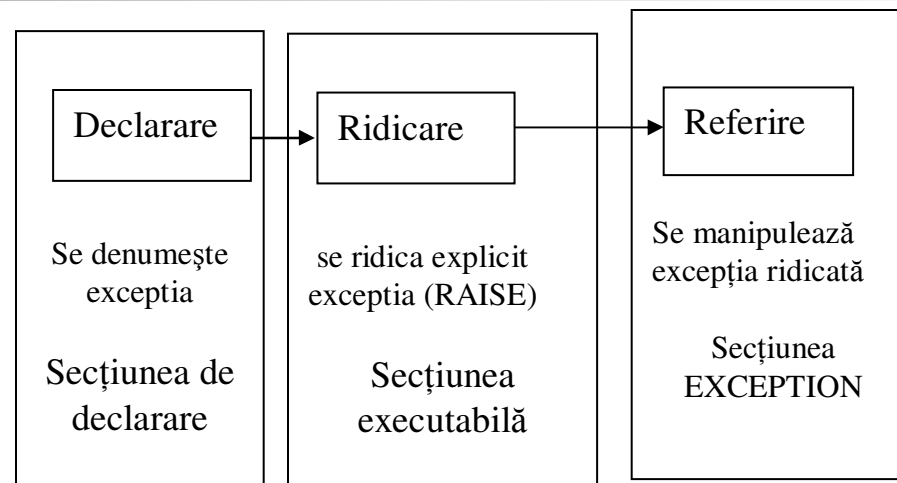


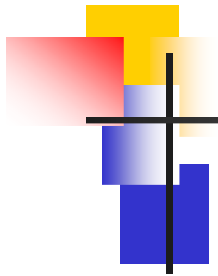
# Exemplu

---

```
DECLARE
    Cod_er NUMBER;
    Mes_er VARCHAR2(255);
BEGIN
    ...
EXCEPTION
    ...
    WHEN OTHERS THEN
        ROLLBACK;
        Cod_er:=SQLCODE;
        Mes_er:= SQLERRM;
        INSERT INTO erori (e_user, e_data, e_cod, e_mesaj)
            values (user, sysdate, cod_er, mes_er);
END;
```

# Capturarea excepțiilor definite de utilizator





# Exemplu

```
DECLARE
    id_invalid EXCEPTION;
    nume VARCHAR2(20):='&nume';
    deptno number :=&deptno;
BEGIN
    UPDATE departments
    SET department_name = nume
    WHERE department_id=deptno;
    IF SQL%NOTFOUND THEN
        RAISE id_invalid;
    END IF;
    COMMIT;
EXCEPTION
    WHEN id_invalid THEN
        DBMS_OUTPUT.PUT_LINE (' identificator invalid');
END;
```