

Laborator 3:

STRUCTURI FUNDAMENTALE DE PROGRAMARE

1. STRUCTURI ALTERNATIVE

Structura **IF..THEN..END IF**

```
IF cond1 THEN  
    Secventa instr1  
ELSE  
    Secventa instr2  
END IF;
```

```
IF conditie1 THEN  
    Secventa instr1  
ELSE  
    IF conditie2 THEN  
        Secventa instr2  
    END IF;
```

Se poate folosi clauza **ELSIF** în loc de **IF** imbricate

```
IF conditie1 THEN  
    Secventa instr1  
ELSIF conditie2 THEN Secventa instr2;  
ELSIF conditie3 THEN Secventa instr3;  
-----  
ELSIF conditieN THEN Secventa instrn;
```

Exemplu:

În funcție de codul departamentului (valorile posibile sunt 10, 20 și 30) să se afișeze numele și salariul angajatului cu cel mai mare salariu din departament.

```
DECLARE
  v_name emp.ename%type;
  v_sal emp.sal%type;
  v_deptno emp.deptno%type;
BEGIN
  SELECT ename, sal INTO v_name, v_sal
  FROM emp
  WHERE sal =
    ( SELECT MAX(sal)
      FROM emp
      WHERE deptno = 10);
  v_deptno := 10;
  DBMS_OUTPUT.PUT_LINE('Departamentul = '||v_deptno);
  IF v_deptno = 10 THEN
    DBMS_OUTPUT.PUT_LINE('nume = '||v_name||' salariu = '||v_sal);
  END IF;

  SELECT ename, sal INTO v_name, v_sal
  FROM emp
  WHERE sal =
    ( SELECT MAX(sal)
      FROM emp
      WHERE deptno = 20);
  v_deptno := 20;
  DBMS_OUTPUT.PUT_LINE('Departamentul = '||v_deptno);
  IF v_deptno = 20 THEN
    DBMS_OUTPUT.PUT_LINE('nume = '||v_name||' salariu = '||v_sal);
  END IF;

  SELECT ename, sal INTO v_name, v_sal
  FROM emp
  WHERE sal =
    ( SELECT MAX(sal)
      FROM emp
      WHERE deptno = 30);
  v_deptno := 30;
  DBMS_OUTPUT.PUT_LINE('Departamentul = '||v_deptno);
  IF v_deptno = 30 THEN
    DBMS_OUTPUT.PUT_LINE('nume = '||v_name||' salariu = '||v_sal);
  END IF;
END;
```

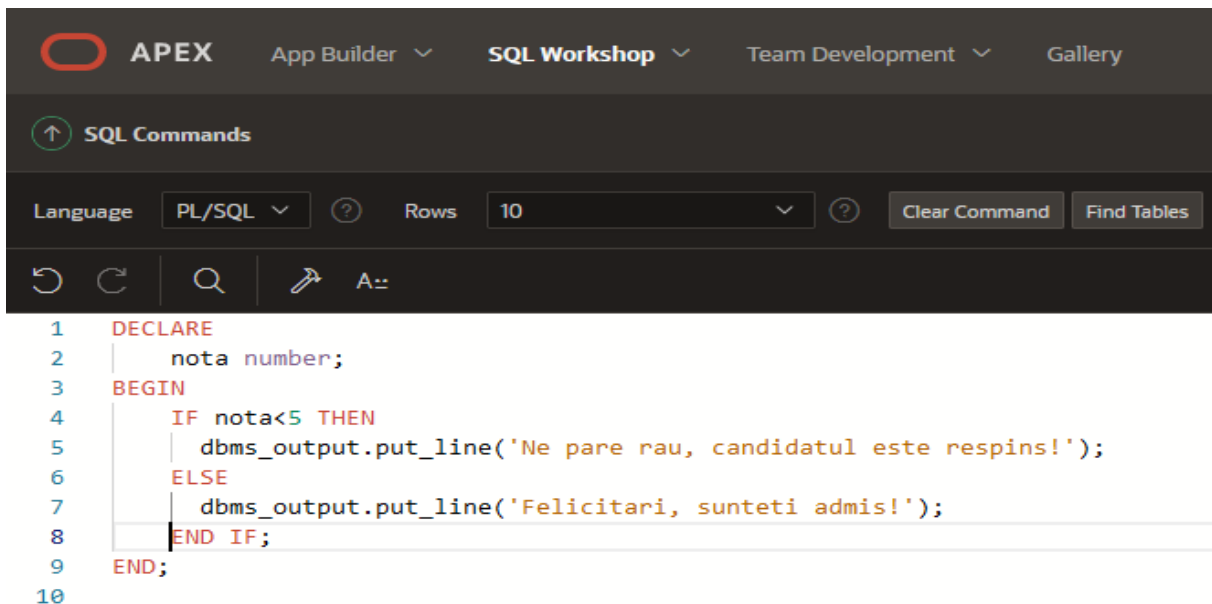
Executia codului anterior:

```
Results Explain Describe Saved SQL
Departamentul = 10
nume = KING salariu = 5000
Departamentul = 20
nume = FORD salariu = 3900
Departamentul = 30
nume = BLAKE salariu = 2850
```

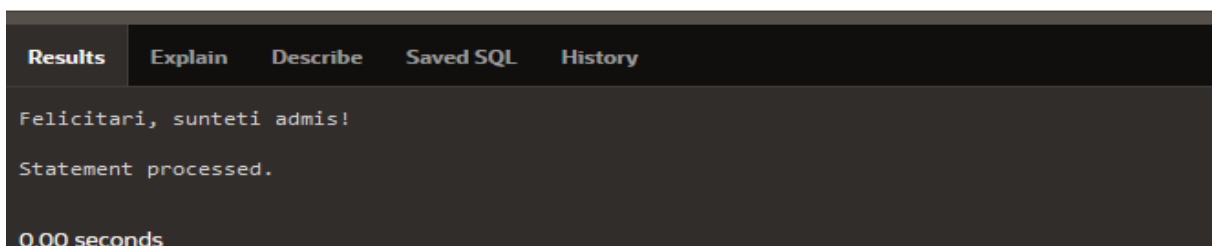
Atentie la variabilele de tip NULL si evaluarea in IF!

De exemplu, în următoarea situație se va afișa “Felicitări, sunteți admis!” din cauza faptului că variabila nota este declarată, dar nu este inițializată, fiind deci NULL:

```
DECLARE
    nota number;
BEGIN
    IF nota<5 THEN
        dbms_output.put_line('Ne pare rau, candidatul este respins!');
    ELSE
        dbms_output.put_line('Felicitari, sunteti admis!');
    END IF;
END;
```



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, there is a 'SQL Commands' section with a search icon. The 'Language' is set to 'PL/SQL', and 'Rows' is set to '10'. There are buttons for 'Clear Command' and 'Find Tables'. The main area displays the PL/SQL code from the previous block, with line numbers 1 through 10. The code is: 1 DECLARE, 2 | nota number;, 3 BEGIN, 4 | IF nota<5 THEN, 5 | | dbms_output.put_line('Ne pare rau, candidatul este respins!');, 6 | ELSE, 7 | | dbms_output.put_line('Felicitari, sunteti admis!');, 8 | END IF;, 9 END;, 10



The screenshot shows the 'Results' tab of the APEX SQL Workshop. The output is: Felicitari, sunteti admis! Statement processed. 0.00 seconds

Observați cazurile de mai jos:

```
X:=10;  
Y:=NULL;  
IF x!=y then  
--intoarce NULL si nu TRUE  
END IF;  
sau  
a:=NULL;  
b:=NULL;  
IF a=b then  
--intoarce NULL si nu TRUE  
END IF;
```

Structura **CASE ... WHEN... THEN...**

Sunt 2 variante:

1. **expresii CASE (CASE Expressions)** care întorc un rezultat într-o variabilă. Se termină cu **END**
2. **sintaxa CASE (CASE Statement)** care execută o anumită instrucțiune. Se termină cu **END CASE**, iar fiecare rând se termină cu ;

1. **CASE Expressions:**

```
Variabila:=  
CASE [Selector]  
WHEN expression1 THEN result1  
WHEN expression2 THEN result2  
-----  
WHEN expressionN THEN resultN  
[ELSE result N+1]  
END;
```

2. **CASE Statement:**

```
CASE [Selector]  
WHEN expression1 THEN action1;  
WHEN expression2 THEN action2;  
-----  
WHEN expressionN THEN actionN;  
[ELSE action N+1];  
END CASE;
```

2. STRUCTURI REPETITIVE

Structura LOOP.....END LOOP

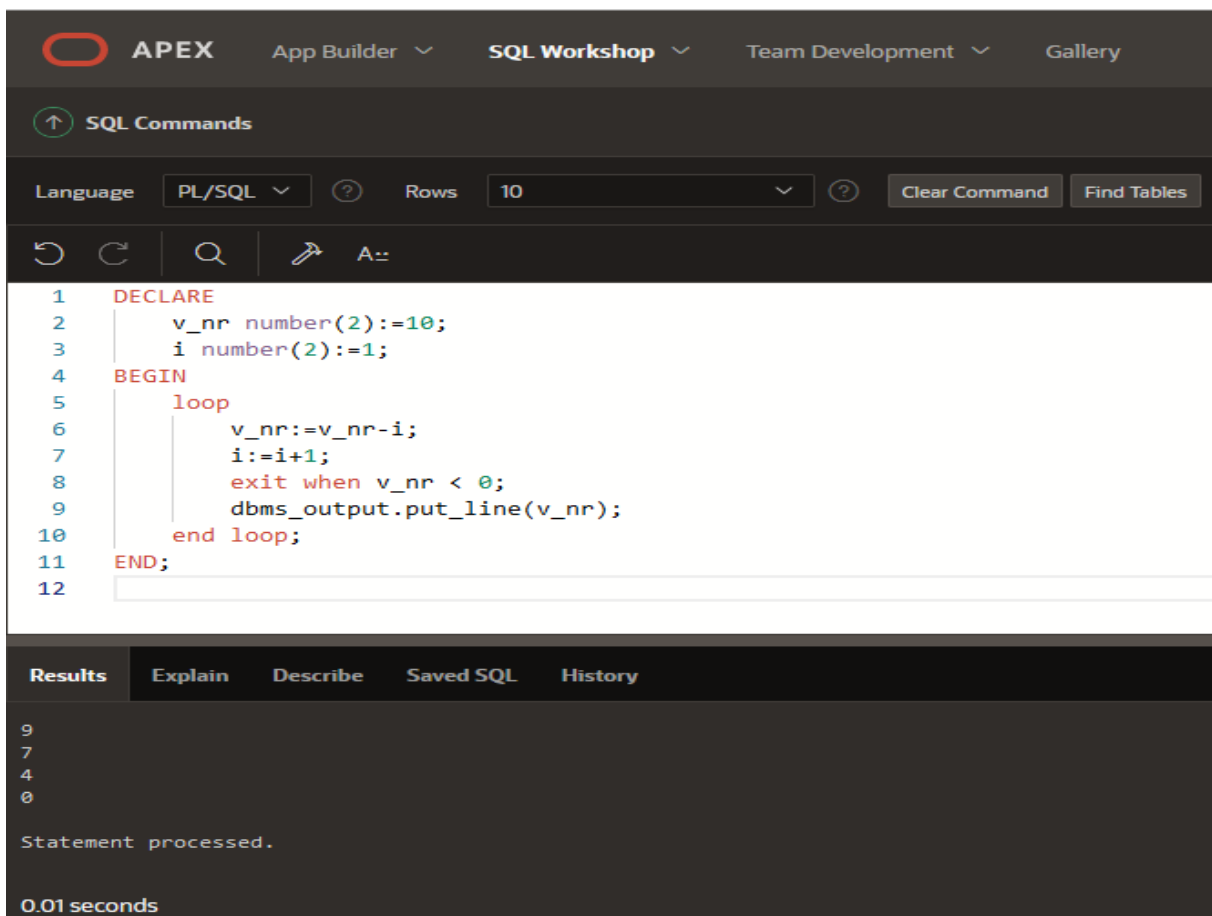
LOOP

Secventa comenzi;
EXIT [WHEN cond];
END LOOP;

Exemplu:

Se afișează pe ecran utilizând structura LOOP...END LOOP numerele 9,7, 4, 0.

```
DECLARE
    v_nr number(2):=10;
    i number(2):=1;
BEGIN
    loop
        v_nr:=v_nr-i;
        i:=i+1;
        exit when v_nr < 0;
        dbms_output.put_line(v_nr);
    end loop;
END;
```



The screenshot displays the Oracle APEX SQL Workshop interface. At the top, the navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section shows the PL/SQL code being executed. The code is as follows:

```
1 DECLARE
2     v_nr number(2):=10;
3     i number(2):=1;
4 BEGIN
5     loop
6         v_nr:=v_nr-i;
7         i:=i+1;
8         exit when v_nr < 0;
9         dbms_output.put_line(v_nr);
10    end loop;
11 END;
```

The 'Results' tab at the bottom shows the output of the execution: 9, 7, 4, and 0. Below the output, it states 'Statement processed.' and '0.01 seconds'.

Se afișează în ordine angajații care au salariul mai mic decât salariul mediu din firma.

```
DECLARE
  v_sal emp.sal%type;
  v_salmediu emp.sal%type;
BEGIN
  SELECT avg(sal)
  INTO v_salmediu
  FROM emp;
  dbms_output.put_line('Salariul mediu este: '||v_salmediu);

  FOR c IN ( SELECT EMPNO, ENAME, SAL
            FROM emp
            WHERE deptno = 10 )
  LOOP
    IF(v_salmediu < c.sal) THEN
      DBMS_OUTPUT.PUT_LINE ('Salariul pentru angajatul ' || c.ename || '
este: ' || c.sal);
    END IF;
  END LOOP;
END;
```

The screenshot shows the Oracle SQL Workshop interface. At the top, there are navigation tabs for APEX, App Builder, SQL Workshop, Team Development, and Gallery. Below the tabs, there is a section for SQL Commands with a search icon and a dropdown menu. The main area displays the SQL script from the previous block, with line numbers 1 through 19. The script is executed, and the results are shown in a table below. The results table has four columns: Results, Explain, Describe, Saved SQL, and History. The output of the script is displayed in the Results column.

```
1 DECLARE
2   v_sal emp.sal%type;
3   v_salmediu emp.sal%type;
4 BEGIN
5   SELECT avg(sal)
6   INTO v_salmediu
7   FROM emp;
8   dbms_output.put_line('Salariul mediu este: '||v_salmediu);
9
10  FOR c IN ( SELECT EMPNO, ENAME, SAL
11            FROM emp
12            WHERE deptno = 10 )
13  LOOP
14    IF(v_salmediu < c.sal) THEN
15      DBMS_OUTPUT.PUT_LINE ('Salariul pentru angajatul ' || c.ename || '
16este: ' || c.sal);
17    END IF;
18  END LOOP;
19 END;
```

Results	Explain	Describe	Saved SQL	History
Salariul mediu este: 2180.18 Salariul pentru angajatul KING este: 5000 Salariul pentru angajatul CLARK este: 2450				

Statement processed.

Structura WHILE.....LOOP....END LOOP

```
WHILE conditie LOOP  
  Secventa comenzi 1;  
  Secventa comenzi 2;  
  EXIT [WHEN conditie];  
END LOOP;
```

Exemplu:

Se afișează pe ecran utilizând structura **WHILE LOOP...END LOOP** numerele 9,7, 4, 0.

```
DECLARE  
  v_nr number(2):=10;  
  i number(2):=1;  
BEGIN  
  while v_nr > 0 loop  
    v_nr:=v_nr-i;  
    i:=i+1;  
    dbms_output.put_line(v_nr);  
  end loop;  
END;
```

The screenshot displays the Oracle APEX SQL Workshop interface. At the top, the navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section is active, showing the language set to 'PL/SQL' and 'Rows' set to '10'. The SQL editor contains the following code:

```
1 DECLARE  
2   v_nr number(2):=10;  
3   i number(2):=1;  
4 BEGIN  
5   while v_nr > 0 loop  
6     v_nr:=v_nr-i;  
7     i:=i+1;  
8     dbms_output.put_line(v_nr);  
9   end loop;  
10 END;  
11
```

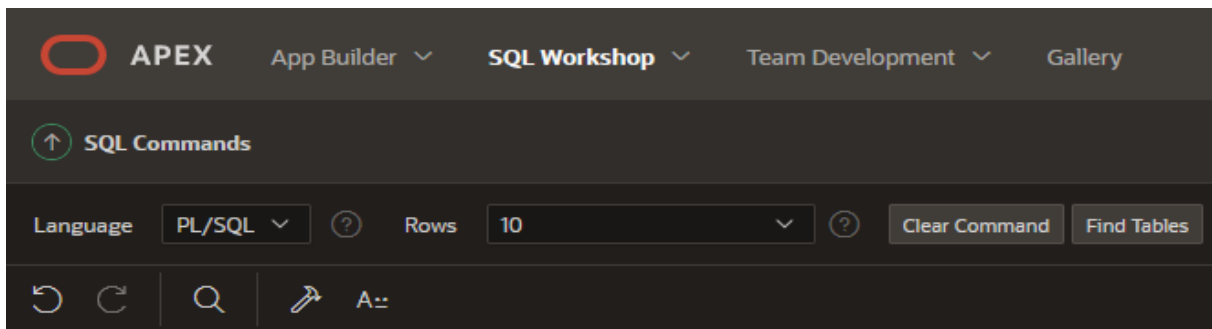
The 'Results' tab is selected, showing the output of the query:

```
9  
7  
4  
0
```

Below the results, the status 'Statement processed.' and the execution time '0.01 seconds' are displayed.

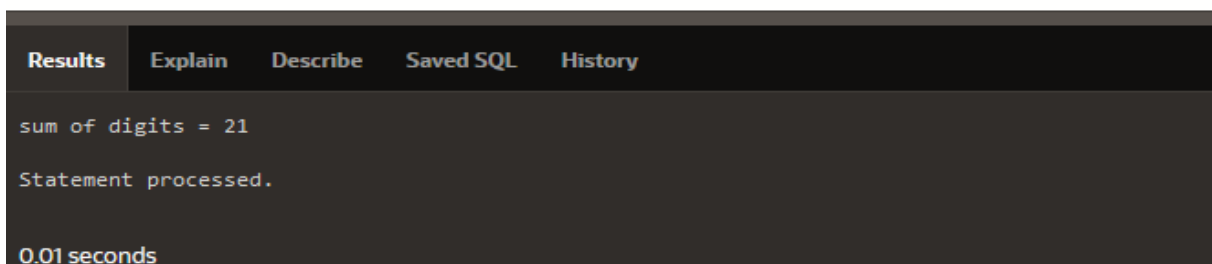
Sa se afiseze suma cifrelor unui numar intreg.

```
DECLARE
  n INTEGER;
  temp_sum INTEGER;
  r   INTEGER;
BEGIN
  n := 123456;
  temp_sum := 0;
  WHILE n <> 0 LOOP
    r := MOD(n, 10);
    temp_sum := temp_sum + r;
    n := Trunc(n / 10);
  END LOOP;
  dbms_output.Put_line('sum of digits = ' || temp_sum);
END;
```



The screenshot shows the APEX SQL Workshop interface. At the top, there are navigation tabs for 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this is a 'SQL Commands' section with a search icon and the text 'SQL Commands'. There are controls for 'Language' (set to 'PL/SQL'), 'Rows' (set to '10'), and buttons for 'Clear Command' and 'Find Tables'. Below these controls is a toolbar with icons for undo, redo, search, and a command prompt 'A:'. The main area displays the PL/SQL code from the previous block, with line numbers 1 through 16 on the left.

```
1 DECLARE
2     n   INTEGER;
3     temp_sum INTEGER;
4     r   INTEGER;
5 BEGIN
6     n := 123456;
7     temp_sum := 0;
8
9     WHILE n <> 0 LOOP
10        r := MOD(n, 10);
11        temp_sum := temp_sum + r;
12        n := Trunc(n / 10);
13    END LOOP;
14    dbms_output.Put_line('sum of digits = ' || temp_sum);
15 END;
16
```



The screenshot shows the 'Results' tab of the APEX SQL Workshop. It has sub-tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The main area displays the output of the SQL command: 'sum of digits = 21'. Below this, it says 'Statement processed.' and at the bottom, '0.01 seconds'.

Structura FOR.....LOOP....END LOOP

```
FOR var IN [REVERSE] valmin..valmax LOOP
  Secventa comenzi;
  EXIT [WHEN conditie];
END LOOP;
```

Observatii:

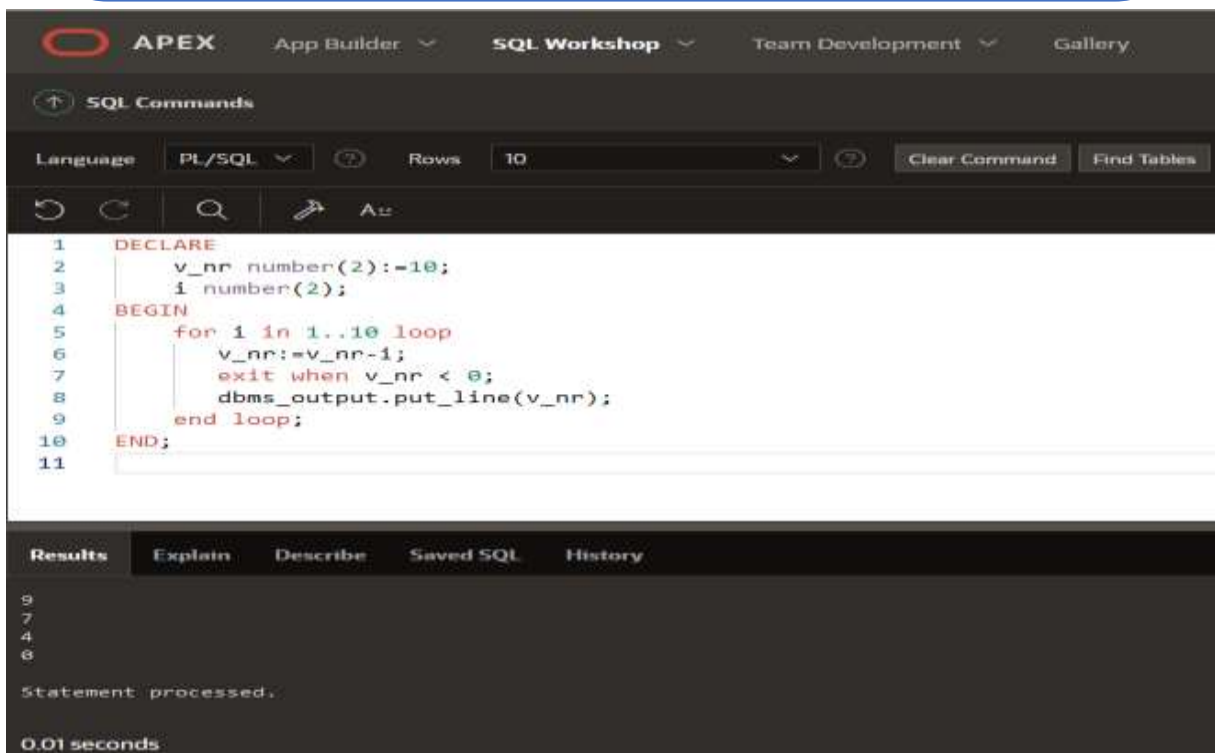
Valorile intervalului pot fi de orice tip, dar sa aiba valori care pot fi convertite la un intreg (de exemplu 20/13 sau 11/5). Daca aceste 2 valori vor fi egale ca intregi atunci instructiunile din interiorul ciclului se executa o singura data. De exemplu secventa:

```
FOR i IN 3..3 LOOP
  Secventa comenzi;
END LOOP;
```

Exemple:

Se afișează pe ecran utilizând structura FOR...END LOOP numerele 9,7, 4, 0.

```
DECLARE
  v_nr number(2):=10;
  i number(2);
BEGIN
  for i in 1..10 loop
    v_nr:=v_nr-i;
    exit when v_nr < 0;
    dbms_output.put_line(v_nr);
  end loop;
END;
```



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, the 'SQL Commands' section is active, showing a PL/SQL script. The script is as follows:

```
1 DECLARE
2   v_nr number(2):=10;
3   i number(2);
4 BEGIN
5   for i in 1..10 loop
6     v_nr:=v_nr-i;
7     exit when v_nr < 0;
8     dbms_output.put_line(v_nr);
9   end loop;
10 END;
```

The 'Results' tab is selected, displaying the output of the script: 9, 7, 4, 0. Below the output, it states 'Statement processed.' and '0.01 seconds'.

Se afișează în ordine angajații cu codurile în intervalul 100-110 atât timp cât salariul acestora este mai mic decât media:

```
DECLARE
  v_sal angajati.salariul%type;
  v_salMediu v_sal%type;
  -- i nu mai trebuie declarat
BEGIN
  SELECT avg(salariul) into v_salmediu from angajati;
  dbms_output.put_line('Salariul mediu este: '||v_salmediu);
  for i in 100..110 loop
    select salariul into v_sal from angajati where id_angajat=i;
    dbms_output.put_line('Salariatul cu codul '||i||' are salariul: '||v_sal);
    exit when v_sal<v_salmediu;
  end loop;
end;
```

Structuri LOOP imbricate (se vor eticheta loop-urile)

```
BEGIN
  <<LOOP_EXTERN>>
  LOOP
    v_var:=v_var+1;
    EXIT WHEN v_var>10;
    <<LOOP_INTERN>>
    LOOP
      .....
      EXIT LOOP_EXTERN WHEN conditie1;
      EXIT WHEN conditie2;
      .....
    END LOOP;
    .....
  END LOOP;
END;
```

Probleme propuse spre rezolvare

1. Specificați ce se va afișa la rularea următorului bloc PL/SQL:

a) Situația în care identificatorul unei variabile este inclus între ghilimele duble și referința către variabila este, de asemenea, inclusă între ghilimele duble.

```
DECLARE
  "WELCOME" varchar2(10) := 'welcome'; -- denumire de variabila cu ghilimele duble
BEGIN
  DBMS_Output.Put_Line("Welcome"); -- utilizarea variabilei respective
END;
```

```
Language: PL/SQL Rows: 20 Clear Command Find Tables
1 DECLARE
2   "WELCOME" varchar2(10) := 'welcome'; -- denumire de variabila cu ghilimele duble
3 BEGIN
4   DBMS_Output.Put_Line('Welcome'); -- utilizarea variabilei respective
5 END;
6
7

Results Explain Describe Save SQL History
Error at line 4/29: ORA-00512: line 4, column 25:
PL/SQL: Identifier 'welcome' must be declared
ORA-00512: at 'SYS.WMV_DBMS_SQL_APEX_320100', line 826
ORA-00512: line 4, column 3:
PL/SQL: Statement ignored
ORA-00512: at 'SYS.DBMS_SYS_SQL', line 3438
ORA-00512: at 'SYS.WMV_DBMS_SQL_APEX_320100', line 811
ORA-00512: at 'APEX_320100.WMV_FLOW_DYNAMIC_EXEC', line 2040

2. 'WELCOME' varchar2(10) := 'welcome'; -- denumire de variabila cu ghilimele duble
3. BEGIN
4. DBMS_Output.Put_Line('Welcome'); -- utilizarea variabilei respective
5. END;
```

b) Situația în care identificatorul unei variabile nu este inclus între ghilimele duble și referința către variabila este, de asemenea, inclusă între ghilimele duble.

```
DECLARE
  WELCOME varchar2(10) := 'welcome'; -- denumire de variabila fara ghilimele duble
BEGIN
  DBMS_Output.Put_Line("Welcome"); -- utilizarea variabilei respective, dar cu ghilimele
END;
```

```
Language: PL/SQL Rows: 20 Clear Command Find Tables
1 DECLARE
2 WELCOME varchar2(10) := 'welcome'; -- denumire de variabila fara ghilimele duble
3 BEGIN
4 DBMS_Output.Put_Line("Welcome"); -- utilizarea variabilei respective, dar cu ghilimele
5 END;
6
7
```

Results Explain Describe Saved SQL History

```
Error at line 4/25: ORA-06550: line 4, column 25
PLS-00201: Identifier 'welcome' must be declared
ORA-06552: at "SYS.MAN_DBMS_SQL_APEX_220200", line 426
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
ORA-06552: at "SYS.DBMS_SVS_SQL", line 1658
ORA-06552: at "SYS.MAN_DBMS_SQL_APEX_220200", line 411
ORA-06552: at "APEX_220200.MAN_FLOW_DYNAMIC_EXEC", line 2046

2. WELCOME varchar2(10) := 'welcome'; -- denumire de variabila fara ghilimele
duble
3. BEGIN
4. DBMS_Output.Put_Line("Welcome"); -- utilizarea variabilei respective, dar cu
ghilimele
5. END;
```

0.00 seconds

c) Situația în care identificatorul unei variabile nu este inclus între ghilimele duble, la fel și referința către variabila nu este inclusă între ghilimele duble sau simple.

DECLARE

WELCOME varchar2(10) := 'welcome'; -- denumire de variabila fara ghilimele duble sau simple

BEGIN

DBMS_Output.Put_Line(Welcome); -- utilizarea variabilei respective, dar fara ghilimele simple sau duble

END;

```
SQL Commands
Language: PL/SQL Rows: 20 Clear Command Find Tables
1 DECLARE
2 WELCOME varchar2(10) := 'welcome'; -- denumire de variabila fara ghilimele duble sau simple
3 BEGIN
4 DBMS_Output.Put_Line(Welcome); -- utilizarea variabilei respective, dar fara ghilimele simple sau duble
5 END;
6
7
```

Results Explain Describe Saved SQL History

```
welcome
Statement processed.
```

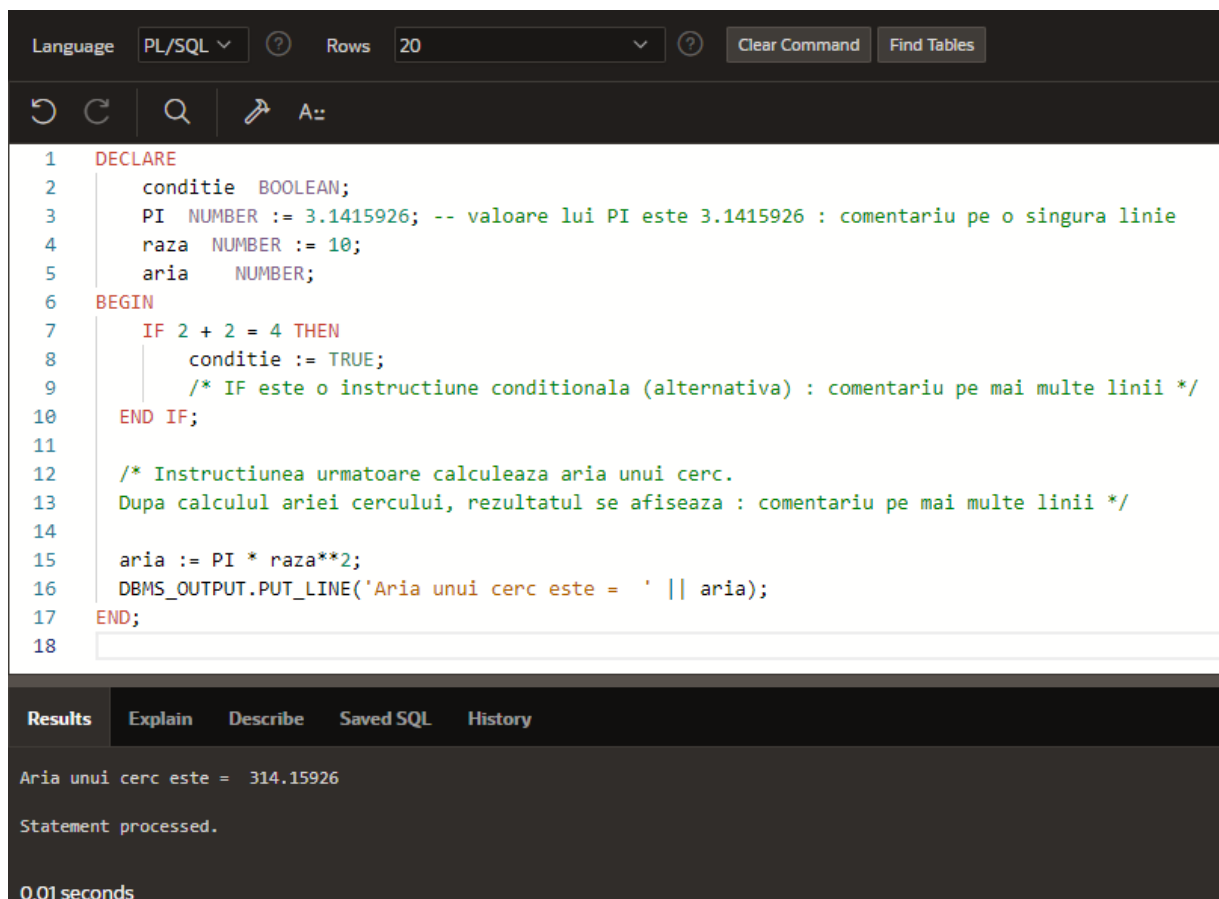
0.01 seconds

2) Exemplu de utilizare a comentariilor pe o singura linie si pe mai multe linii.

```
DECLARE
  conditie BOOLEAN;
  PI NUMBER := 3.1415926; -- valoarea lui PI este 3.1415926 : comentariu pe o singura linie
  raza NUMBER := 10;
  aria NUMBER;
BEGIN
  IF 2 + 2 = 4 THEN
    conditie := TRUE;
    /* IF este o instructiune conditionala (alternativa) : comentariu pe mai multe linii */
  END IF;

  /* Instructiunea urmatoare calculeaza aria unui cerc.
  Dupa calculul ariei cercului, rezultatul se afiseaza : comentariu pe mai multe linii */

  aria := PI * raza**2;
  DBMS_OUTPUT.PUT_LINE('Aria unui cerc este = ' || aria);
END;
```



```
Language PL/SQL ? Rows 20 ? Clear Command Find Tables
1 DECLARE
2   conditie BOOLEAN;
3   PI NUMBER := 3.1415926; -- valoare lui PI este 3.1415926 : comentariu pe o singura linie
4   raza NUMBER := 10;
5   aria NUMBER;
6 BEGIN
7   IF 2 + 2 = 4 THEN
8     conditie := TRUE;
9     /* IF este o instructiune conditionala (alternativa) : comentariu pe mai multe linii */
10  END IF;
11
12  /* Instructiunea urmatoare calculeaza aria unui cerc.
13  Dupa calculul ariei cercului, rezultatul se afiseaza : comentariu pe mai multe linii */
14
15  aria := PI * raza**2;
16  DBMS_OUTPUT.PUT_LINE('Aria unui cerc este = ' || aria);
17 END;
18
```

Results Explain Describe Saved SQL History

Aria unui cerc este = 314.15926

Statement processed.

0.01 seconds

3) Exemplu de utilizare a parantezelor si evaluarea expresiilor in functie de precedenta operatorilor.

4) Exemplu de secvența PL/SQL pentru a descrie utilizarea valorilor NULL în comparația egală, comparația inegală și comparația NOT NULL egal NULL.

```
DECLARE
  m NUMBER := 21;
  n NUMBER := NULL;
  o NUMBER := NULL;
  p NUMBER := NULL;
  q INTEGER := 89;
  r INTEGER := 45;
  large INTEGER;
BEGIN
  IF m != n THEN -- conditia se evalueaza la NULL, nu la TRUE
    DBMS_OUTPUT.PUT_LINE('m != n'); -- nu se executa
  ELSIF m = n THEN -- la fel, conditia se evalueaza la valoarea NULL
    DBMS_OUTPUT.PUT_LINE('m = n');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Nu se poate spune daca m si n sunt sau nu egale');
  END IF;
  -----
  IF o = p THEN -- conditia se evalueaza la NULL, nu la TRUE
    DBMS_OUTPUT.PUT_LINE('o = p'); -- nu se executa
  ELSIF o != p THEN -- la fel, conditia se evalueaza la valoarea NULL
    DBMS_OUTPUT.PUT_LINE('o != p'); -- nu se executa
  ELSE
    DBMS_OUTPUT.PUT_LINE('Nu se poate spune daca cele doua valori de tip NULL sunt
egale');
  END IF;
  -----
  IF (q > r) -- daca q sau r sunt NULL, atunci (q > r) este NULL
    THEN large := q; -- se executa daca (q > r) este TRUE
  ELSE large := r; -- se executa daca (q > r) este FALSE sau NULL
    DBMS_OUTPUT.PUT_LINE('Valoarea cea mai mare = '||large);
  END IF;
  -----
  IF NOT (q > r) -- Daca q sau r este NULL, atunci NOT (q > r) este NULL
    THEN large := r; -- se executa daca NOT (q > r) este TRUE
  ELSE large := q; -- se executa daca NOT (q > r) este FALSE sau NULL
    DBMS_OUTPUT.PUT_LINE('Valoarea cea mai mare = '||large);
  END IF;
END;
```

Language **PL/SQL** ? Rows **20** ? **Clear Command** **Find Tables**

↶ ↷ 🔍 📌 A::

```

1 DECLARE
2     m NUMBER := 21;
3     n NUMBER := NULL;
4     o NUMBER := NULL;
5     p NUMBER := NULL;
6     q INTEGER := 89;
7     r INTEGER := 45;
8     large INTEGER;
9 BEGIN
10    IF m != n THEN -- conditia se evalueaza la NULL, nu la TRUE
11        | DBMS_OUTPUT.PUT_LINE('m != n'); -- nu se executa
12    ELSIF m = n THEN -- la fel, conditia se evalueaza la valoarea NULL
13        | DBMS_OUTPUT.PUT_LINE('m = n');
14    ELSE
15        | DBMS_OUTPUT.PUT_LINE('Nu se poate spune daca m si n sunt sau nu egale');
16    END IF;
17
18    -----
19    IF o = p THEN -- conditia se evalueaza la NULL, nu la TRUE
20        | DBMS_OUTPUT.PUT_LINE('o = p'); -- nu se executa
21    ELSIF o != p THEN -- la fel, conditia se evalueaza la valoarea NULL
22        | DBMS_OUTPUT.PUT_LINE('o != p'); -- nu se executa
23    ELSE
24        | DBMS_OUTPUT.PUT_LINE('Nu se poate spune daca cele doua valori de tip NULL sunt egale');
25    END IF;
26
27    -----
28    IF (q > r) -- daca q sau r sunt NULL, atunci daca (q > r) este NULL
29        | THEN large := q; -- se executa daca (q > r) este TRUE
30    ELSE large := r; -- se executa daca (q > r) este FALSE sau NULL
31        | DBMS_OUTPUT.PUT_LINE('Valoarea cea mai mare = '||large);
32    END IF;
33
34    -----
35    IF NOT (q > r) -- Daca q sau r este NULL, atunci NOT (q > r) este NULL
36        | THEN large := r; -- se executa daca NOT (q > r) este TRUE
37    ELSE large := q; -- se executa daca NOT (q > r) este FALSE sau NULL
38        | DBMS_OUTPUT.PUT_LINE('Valoarea cea mai mare = '||large);
39    END IF;

```

Results Explain Describe Saved SQL History

```

Nu se poate spune daca m si n sunt sau nu egale
Nu se poate spune daca cele doua valori de tip NULL sunt egale
Valoarea cea mai mare = 89

```

5) Scrieți un program PL/SQL pentru a număra numărul de angajați din departamentul 20 și pentru a verifica dacă acest departament are sau nu posturi vacante. Există 6 de posturi vacante în acest departament.

```

DECLARE
    total_emp NUMBER;
BEGIN
    SELECT Count(*) INTO total_emp
    FROM emp e join dept d
    ON e.deptno = d.deptno
    WHERE e.deptno = 20;

```



```

DBMS_OUTPUT.PUT_LINE('Numarul de angajati din departamentul 20 =
'||To_char(total_emp));
IF total_emp >= 6 THEN
    DBMS_OUTPUT.PUT_LINE('Nu sunt posturi libere in departamantul 20.');
```

```

ELSE
    DBMS_OUTPUT.PUT_LINE('Sunt posturi libere in departamentul 20.');
```

```

END IF;
END;
```

The screenshot shows the Oracle SQL Developer interface. At the top, the 'Language' is set to 'PL/SQL' and 'Rows' is set to '20'. The main area contains a PL/SQL script with the following code:

```

1 DECLARE
2     total_emp NUMBER;
3 BEGIN
4     SELECT Count(*) INTO total_emp
5     FROM emp e join dept d
6     ON e.deptno = d.deptno
7     WHERE e.deptno = 20;
8     DBMS_OUTPUT.PUT_LINE('Numarul de angajati din departamentul 20 = '||TO_CHAR(total_emp));
9     IF total_emp >= 6 THEN
10        DBMS_OUTPUT.PUT_LINE('Nu sunt posturi libere in departamantul 20.');
```

```

11    ELSE
12        DBMS_OUTPUT.PUT_LINE('Sunt posturi libere in departamentul 20.');
```

```

13    END IF;
14 END;
```

Below the script, the 'Results' tab is active, displaying the output of the script:

```

Numarul de angajati din departamentul 20 = 6
Nu sunt posturi libere in departamantul 20.
```

At the bottom of the results window, it says 'Statement processed.' and '0.01 seconds'.

6) Scrieți un program PL/SQL pentru a afișa în ce zi din săptămâna este o anumită dată. Se va utiliza instrucțiunea CASE.

7) Scrieți un program în PL/SQL pentru a afișa primele n numere cu o diferență de 3 și începând de la 1.

Exemplu:

Pentru n = 10 se vor afișa valorile: 1 4 7 10 13 16 19 22 25 28

8) Scrieți un program în PL/SQL pentru a afișa primele 10 numere din șirul lui Fibonacci.

Precizare: numerele din șirul lui Fibonacci sunt: 0, 1, 1, 2, 3, 5, 8,

9) Scrieți un program în PL/SQL pentru a afișa cel mai mare număr între 3 numere întregi.

10) Scrieți un program în PL/SQL pentru a afișa produsul cifrelor pare ale unui număr întreg.

Exemplu:

Dacă numărul este n = 12345, atunci produsul este p = 8 (2 * 4)

Bibliografie web:

<https://www.w3resource.com/>

<https://www.bullraider.com/database/pl-sql/pl-sql-examples>

<https://www.oracletutorial.com/plsql-tutorial/>