

Laborator 3:

STRUCTURI FUNDAMENTALE DE PROGRAMARE

(partea I)

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
<pre>Departamentul = 10 nume = KING salariu = 5000 Departamentul = 20 nume = FORD salariu = 3900 Departamentul = 30 nume = BLAKE salariu = 2850</pre>			

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;
```

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language PL/SQL Rows 10 Clear Command Find Tables

↶ ↷ 🔍 📌 A::

```

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

```

Results Explain Describe Saved SQL History

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 intorc un rezultat intr-o variabila. Se termina cu END
2. **sintaxa CASE (CASE Statement)** care executa o anumita instructiune. Se termina cu END CASE, iar fiecare rand se termina 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:

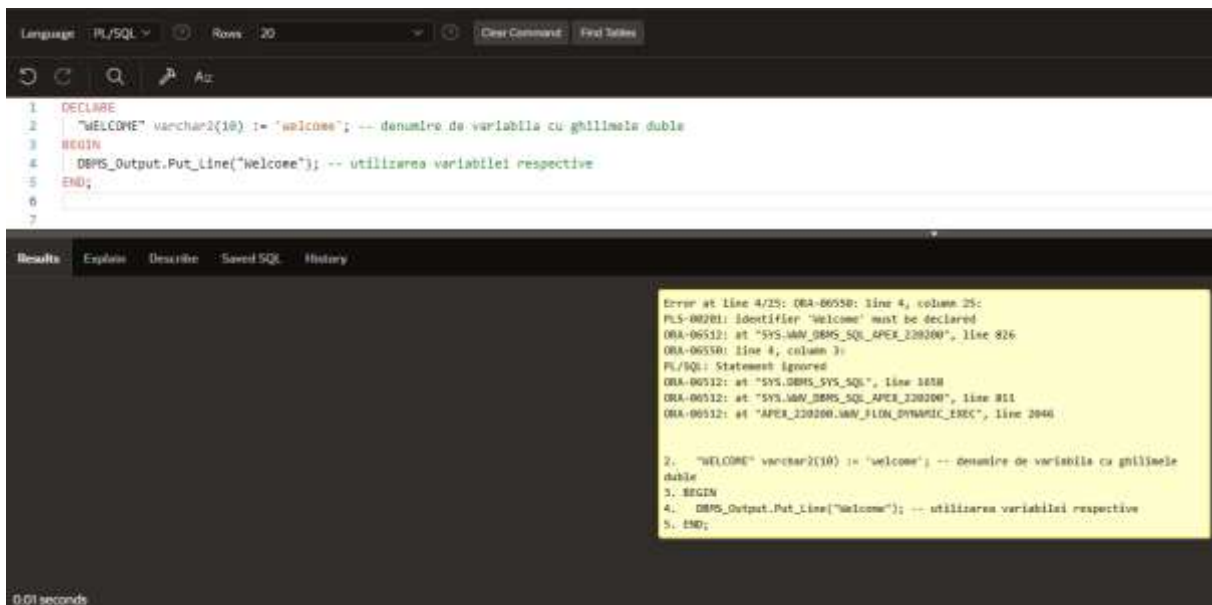
```
CASE [Selector]  
WHEN expression1 THEN action1;  
WHEN expression2 THEN action2;  
-----  
WHEN expressionN THEN actionN;  
[ELSE action N+1];  
END CASE;
```

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;
```



The screenshot shows a SQL IDE interface. The top part displays the PL/SQL code from the previous block. The bottom part shows the execution results, which include an error message:

```
Error at line 4/25: ORA-00510: line 4, column 25:
PLS-00201: Identifier 'welcome' must be declared
ORA-00512: at 'SYS.WW_DBMS_SQL_APEX_120200', line 826
ORA-00510: line 4, column 3:
PL/SQL: Statement ignored
ORA-00512: at 'SYS.WW_DBMS_SQL', line 1058
ORA-00512: at 'SYS.WW_DBMS_SQL_APEX_120200', line 811
ORA-00512: at 'APEX_120200.WW_FLOW_DYNAMIC_EXEC', line 2046
```

Below the error message, the code is repeated, indicating that the error occurred at line 4, column 25.

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

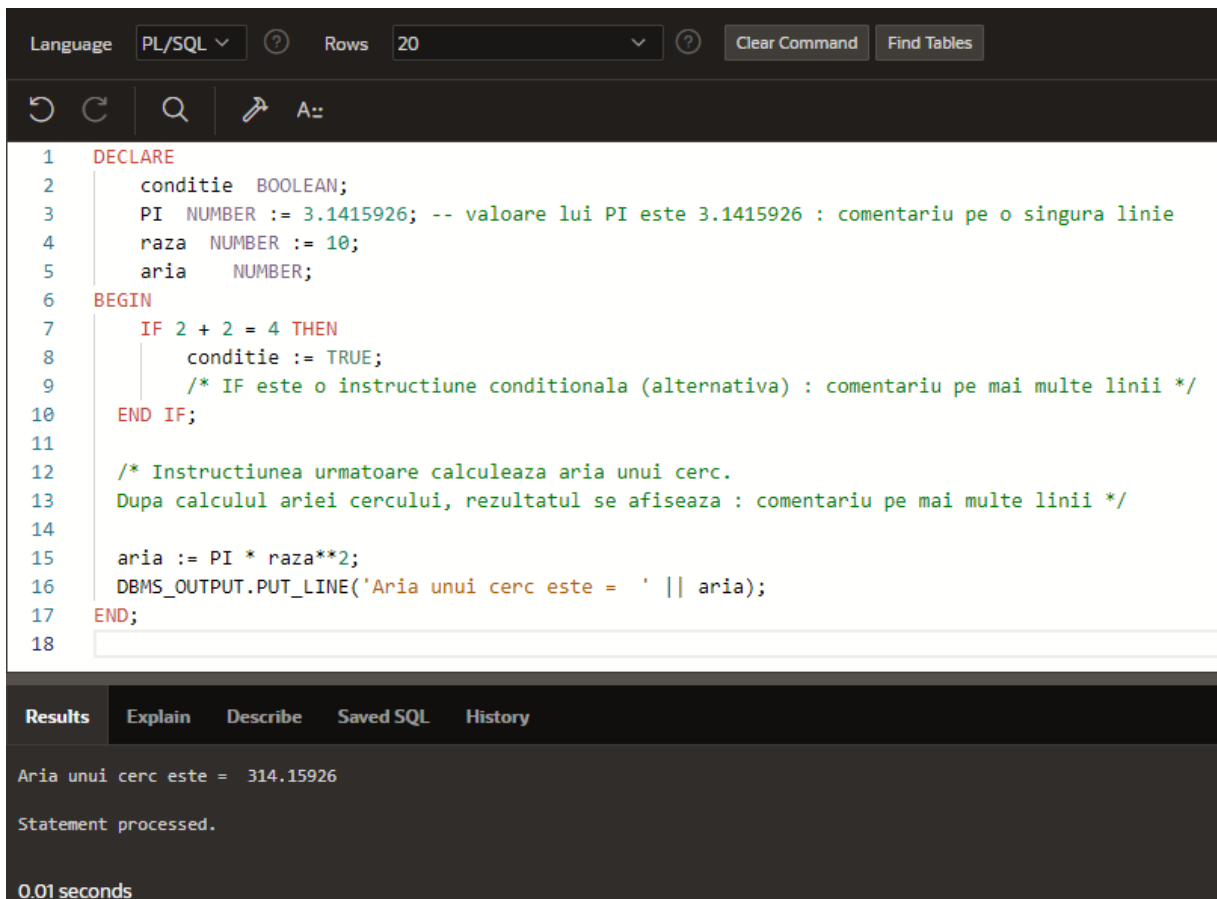
```

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
A:
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    IF o = p THEN -- conditia se evalueaza la NULL, nu la TRUE
19        | DBMS_OUTPUT.PUT_LINE('o = p'); -- nu se executa
20    ELSIF o != p THEN -- la fel, conditia se evalueaza la valoarea NULL
21        | DBMS_OUTPUT.PUT_LINE('o != p'); -- nu se executa
22    ELSE
23        | DBMS_OUTPUT.PUT_LINE('Nu se poate spune daca cele doua valori de tip NULL sunt egale');
24    END IF;
25    -----
26    IF (q > r) -- daca q sau r sunt NULL, atunci daca (q > r) este NULL
27        | THEN large := q; -- se executa daca (q > r) este TRUE
28    ELSE large := r; -- se executa daca (q > r) este FALSE sau NULL
29        | DBMS_OUTPUT.PUT_LINE('Valoarea cea mai mare = '||large);
30    END IF;
31    -----
32    IF NOT (q > r) -- Daca q sau r este NULL, atunci NOT (q > r) este NULL
33        | THEN large := r; -- se executa daca NOT (q > r) este TRUE
34    ELSE large := q; -- se executa daca NOT (q > r) este FALSE sau NULL
35        | DBMS_OUTPUT.PUT_LINE('Valoarea cea mai mare = '||large);
36    END IF;
```

Results [Explain](#) [Describe](#) [Saved SQL](#) [History](#)

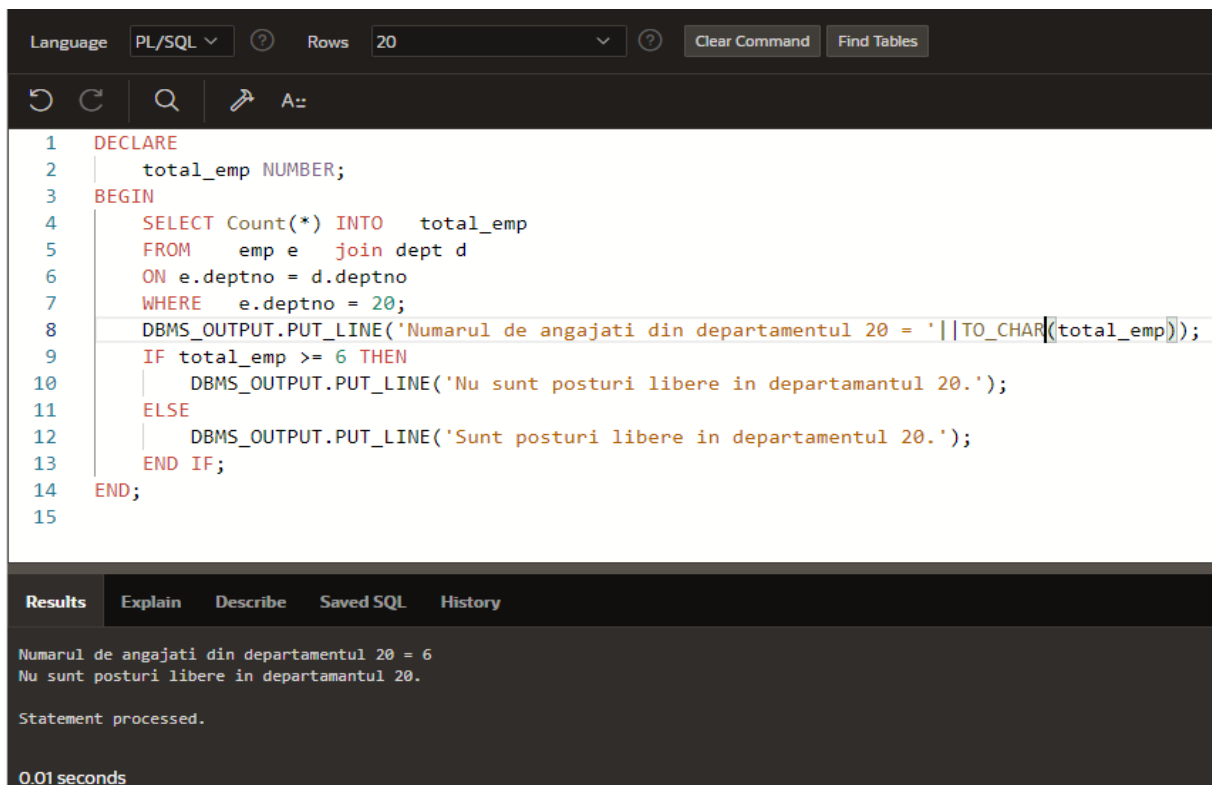
```
Nu se poate spune daca m si n sunt sau nu egale
Nu se poate spune daca cele doua valori de tip NULL sunt egale
Valoarea cea mai mare = 89
```

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 a PL/SQL IDE interface. At the top, there are tabs for 'Language' (set to PL/SQL) and 'Rows' (set to 20). Below the tabs are icons for refresh, search, and a command prompt. The main area displays the following PL/SQL 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 code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing the following output:

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

Statement processed.

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.

Bibliografie web:

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

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

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