Laborator 1: Introducere în PL/SQL

1.1. Caracteristici generale:

- Construcţiile PL/SQL conţin structuri de control procedurale şi comenzi descriptive SQL
- PL/SQL este un limbaj procedural structurat pe bloc, programele putând fi împărţite în blocuri logice
- Blocurile PL/SQL sunt procesate de motorul PL/SQL care poate fi rezident pe ORACLE SERVER sau pe un instrument de dezvoltare (ex.: Oracle Forms, Reports, JDeveloper etc.)
- Multe instrumente ORACLE au propriul motor PL/SQL (ex.: Oracle Forms, Reports, JDeveloper etc.)
- Tipurile de date din SQL pot fi folosite în PL/SQL
- Programarea în PL/SQL este modularizată – se utilizează blocurile care grupează instrucţiunile

1.2. Blocuri PL/SQL:

Orice unitate PL/SQL conţine unul sau mai multe blocuri, complet separate sau imbricate.

Componentele unui bloc PL/SQL:

Un bloc PL/SQL este compus din până la 3 secţiuni: declarativă (opţională), executabilă (obligatorie) şi de tratare a excepţiilor (opţională).

DECLARE *(Opţional)*
variabile, cursori, excepţii
BEGIN *(Obligatoriu)*
comenzi SQL (asigură accesul la baza de date)
structuri de programare procedurală PL/SQL
EXCEPTION *(Opţional)*
acţiuni ce se execută când apare o eroare
END; *(Obligatoriu)*

Observaţii:

- comenzile SQL asigură accesul la baza de date
- operaţiile efectuate cu variabilele PL/SQL în cadrul instrucţiunilor procedurale nu presupun accesarea bazei de date
- se foloseşte (**;**) după fiecare instrucţiune SQL sau instrucţiune de control PL/SQL
- blocul PL/SQL se termină cu (**;**)
- se foloseşte (/) pentru a lansa un bloc anonim în bufferul SQL
- o eroare în PL/SQL este tratată ca o excepţie

Tipuri de blocuri PL/SQL:

Blocuri anonime:

1. Funcţii stocate şi funcţii de aplicaţii
2. Proceduri stocate şi proceduri de aplicaţii
3. Pachete
4. Declanşatoare (triggeri) pe baza de date / de aplicaţii

Blocurile anonime:

- sunt nedenumite
- nu sunt stocate în baza de date
- se declară inline, în locul în care se doreşte execuţia lor
- se execută în momentul rulării

Exemplu:

```
DECLARE
        v_variabila varchar2(5);
BEGIN
        SELECT coloana
        INTO v_variabila
        FROM tabela;
        EXCEPTION
                WHEN excepţie THEN acţiune
END;
/
```

Blocuri anonime imbricate

- se pot imbrica mai multe blocuri;
- acestea se pot eticheta cu <<eticheta_bloc>> , iar variabilele din cadrul blocurilor se pot utiliza astfel: eticheta_bloc.variabila.

```
BEGIN
…….
<< eticheta_bloc >>
DECLARE
        ……..
BEGIN
        ……..
END eticheta_bloc;
END;
/
```

Proceduri, funcţii:

- blocuri PL/SQL cu un nume;
- se pot stoca la nivel de ORACLE SERVER(proceduri/funcţii stocate) sau la nivel de aplicaţie (DEVELOPER – Forms si Reports).

Exemple:

```
CREATE [OR REPLACE] PROCEDURE nume_procedura
IS
…….
BEGIN
…….
[EXCEPTION]
…….
END;
/

CREATE [OR REPLACE] FUNCTION nume_functie
RETURN tip_data
IS
BEGIN
……
RETURN valoare
[EXCEPTION]
……
END;
/
```

Pachete de programe - grupează proceduri, funcţii.

Declanşatori pe baza de date - blocuri PL/SQL asociate tabelelor (de bază sau virtuale) şi lansate automat în execuţie când are loc o comanda de manipulare.

Declanşatori de aplicaţie - blocuri PL/SQL asociate unor evenimente din cadrul aplicaţiei (de exemplu: deplasarea mouse-ului, apăsarea unui buton) şi lansate în execuţie automat.

3. Operatori în PL/SQL

Operator	Caracteristici
+, -, *, /, ** (op. exponenţial)	Operatori aritmetici
AND, OR, NOT	Operatori logici
<, >, =, >=, <=, <>, !=	Operatori de comparaţie
BETWEEN ... AND ...	Operator de verificare a apartenenţei la un interval
IN(listă)	Operator de verificare a apartenenţei la o listă de valori
LIKE	Operator de comparare cu un şablon **%** - oricâte caractere; **_** - un caracter;
IS NULL	Operator care verifică dacă o variabilă are valoarea NULL

||	Operator de concatenare
@	Operator de conectare la distanţă
& **sau &&**	Operatori pentru adresarea variabilelor de substituţie
:=	Operator de atribuire

4. Funcţii SQL disponibile în PL/SQL

- În cadrul instrucţiunilor descriptive sunt suportate toate tipurile de funcţii SQL (inclusiv funcţiile de grup în cadrul instrucţiunii SELECT);
- Instrucţiunile PL/SQL:
  - ✓ Suportă funcţii la nivel de înregistrare (single-row): numerice, caracter, data, de conversie etc.;
  - ✓ **NU** suportă funcţii de grup (SUM, MIN, MAX, AVG, COUNT, STDDEV) sau funcţia DECODE. De exemplu nu se pot utiliza construcţii de forma: *IF DECODE(...) THEN ...* sau *IF AVG(...) THEN ...*

5. Conversii în blocurile PL/SQL

- PL/SQL converteşte tipurile de date dinamic (de exemplu: o valoare numerică la o variabilă char);
- conversii implicite: caracter <-> numeric si caracter <->data;
- conversii explicite: se utilizează funcţiile TO_DATE, TO_NUMBER, TO_CHAR.

6. Comenzi SQL disponibile în PL/SQL

- PL/SQL permite folosirea comenzilor de manipulare a datelor (LMD):
  - ✓ SELECT
  - ✓ INSERT
  - ✓ UPDATE
  - ✓ DELETE
- PL/SQL permite folosirea comenzilor de control al tranzacţiilor:
  - ✓ COMMIT
  - ✓ ROLLBACK
  - ✓ SAVEPOINT

Notă: Un bloc PL/SQL nu este o tranzacţie. Comenzile Commit / Rollback / Savepoint sunt independente de bloc, dar pot să apară în cadrul acestuia.

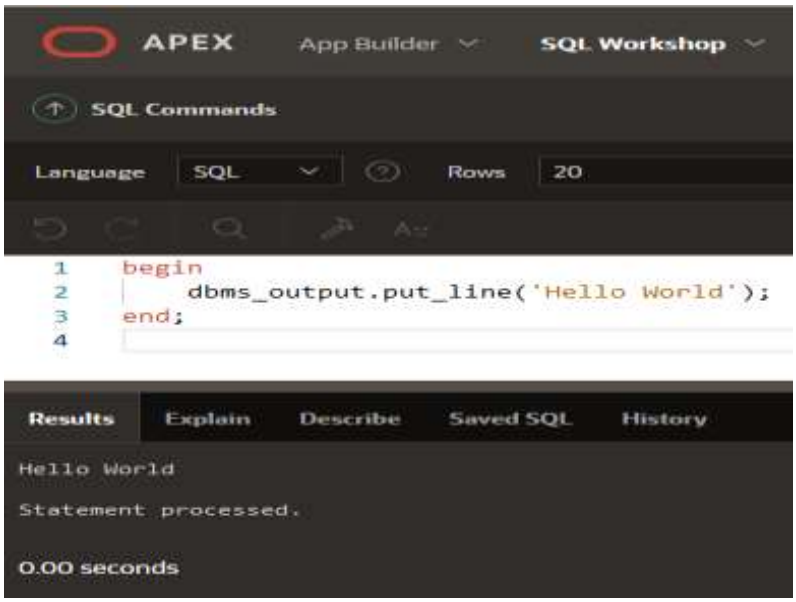
- PL/SQL **NU** acceptă comenzile de definire a datelor (LDD)
  - ✓ CREATE
  - ✓ ALTER
  - ✓ DROP
  - ✓ RENAME
  - ✓ TRUNCATE
- PL/SQL **NU** acceptă comenzile din cadrul limbajului pentru controlul datelor (Data Control Language - DCL)
  - ✓ GRANT
  - ✓ REVOKE

Probleme rezolvate

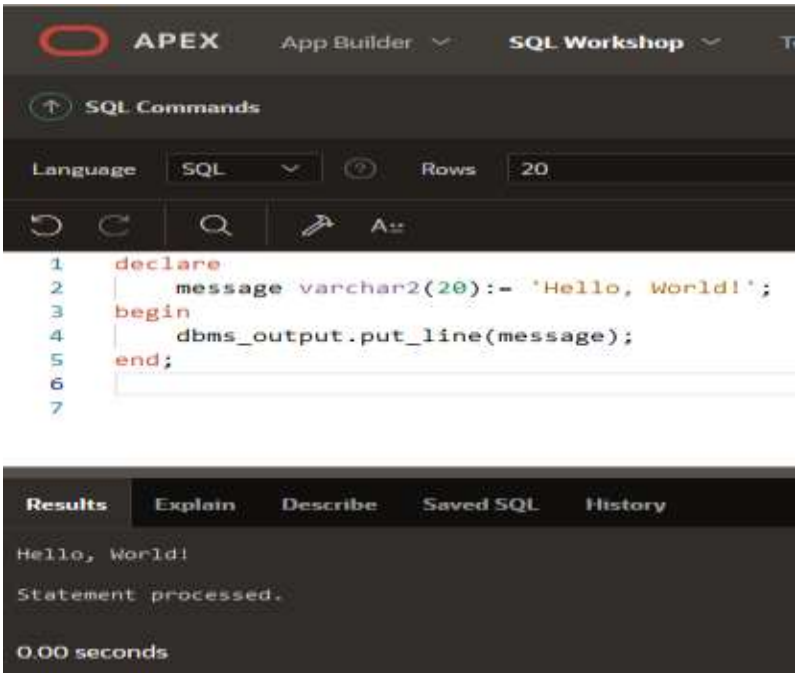
Specificaţi ce se va afişa la rularea următoarelor **programe (blocuri) PL/SQL**:

1)

```
begin
        dbms_output.put_line('Hello World');
end;
```

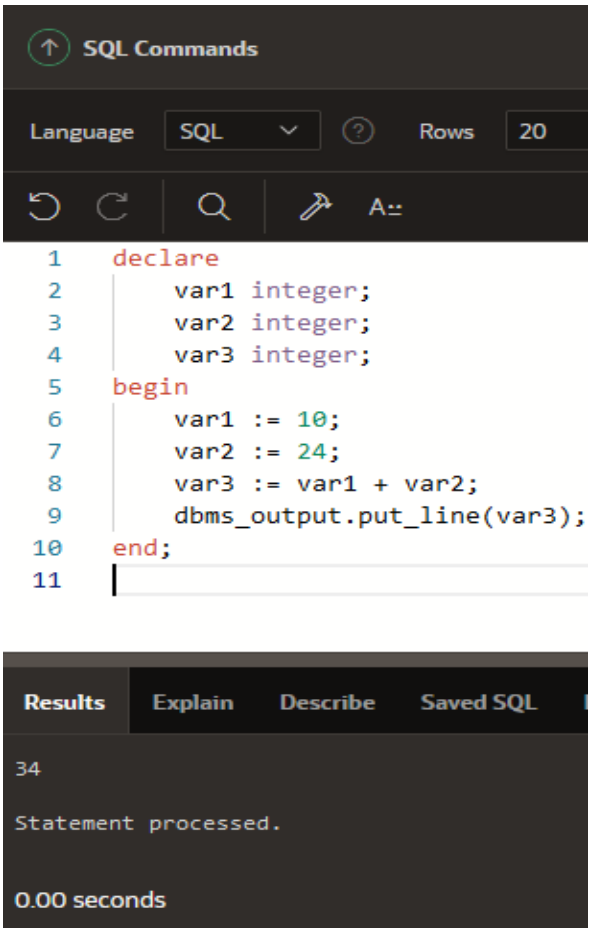



```
declare
        message varchar2(20):= 'Hello, World!';
begin
        dbms_output.put_line(message);
end;
```



2) Program PL/SQL pentru adunarea a doua numere date.

```
declare
        var1 integer;
        var2 integer;
        var3 integer;
begin
        var1:=10;
        var2:=24;
        var3:=var1+var2;
        dbms_output.put_line(var3);
end;
```

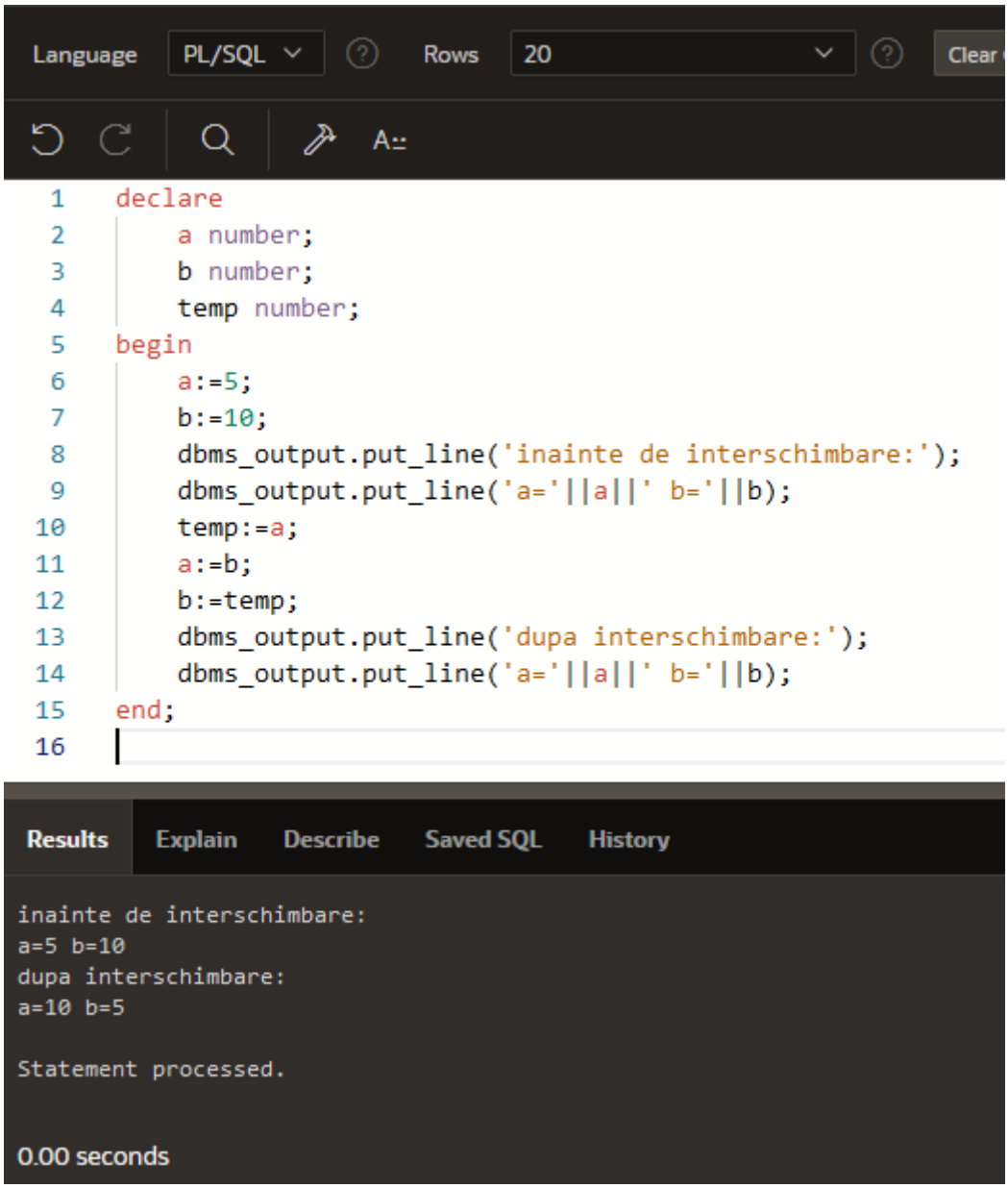


3) Program PL/SQL pentru interschimbarea a doua numere.

Avem doua soluții de interschimbare a doua numere

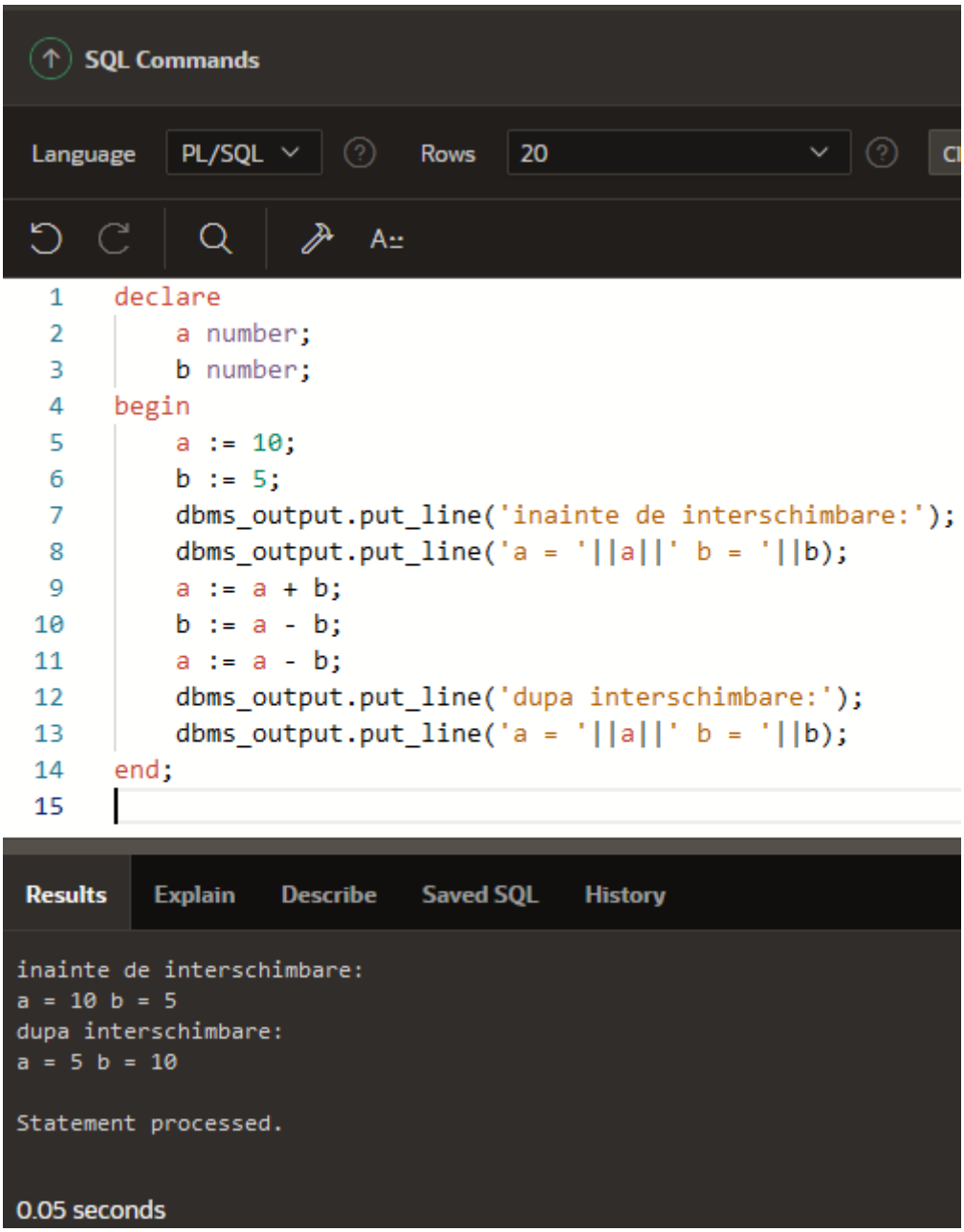
Solutia 1: cu o variabila temporara

```
declare
   a number;
   b number;
   temp number;
begin
   a:=5;
   b:=10;
   dbms_output.put_line('inainte de interschimbare:');
   dbms_output.put_line('a='||a||' b='||b);
   temp:=a;
   a:=b;
   b:=temp;
   dbms_output.put_line('dupa interschimbare:');
   dbms_output.put_line('a='||a||' b='||b);
end;
```



Solutia 2: Fara utilizarea unei variabile temporare

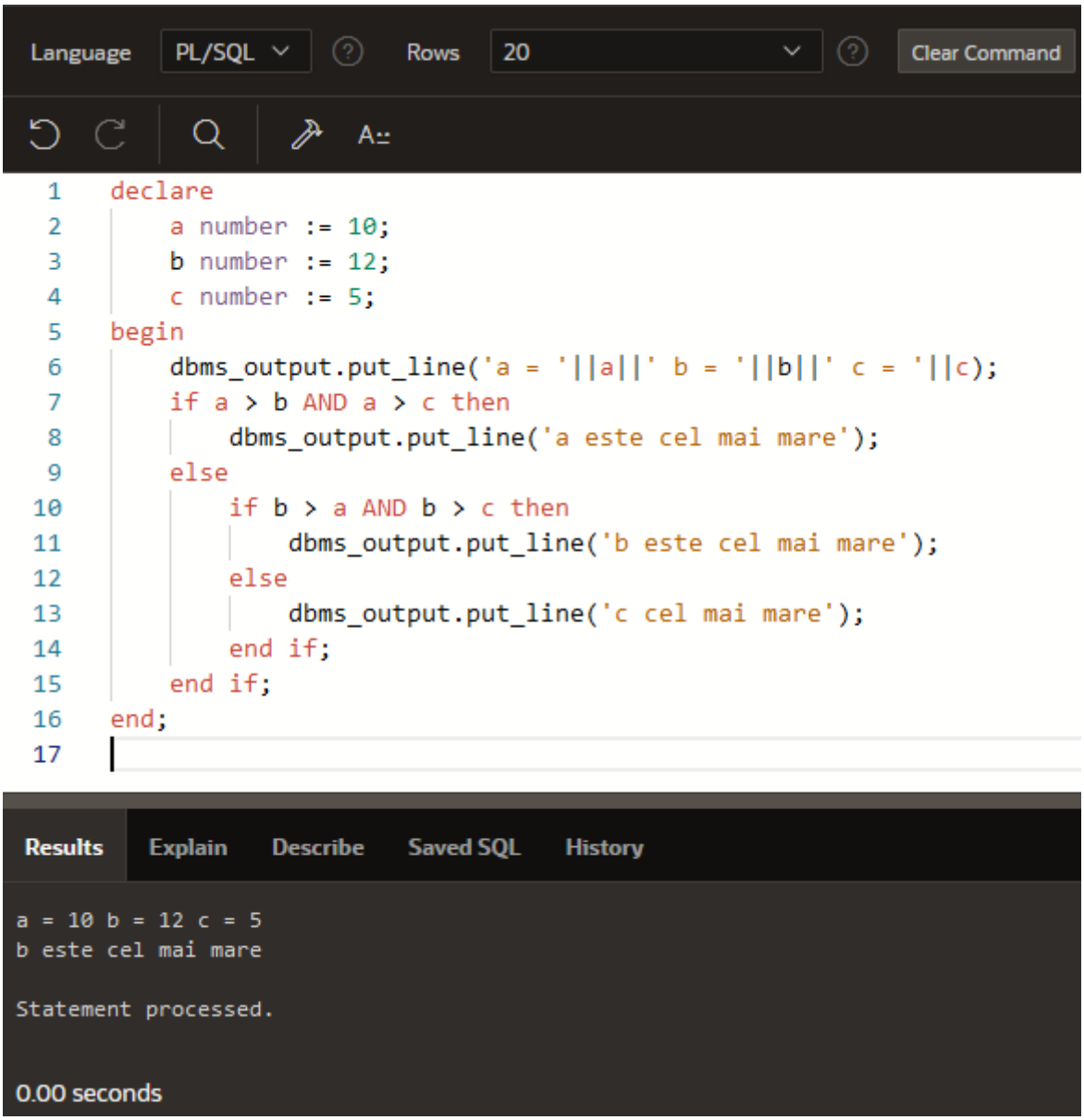
```
declare
   a number;
   b number;
begin
   a := 10;
   b := 5;
   dbms_output.put_line('inainte de interschimbare:');
   dbms_output.put_line('a = '||a||' b = '||b);
   a := a + b;
   b := a - b;
   a := a - b;
   dbms_output.put_line('dupa interschimbare:');
   dbms_output.put_line('a = '||a||' b = '||b);
end;
```



4) Program PL/SQL pentru a afla care este cel mai mare număr dintre 3(trei) valori date.

```
declare
        a number := 10;
        b number := 12;
        c number := 5;
begin
        dbms_output.put_line('a = '||a||' b = '||b||' c = '||c);
        if a > b AND a > c then
                dbms_output.put_line('a este cel mai mare');
        else
                if b > a AND b > c then
                        dbms_output.put_line('b este cel mai mare');
```

```
            else
                dbms_output.put_line('c cel mai mare');
            end if;
        end if;
end;
```

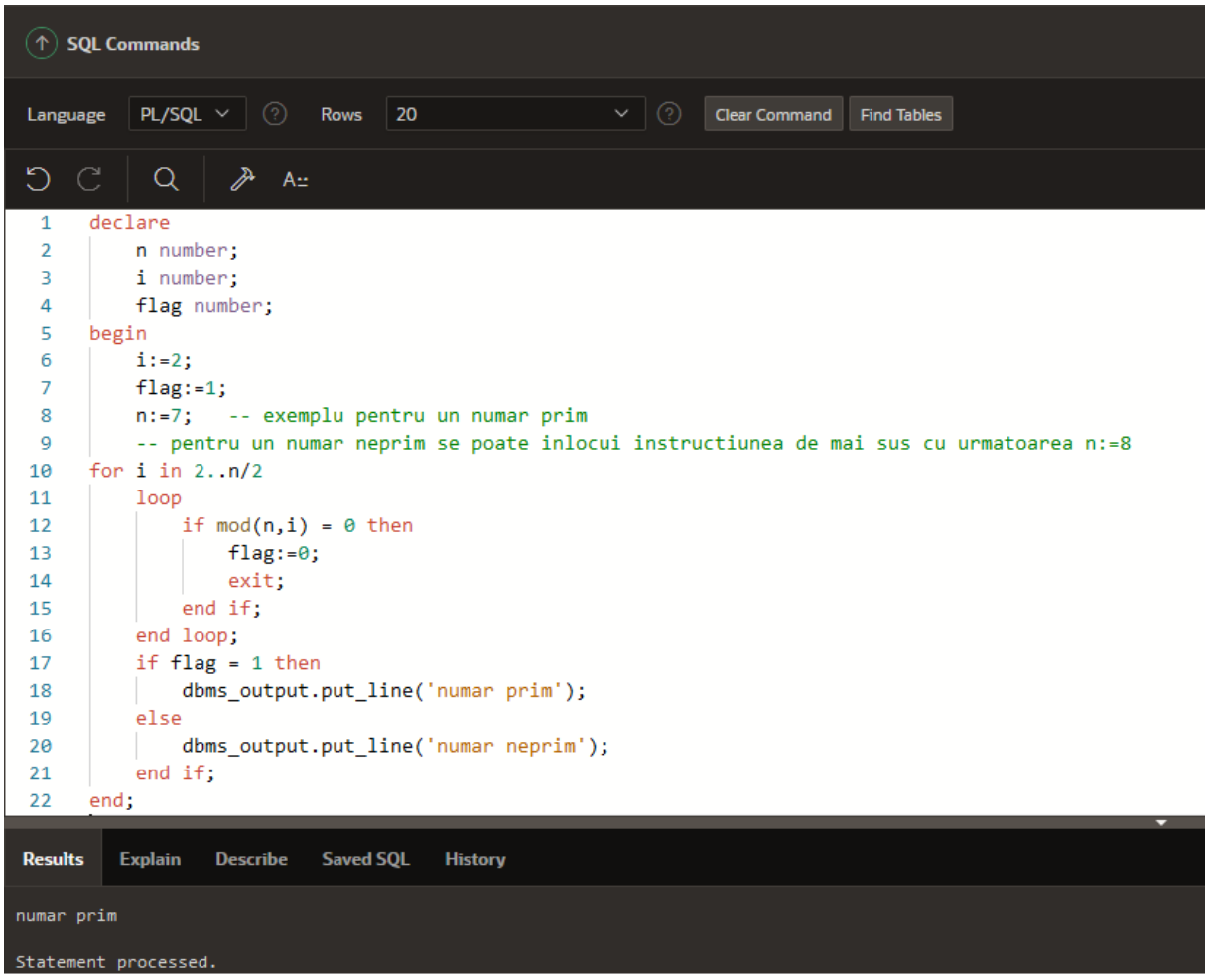


5) Program PL/SQL pentru verificarea primalitatii unui numar.

Un **numar este prim** daca este divizibil cu 1 si cu el insusi. De exemplu, numerele 2, 3, 5, 7, s.a.m.d. In schimb numerele 4, 6, 8, s.a.m.d. nu sunt numere prime.

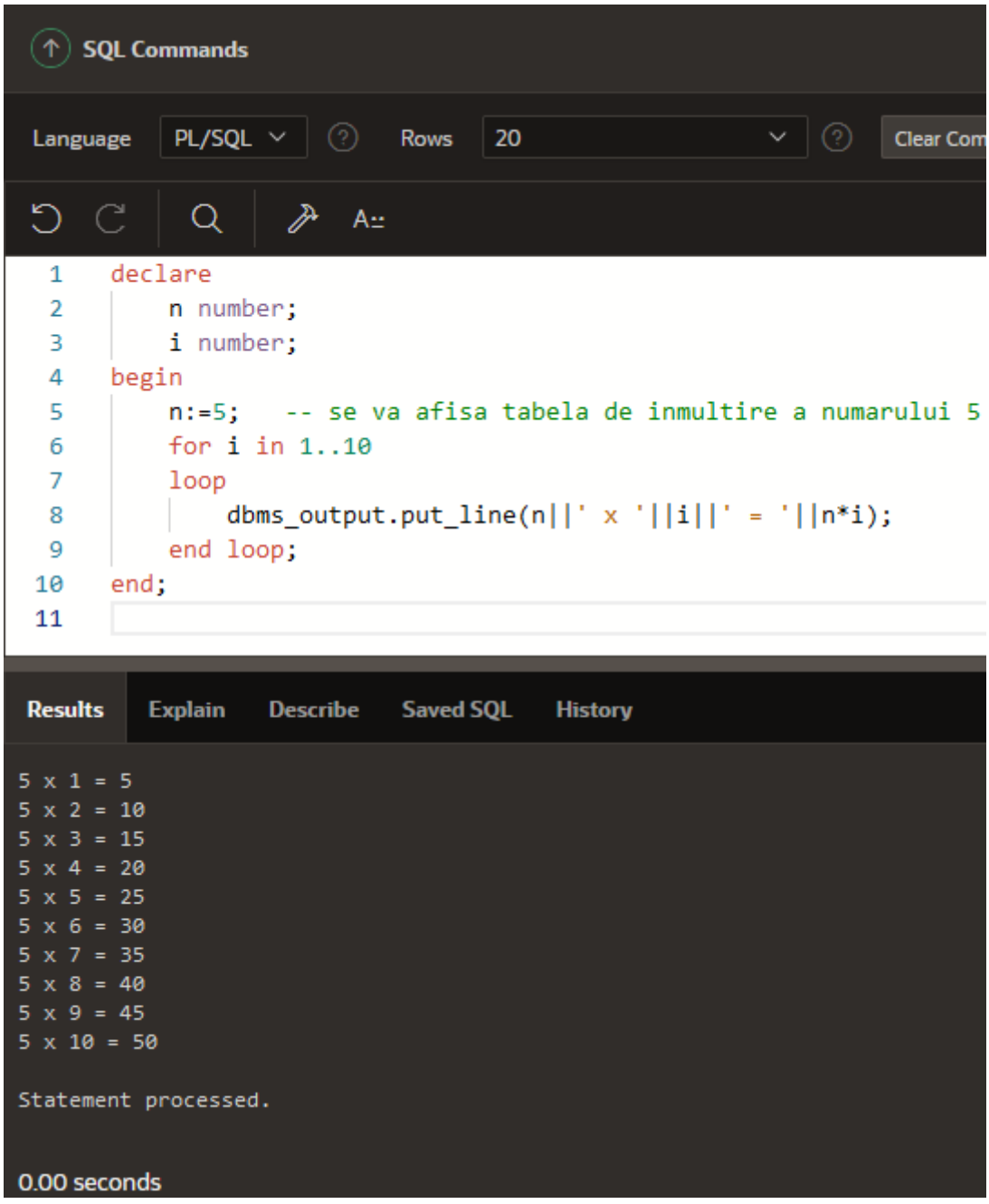
```
declare
        n number;
        i number;
        flag number;
begin
```

```
        i:=2;
        flag:=1;
        n:=7;  -- exemplu pentru un numar prim
-- pentru un numar neprim se poate inlocui instructiunea de mai sus cu urmatoarea n:=8
        for i in 2..n/2
        loop
                if mod(n,i)=0
                then
                        flag:=0;
                        exit;
                end if;
        end loop;
        if flag=1
        then
                dbms_output.put_line('numar prim');
        else
                dbms_output.put_line('numar neprim');
        end if;
end;
```



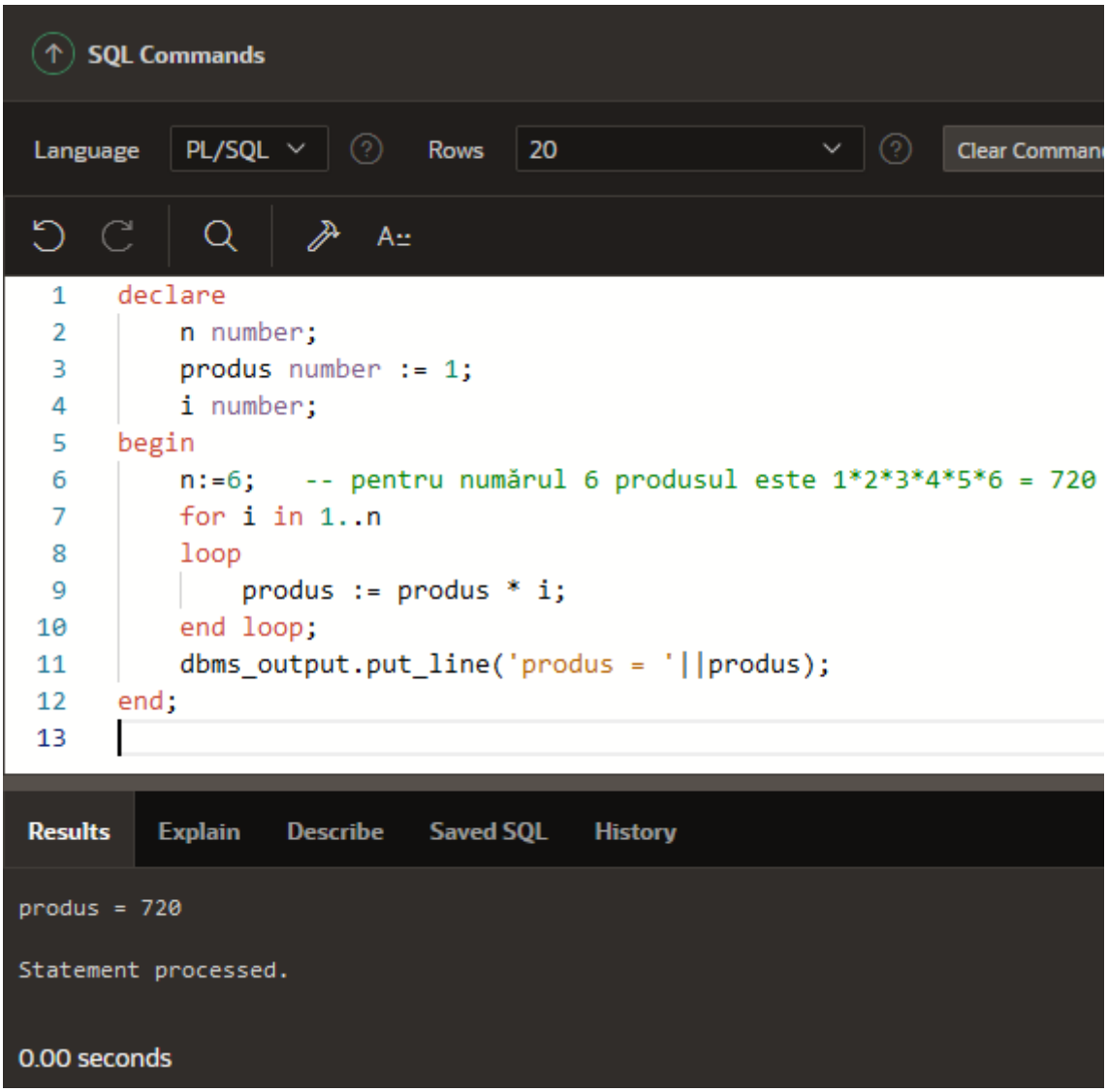
6) Program PL/SQL pentru afișarea tabelei de înmulțire a unui număr dat.

```
declare
        n number;
        i number;
begin
        n:=5;  -- se va afisa tabela de inmultire a numarului 5
        for i in 1..10
        loop
                dbms_output.put_line(n||' x '||i||' = '||n*i);
        end loop;
end;
```



7) Program PL/SQL pentru calculul produsului tuturor numerelor mai mici decât numărul n dat.

```
declare
        n number;
        produs number := 1;
        i number;
begin
        n:=6;  -- pentru numărul 6 produsul este 1*2*3*4*5*6 = 720
        for i in 1..n
        loop
                produs := produs * i;
        end loop;
        dbms_output.put_line('produs = '||produs);
end;
```



8) Program PL/SQL pentru afișarea următoarei figuri:

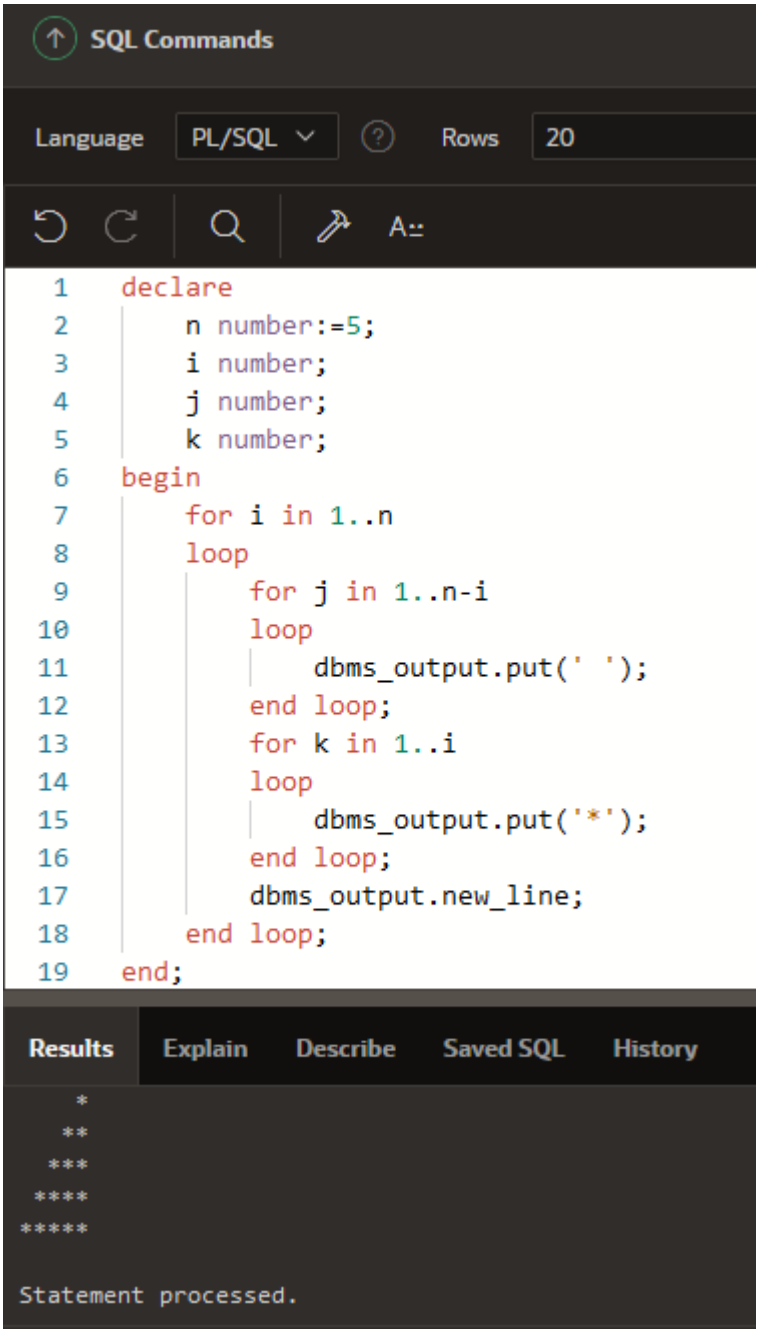
```
*
**
***
****
*****
```

```
declare
        n number:=5;
        i number;
        j number;
begin
        for i in 1..n
        loop
                for j in 1..i
                loop
                        dbms_output.put('*');
                end loop;
                dbms_output.new_line;
        end loop;
end;
```

9) Program PL/SQL pentru afișarea următoarei figuri:

```
    *
   **
  ***
 ****
*****
```

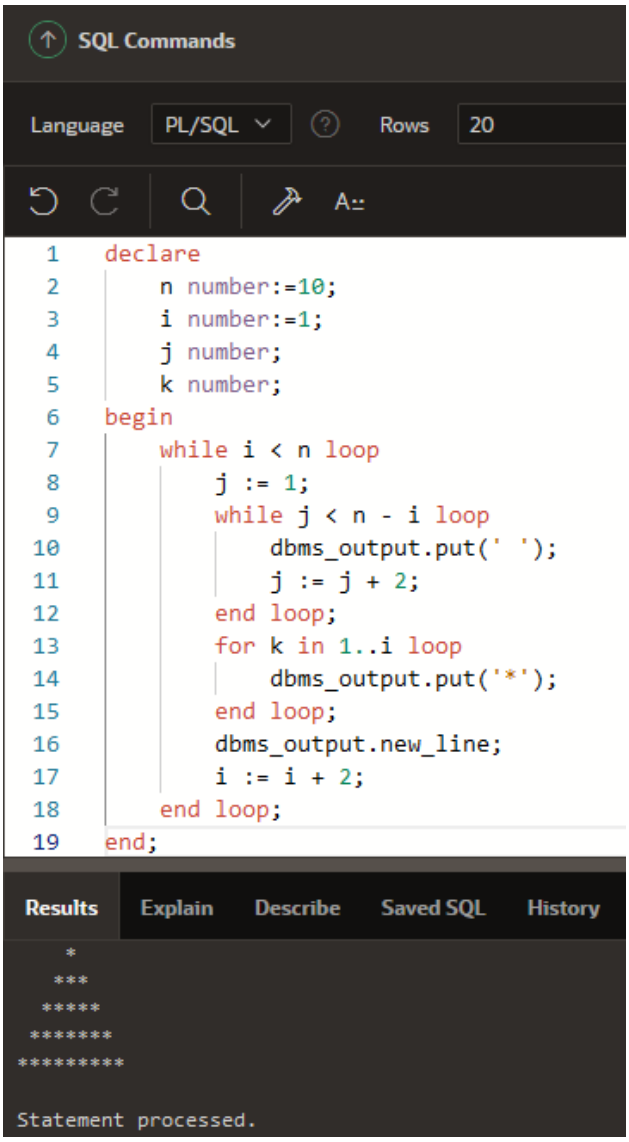
```
declare
        n number:=5;
        i number;
        j number;
        k number;
begin
        for i in 1..n
        loop
                for j in 1..n-i
                loop
                    dbms_output.put(' ');
                end loop;
                for k in 1..i
                loop
                    dbms_output.put('*');
                end loop;
                dbms_output.new_line;
        end loop;
end;
```



10) Program PL/SQL pentru afișarea următoarei figuri:

```
    *
   ***
  *****
 *******
*********
```

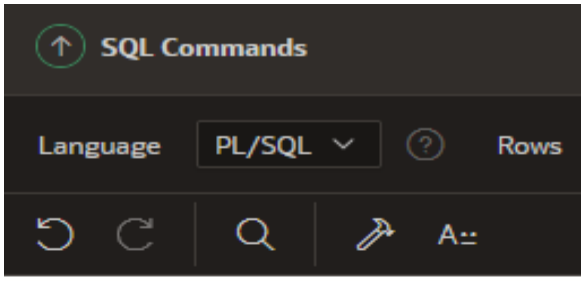
```
declare
        n number:=10;
        i number:=1;
        j number;
        k number;
begin
        while i<n loop
                j:=1;
                while j<n-i loop
                        dbms_output.put(' ');
                        j:=j+2;
                end loop;
                for k in 1..i
                loop
                        dbms_output.put('*');
                end loop;
                dbms_output.new_line;
                i:=i+2;
        end loop;
end;
```



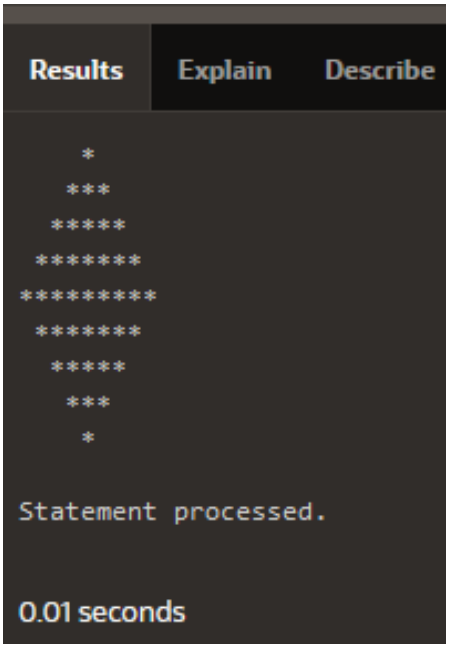
11) Program PL/SQL pentru afișarea următoarei figuri:

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

```
declare
        n number:=10;
        i number:=1;
        j number;
        k number;
begin
        while i < n loop
                j := 1;
                while j < n - i loop
                        dbms_output.put(' ');
                        j := j + 2;
                end loop;
                for k in 1..i loop
                        dbms_output.put('*');
                end loop;
                dbms_output.new_line;
                i := i + 2;
        end loop;
        i := 7;
        while i >= 1 loop
                j := 9;
                while j > i loop
                        dbms_output.put(' ');
                        j := j - 2;
                end loop;
                for k in 1..i loop
                        dbms_output.put('*');
                end loop;
                dbms_output.new_line;
                i := i - 2;
        end loop;
end;
```




```
1   declare
2       n number:=10;
3       i number:=1;
4       j number;
5       k number;
6   begin
7       while i < n loop
8           j := 1;
9           while j < n - i loop
10              dbms_output.put(' ');
11              j := j + 2;
12          end loop;
13          for k in 1..i loop
14              dbms_output.put('*');
15          end loop;
16          dbms_output.new_line;
17          i := i + 2;
18      end loop;
19      i := 7;
20      while i >= 1 loop
21          j := 9;
22          while j > i loop
23              dbms_output.put(' ');
24              j := j - 2;
25          end loop;
26          for k in 1..i loop
27              dbms_output.put('*');
28          end loop;
29          dbms_output.new_line;
30          i := i - 2;
31      end loop;
32  end;
```



Probleme propuse spre rezolvare

Scrieți **programe PL/SQL (blocuri anonime)** pentru următoarele cerințe:

1) Sa se calculeze diferența a doua numere întregi. De exemplu: primul număr este 87 si cel de-al doilea număr este 34.
2) Sa se afișeze care este cel mai mic număr dintre 2(doua) numere întregi date. De exemplu: primul număr este 123, iar al doilea număr este 11.
3) Sa se verifice daca un număr întreg dat este sau număr par. De exemplu numărul 34.
4) Sa se afișeze care este cel mai mic număr dintre 3(trei) numere întregi date. De exemplu: primul număr este 323, al doilea număr 131, iar al treilea număr este 1234.
5) Sa se afișeze tabela de adunare a unui număr dat. De exemplu, pentru numărul 8.
6) Sa calculeze suma primelor n numere întregi mai mici decât n. De exemplu, pentru numărul n=6, suma este egala cu 1+2+3+4+5+6 = 21
7) Sa se afișeze următoarea figură:

```
*****
****
***
**
*
```

8) Sa se afișeze următoarea figură:

```
*****
 ****
  ***
   **
    *
```

9) Sa se afișeze următoarea figură:

```
*********
 *******
  *****
   ***
    *
```

10) Sa se afișeze următoarea figură:

```
*********
 *******
  *****
   ***
    *
    *
   ***
  *****
 *******
```

Bibliografie web: **https://www.thecrazyprogrammer.com/plsql-programs-examples**