

### **Prácticas sobre conceptos básicos de: PL-SQL**

Antes de hacer los ejercicios, analizar y realizar los ejemplos que os he hecho para que veáis de forma práctica como plantearlos. Comentamos cualquier duda. Repasar el tema de estructuras de control PL/SQL.

#### **DEFINICIÓN DE BUCLES.**

- Los bucles simples **LOOP** tienen la siguiente sintaxis:

#### **LOOP**

secuencia de instrucciones;  
EXIT [WHEN condición];  
**END LOOP;**

La sintaxis EXIT [WHEN condición]; es equivalente a:

```
IF condición THEN  
  EXIT;  
END IF;
```

La sintaxis completa es :

#### **LOOP**

```
IF condición THEN  
  EXIT;  
END IF;
```

#### Ejemplo1

```
declare  
v_contador number:=1;  
begin  
loop  
v_contador := v_contador +1;  
if v_contador >20 THEN  
EXIT ;  
end if;  
end loop;  
dbms_output.put_line(v_contador);  
end;
```



```
1 declare
2 v_contador number:=1;
3 begin
4 loop
5 v_contador := v_contador +1;
6 if v_contador >20 THEN
7 EXIT ;
8 end if;
9 end loop;
10 dbms_output.put_line(v_contador);
11 end;
```

**Resultados**   Explicar   Describir   SQL Guardado   Historial

21

Sentencia procesada.

0,01 segundos

- Los bucles **WHILE** tienen la siguiente sintaxis:

### WHILE condición LOOP

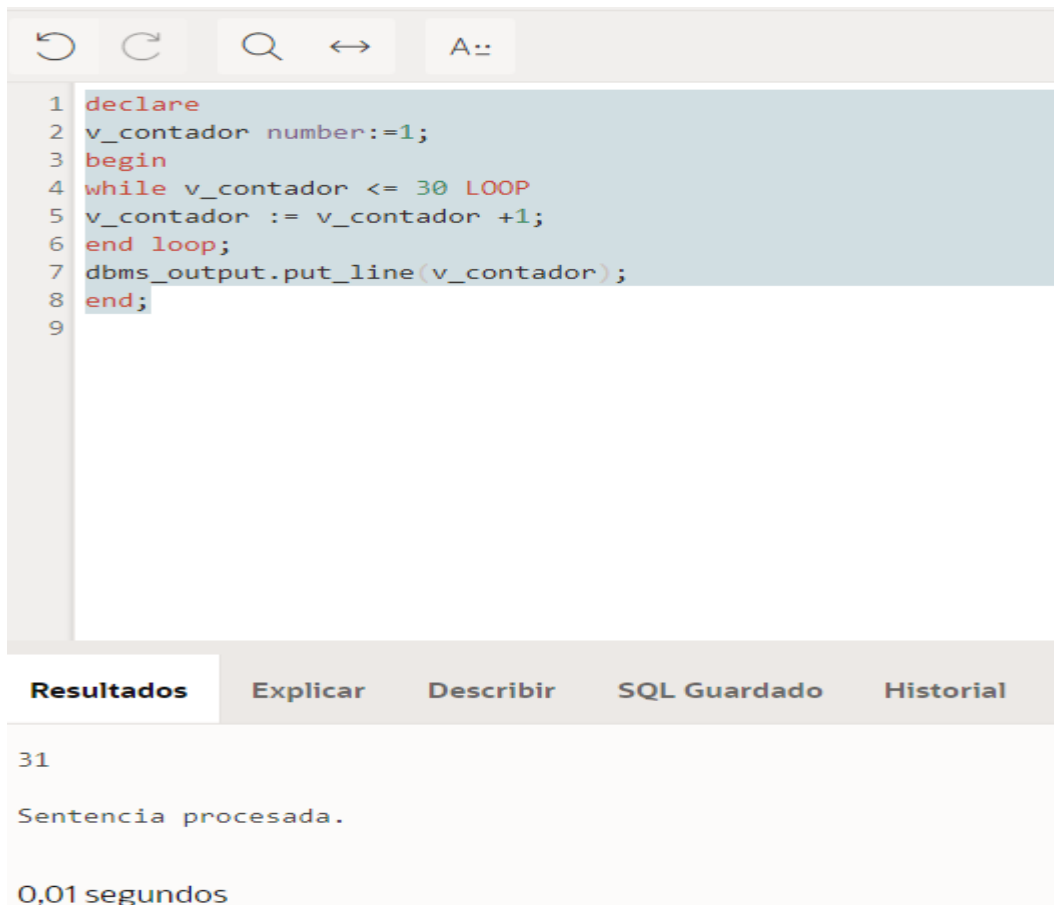
secuencia de instrucciones

END LOOP;

Se produce una evaluación de la condición previa a cada iteración del bucle. Si la condición es verdadera, se ejecuta la secuencia de órdenes , pero si es falsa, el bucle termina y se transfiere el control a las instrucciones posteriores a END LOOP.

#### Ejemplo2

```
declare
v_contador number:=1;
begin
while v_contador <= 30 LOOP
v_contador := v_contador +1;
end loop;
dbms_output.put_line(v_contador);
end;
```



The screenshot shows a SQL IDE interface. The top toolbar contains icons for undo, redo, search, and a dropdown menu. The main editor area contains the following PL/SQL code:

```
1 declare
2 v_contador number:=1;
3 begin
4 while v_contador <= 30 LOOP
5 v_contador := v_contador +1;
6 end loop;
7 dbms_output.put_line(v_contador);
8 end;
9
```

Below the editor, there are five tabs: **Resultados**, **Explicar**, **Describir**, **SQL Guardado**, and **Historial**. The **Resultados** tab is active, displaying the output of the execution:

```
31
Sentencia procesada.
0,01 segundos
```

- Los bucles **FOR** tienen la siguiente sintaxis

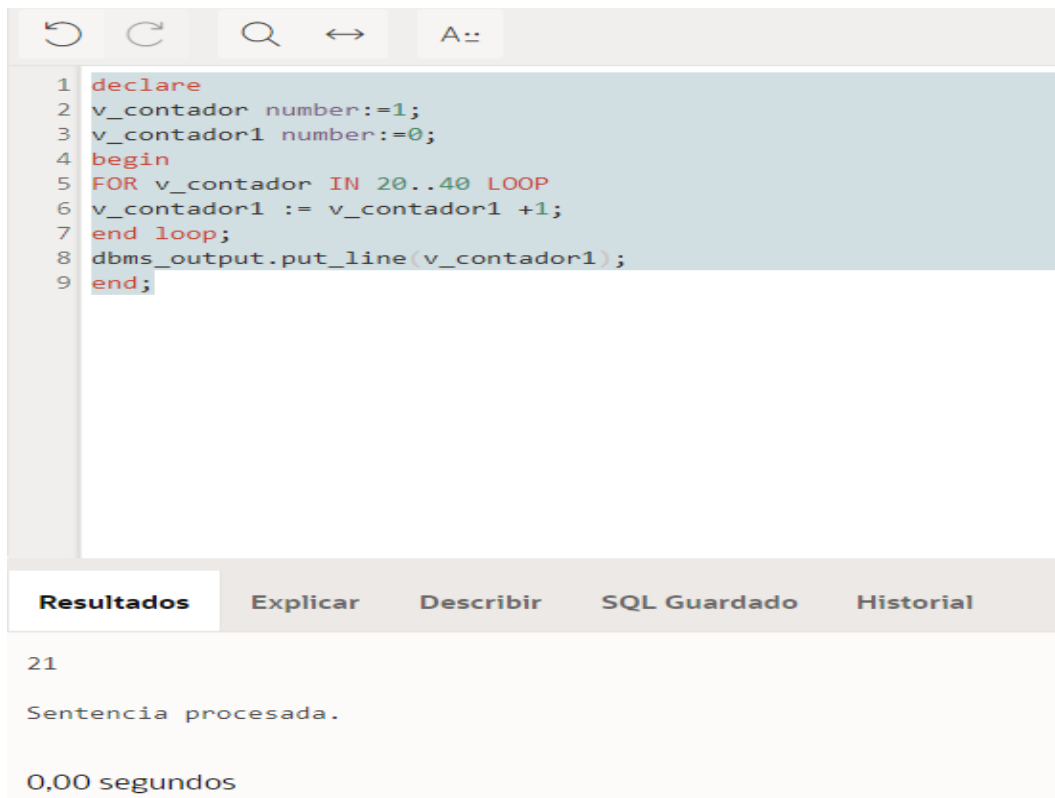
**FOR** contador **IN** [**REVERSE**] límite inferior..límite superior  
**LOOP**  
  secuencia de instrucciones  
**END LOOP**;

Los bucles **FOR** numéricos disponen de un número de iteraciones definido.

### Ejemplo3

```
declare
v_contador number:=1;
v_contador1 number:=0;
begin
FOR v_contador IN 20..40 LOOP
v_contador1 := v_contador1 +1;
```

```
end loop;  
dbms_output.put_line(v_contador1);  
end;
```



The screenshot shows a SQL IDE interface. The top toolbar contains icons for undo, redo, search, and other functions. The main editor area displays the following PL/SQL code:

```
1 declare  
2 v_contador number:=1;  
3 v_contador1 number:=0;  
4 begin  
5 FOR v_contador IN 20..40 LOOP  
6 v_contador1 := v_contador1 +1;  
7 end loop;  
8 dbms_output.put_line(v_contador1);  
9 end;
```

Below the editor, there is a tabbed interface with the following tabs: Resultados, Explicar, Describir, SQL Guardado, and Historial. The 'Resultados' tab is active, showing the output of the execution:

```
21  
  
Sentencia procesada.  
  
0,00 segundos
```

➤ Órdenes **GOTO** y etiquetas.

**PL/SQL** dispone de la orden de salto **GO TO**. Para identificar el lugar al que se tiene que realizar el salto, se utiliza una etiqueta que suele encerrarse entre corchetes angulares dobles. La sintaxis es la siguiente:

**GOTO** nombre de etiqueta

secuencia posible

**BEGIN**

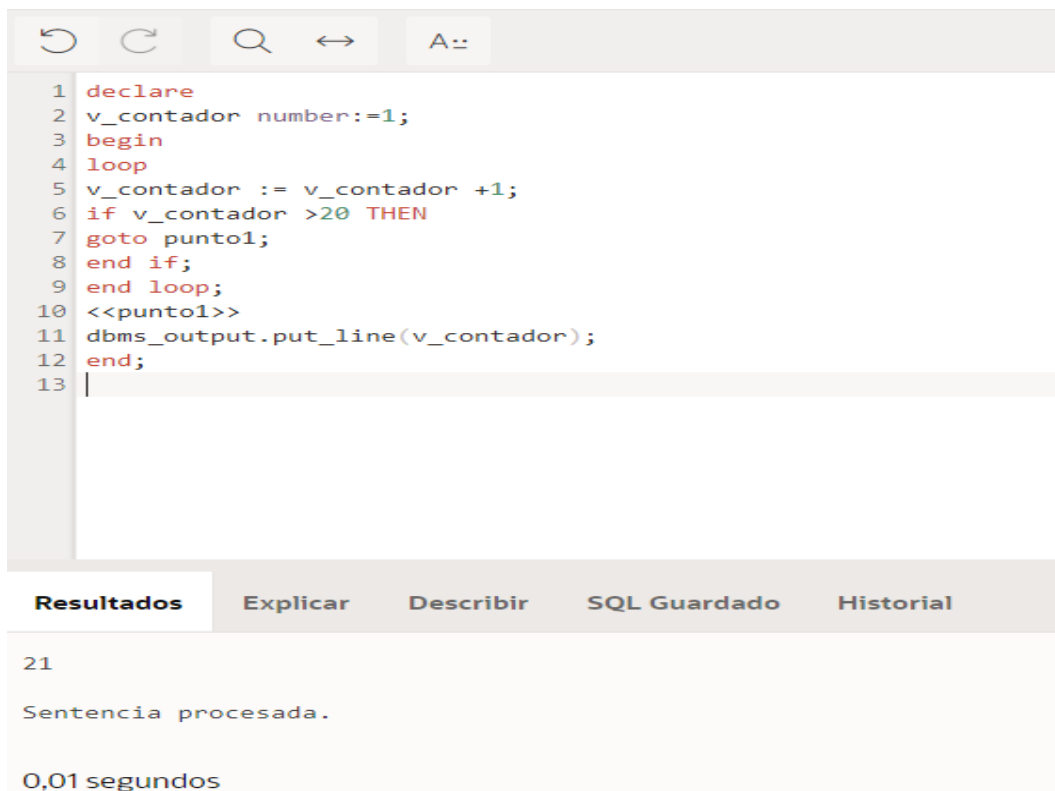
· **GOTO** nombre de etiqueta;

· <<nombre etiqueta>>

· **END;**

### Ejemplo4

```
declare
v_contador number:=1;
begin
loop
v_contador := v_contador +1;
if v_contador >20 THEN
goto punto1;
end if;
end loop;
<<punto1>>
dbms_output.put_line(v_contador);
end;
```



```
1 declare
2 v_contador number:=1;
3 begin
4 loop
5 v_contador := v_contador +1;
6 if v_contador >20 THEN
7 goto punto1;
8 end if;
9 end loop;
10 <<punto1>>
11 dbms_output.put_line(v_contador);
12 end;
13 |
```

**Resultados**   Explicar   Describir   SQL Guardado   Historial

21

Sentencia procesada.

0,01 segundos

### **Práctica1**

1. Construir un bloque LOOP para insertar solo una fila en la tabla EMPLE. Utilizar la sentencia INSERT para ello. Mensaje al final del proceso.

### **Práctica2**

1. Construir un bloque WHILE para insertar solo una fila en la tabla EMPLE. Utilizar la sentencia INSERT para ello. Mensaje al final del proceso.

### **Práctica3**

1. Construir un bloque FOR para insertar solo una fila en la tabla EMPLE. Utilizar la sentencia INSERT para ello. Mensaje al final del proceso.

### **Práctica4**

1. Construir un bloque con LOOP y GOTO de salida para insertar solo una fila en la tabla EMPLE. Utilizar la sentencia INSERT para ello. Mensaje al final del proceso.

### **Práctica5**

Construir un bloque FOR numérico que muestre 5 líneas por pantalla . Visualizaremos dos campos : El contador de 1 a 10 y otro campo que sea una expresión algebraica asociada a una variable que sea igual a (contador \* 2) + un dato constante con un valor de 50. Mensaje al final del proceso.