Using the EXIT Statement

The `EXIT` statement forces a loop to complete unconditionally. When an `EXIT` statement is encountered, the loop completes immediately and control passes to the next statement.

Muestrame 25 números múltiplos de 5.

```
DECLARE
 CONSTANTE NUMBER(1):=5;
 CONTADOR NUMBER(2):=1;

BEGIN

  LOOP

   DBMS_OUTPUT.PUT_LINE(CONTADOR*CONSTANTE);
   CONTADOR:=CONTADOR+1;

   EXIT WHEN CONTADOR=26;

   END LOOP;

END;
```

Ejemplo 1

- **Dado un número solicitado de no más de 3 cifras, muestra todos los números naturales más pequeños que él ordenados de mayor a menor (Naturales: positivos mayores o iguales a 0).**

```
DECLARE

NUMERO NUMBER(2):=&NUMERO;
BEGIN
  LOOP

   DBMS_OUTPUT.PUT_LINE(NUMERO-1);
   NUMERO:=NUMERO-1;
   EXIT WHEN NUMERO=0;

   END LOOP;

END;
```

Ejemplo 2

- **Dado un número solicitado de no más de 3 cifras, muestra la suma de todos los números naturales más pequeños que él.**

```
DECLARE
 NUMERO NUMBER(2):=&NUMERO;
 ACUMULADOR NUMBER(4):=0;

BEGIN
 LOOP
 NUMERO:=NUMERO-1;
 ACUMULADOR:=ACUMULADOR+NUMERO;
 EXIT WHEN NUMERO=0;
 END LOOP;
 DBMS_OUTPUT.PUT_LINE('La suma es '||ACUMULADOR);
END;
```

Ejemplo 3

- **Averigua si un número dado es primo**

```
DECLARE
  NUMERO NUMBER(3):=&NUMERO;
  CONTADOR NUMBER(2):=0;
  DIVISOR NUMBER(3):=0;

BEGIN
  LOOP
    DIVISOR:=DIVISOR+1;
    IF MOD(NUMERO,DIVISOR)=0 THEN CONTADOR:=CONTADOR+1;
    END IF;
    EXIT WHEN DIVISOR=NUMERO OR CONTADOR>2;
  END LOOP;

  IF CONTADOR=1 OR CONTADOR=2 THEN DBMS_OUTPUT.PUT_LINE('El número '||NUMERO||' es primo');
  ELSE DBMS_OUTPUT.PUT_LINE('El número '||NUMERO||' no es primo');
  END IF;
END;
```

Ejemplo 4

- **Dada una letra, muestra el resto del abecedario**

```
DECLARE
 LETRA VARCHAR2(1):='&LETRA';
 CONTADOR NUMBER(3):=0;
BEGIN
 LOOP
 CONTADOR:=ASCII(LETRA)+1;
 LETRA:=CHR(CONTADOR);
 DBMS_OUTPUT.PUT_LINE(LETRA);
 EXIT WHEN CHR(CONTADOR)='Z';
 END LOOP;
END;
```

Ejemplo 5

- **Amplia el ejercicio anterior e indica al final cuántas letras has mostrado.**

```
DECLARE
 LETRA VARCHAR2(1):='&LETRA';
 CONTADOR NUMBER(3):=0;
 CONTADORLETRAS NUMBER(2):=0;
BEGIN
 LOOP
 CONTADOR:=ASCII(LETRA)+1;
 LETRA:=CHR(CONTADOR);
 DBMS_OUTPUT.PUT_LINE(LETRA);
 CONTADORLETRAS:=CONTADORLETRAS+1;
 EXIT WHEN CHR(CONTADOR)='Z';
 END LOOP;
 DBMS_OUTPUT.PUT_LINE('Total de letras visualizadas: '||CONTADORLETRAS);
END;
```

Using the WHILE-LOOP Statement

The `WHILE-LOOP` statement executes the statements in the loop body as long as a condition is true:

```
WHILE condition LOOP
sequence_of_statements
END LOOP;
```

Ejemplo 6

- **Calcula el producto de 2 números por el método de las sumas sucesivas.**

```
DECLARE
 n1 number(3):=&n1;
 n2 number(3):=&n2;
 CONTADOR number(6):=1;
BEGIN

WHILE n2!=0 LOOP
 dbms_output.put_line(n1*contador);
 CONTADOR:=CONTADOR+1;
 n2:=n2-1;
END LOOP;
dbms_output.put_line(n1||' x '||(contador-1)||' = '||n1*(contador-1));
END;
```

Ejemplo 7

- **Calcula el cociente y el resto de dividir dos números por el método de las restas sucesivas**

```
DECLARE
 n1 number(3):=&n1;
 n2 number(3):=&n2;
 COCIENTE number(3):=0;
 RESTO number(3):=0;
BEGIN
WHILE n1>=n2 LOOP
 n1:=n1-n2;
 COCIENTE:=COCIENTE+1;
END LOOP;
 RESTO:=n1;
 dbms_output.put_line('El cociente es '||COCIENTE||' y el resto es '||RESTO);
END;
```

Using the FOR-LOOP Statement

Simple FOR loops iterate over a specified range of integers. The number of iterations is known before the loop is entered. A double dot (..) serves as the range operator. The range is evaluated when the FOR loop is first entered and is never re-evaluated. If the lower bound equals the higher bound, the loop body is executed once.

```
BEGIN
FOR i IN REVERSE 1..3 LOOP -- assign the values 1,2,3 to i
DBMS_OUTPUT.PUT_LINE (TO_CHAR(i));
END LOOP;
END;
```

Ejemplo 8

- **Muestra en pantalla los primeros 30 múltiplos de 5**

```
DECLARE
 CONSTANTE NUMBER(2):=5;
BEGIN
 FOR i IN 1..30 LOOP
  DBMS_OUTPUT.PUT_LINE(CONSTANTE*i);
 END LOOP;
END;
```

Ejemplo 9

- **Dada una cadena, muestra cada una de sus letras en distintas líneas de pantalla.**

```
DECLARE
 CADENA VARCHAR2(30):='&CADENA';
 LONGITUDCADENA NUMBER(2);
BEGIN
 LONGITUDCADENA:=LENGTH(CADENA);
 FOR i IN 1..LONGITUDCADENA LOOP
  DBMS_OUTPUT.PUT_LINE(SUBSTR(CADENA,i,1));
 END LOOP;
END;
```

Utiliza el bucle que consideres más adecuado

Ejemplo 10

- **Dada una cadena, averigua si es un palíndromo**
  Ejemplos:
  "Se van sus naves"
  "Dábale arroz a la zorra el abad"
  "La ruta nos aportó otro paso natural"
  NOTA: Utiliza replace(cadena, ' ', '') para quitar los espacios intercalados.´

```
DECLARE
 CADENA VARCHAR2(100):='&CADENA';
 CADENA1 VARCHAR2(100);
 CADENA2 VARCHAR2(100):='';
BEGIN
 CADENA1:=UPPER(REPLACE(CADENA,' ',''));
 FOR j IN REVERSE 1..LENGTH(CADENA1) LOOP
  CADENA2:=CADENA2||SUBSTR(CADENA1,j,1);
 END LOOP;
 IF CADENA1=CADENA2 THEN
 DBMS_OUTPUT.PUT_LINE(cadena||' Es un palíndromo');
 ELSE
 DBMS_OUTPUT.PUT_LINE(cadena|| ' No es un palíndromo');
 END IF;
END;
```

Ejemplo 11

- **Muestra en pantalla la tabla de multiplicar de un número que proponga el usuario.**

```
DECLARE
 NUMERO NUMBER(2):=&NUMERO;
BEGIN
 FOR i IN 1..10 LOOP
  DBMS_OUTPUT.PUT_LINE(NUMERO||' x '||i||' = '||NUMERO*i);
 END LOOP;
END;
```

Ejemplo 12

- **Muestra en pantalla todas las tablas de multiplicar.**

```
BEGIN
 FOR l IN 1..10 LOOP
   FOR i IN 1..10 LOOP
    DBMS_OUTPUT.PUT_LINE(l||' x '||i||' = '||l*i);
   END LOOP;
  DBMS_OUTPUT.PUT_LINE('');
 END LOOP;
END;
```