



Universidad Tecnológica de Panamá  
Facultad de Ingeniería de Sistemas Computacionales  
Licenciatura en Ingeniería de Sistemas de Información

Departamento de Sistemas de Información, Control y Evaluación de  
Recursos Informáticos

Sistemas de Bases de Datos II

Equipo #1

Laboratorio #4: "Implementación de Bloques PL/SQL"

Facilitador:

Ing. Henry Lezcano

Integrantes:

Aguilar, Milagros	3-740-771
Atencio, Anel	8-950-868
Márquez, Paola	8-949-1108
Rojas, Reynaldo	8-950-792

Grupo:

11F131

II Semestre, 2020

1. Desarrolle un bloque anónimo que capture (&captura) el nombre de una ciudad española y mande a línea de comando el nombre del equipo que representa la ciudad. El ejercicio será para 3 ciudades. Utilice la estructura del CASE por la estructura de control IF-THEN-ELSE. No se permiten las mismas ciudades por equipo.

```
DECLARE
```

```
-- Declaración de las variables
```

```
v_ciudad varchar(25);
```

```
BEGIN
```

```
-- Sentencia para captura
```

```
v_ciudad:= '&n_ciudad';
```

```
dbms_output.put_line('Nombre del equipo:');
```

```
-- Inicio del case
```

```
CASE v_ciudad
```

```
when 'Valencia' then
```

```
dbms_output.put_line('Valencia CF');
```

```
when 'Madrid' then
```

```
dbms_output.put_line('Atlético de Madrid');
```

```
when 'Sevilla' then
```

```
dbms_output.put_line('Real Betis');
```

```
else
```

```
dbms_output.put_line('Ciudad no válida.');
```

```
END CASE;
```

```
EXCEPTION
```

```
when no_data_found then
```

```
dbms_output.put_line('Error. No se insertó ningún dato');
```

```
END;
```

## PROGRAMA EN EJECUCIÓN

```
SQL> DECLARE
  2  -- Declaración de las variables
  3  v_ciudad varchar(25);
  4  BEGIN
  5  -- Sentencia para captura
  6  v_ciudad:= '&n_ciudad';
  7  dbms_output.put_line('Nombre del equipo:');
  8  -- Inicio del case
  9  CASE v_ciudad
 10  when 'Valencia' then
 11  dbms_output.put_line('Valencia CF');
 12  when 'Madrid' then
 13  dbms_output.put_line('Atlético de Madrid');
 14  when 'Sevilla' then
 15  dbms_output.put_line('Real Betis');
 16  else
 17  dbms_output.put_line('Ciudad no válida.');
```

Enter value for n\_ciudad: Valencia  
old 6: v\_ciudad:= '&n\_ciudad';  
new 6: v\_ciudad:= 'Valencia';  
Nombre del equipo:  
Valencia CF

PL/SQL procedure successfully completed.

```
SQL> DECLARE
  2  -- Declaración de las variables
  3  v_ciudad varchar(25);
  4  BEGIN
  5  -- Sentencia para captura
  6  v_ciudad:= '&n_ciudad';
  7  dbms_output.put_line('Nombre del equipo:');
  8  -- Inicio del case
  9  CASE v_ciudad
 10  when 'Valencia' then
 11  dbms_output.put_line('Valencia CF');
 12  when 'Madrid' then
 13  dbms_output.put_line('Atlético de Madrid');
 14  when 'Sevilla' then
 15  dbms_output.put_line('Real Betis');
 16  else
 17  dbms_output.put_line('Ciudad no válida.');
```

Enter value for n\_ciudad: Madrid  
old 6: v\_ciudad:= '&n\_ciudad';  
new 6: v\_ciudad:= 'Madrid';  
Nombre del equipo:  
Atlético de Madrid

PL/SQL procedure successfully completed.

```

SQL> DECLARE
  2  -- Declaración de las variables
  3  v_ciudad varchar(25);
  4  BEGIN
  5  -- Sentencia para captura
  6  v_ciudad:= '&n_ciudad';
  7  dbms_output.put_line('Nombre del equipo:');
  8  -- Inicio del case
  9  CASE v_ciudad
10  when 'Valencia' then
11  dbms_output.put_line('Valencia CF');
12  when 'Madrid' then
13  dbms_output.put_line('Atlético de Madrid');
14  when 'Sevilla' then
15  dbms_output.put_line('Real Betis');
16  else
17  dbms_output.put_line('Ciudad no válida.');
```

18 END CASE;

```

19 END;
20 /
Enter value for n_ciudad: Sevilla
old   6: v_ciudad:= '&n_ciudad';
new   6: v_ciudad:= 'Sevilla';
Nombre del equipo:
Real Betis

PL/SQL procedure successfully completed.
```

```

SQL> DECLARE
  2  -- Declaración de las variables
  3  v_ciudad varchar(25);
  4  BEGIN
  5  -- Sentencia para captura
  6  v_ciudad:= '&n_ciudad';
  7  dbms_output.put_line('Nombre del equipo:');
  8  -- Inicio del case
  9  CASE v_ciudad
10  when 'Valencia' then
11  dbms_output.put_line('Valencia CF');
12  when 'Madrid' then
13  dbms_output.put_line('Atlético de Madrid');
14  when 'Sevilla' then
15  dbms_output.put_line('Real Betis');
16  else
17  dbms_output.put_line('Ciudad no válida.');
```

18 END CASE;

```

19 END;
20 /
Enter value for n_ciudad: Barcelona
old   6: v_ciudad:= '&n_ciudad';
new   6: v_ciudad:= 'Barcelona';
Nombre del equipo:
Ciudad no válida.

PL/SQL procedure successfully completed.
```

- ```
create table Estudiantes (
    num_est number(5) not null,
    ced_est varchar2(12) not null,
    nombre_est varchar(50) not null,
    calif_final number(3) not null,
    constraint pk_estudiantes_num_est primary key (num_est)
);

insert into Estudiantes values (00001, '8-950-100', 'Marcos Gonzalez', 61);
insert into Estudiantes values (00002, '8-950-200', 'Eduardo Perez', 71);
insert into Estudiantes values (00003, '8-950-300', 'Alice Lara', 81);
insert into Estudiantes values (00004, '8-950-400', 'Hayleen Torres', 91);
insert into Estudiantes values (00005, '8-950-500', 'Tatiana Rodriguez', 100);
```

The screenshot shows the SQL Developer interface with two panes. The top pane, titled 'Hoja de Trabajo' and 'Generador de Consultas', contains the following SQL code:

```
create table Estudiantes (  
    num_est number(5) not null,  
    ced_est varchar2(12) not null,  
    nombre_est varchar(50) not null,  
    calif_final number(3) not null,  
    constraint pk_estudiantes_num_est primary key (num_est)  
);  
  
insert into Estudiantes values (00001, '8-950-100', 'Marcos Gonzalez', 61);  
insert into Estudiantes values (00002, '8-950-200', 'Eduardo Perez', 71);  
insert into Estudiantes values (00003, '8-950-300', 'Alice Lara', 81);  
insert into Estudiantes values (00004, '8-950-400', 'Hayleen Torres', 91);  
insert into Estudiantes values (00005, '8-950-500', 'Tatiana Rodriguez', 100);
```

The bottom pane, titled 'Salida de Script', shows the output of the script execution:

Table ESTUDIANTES creado.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

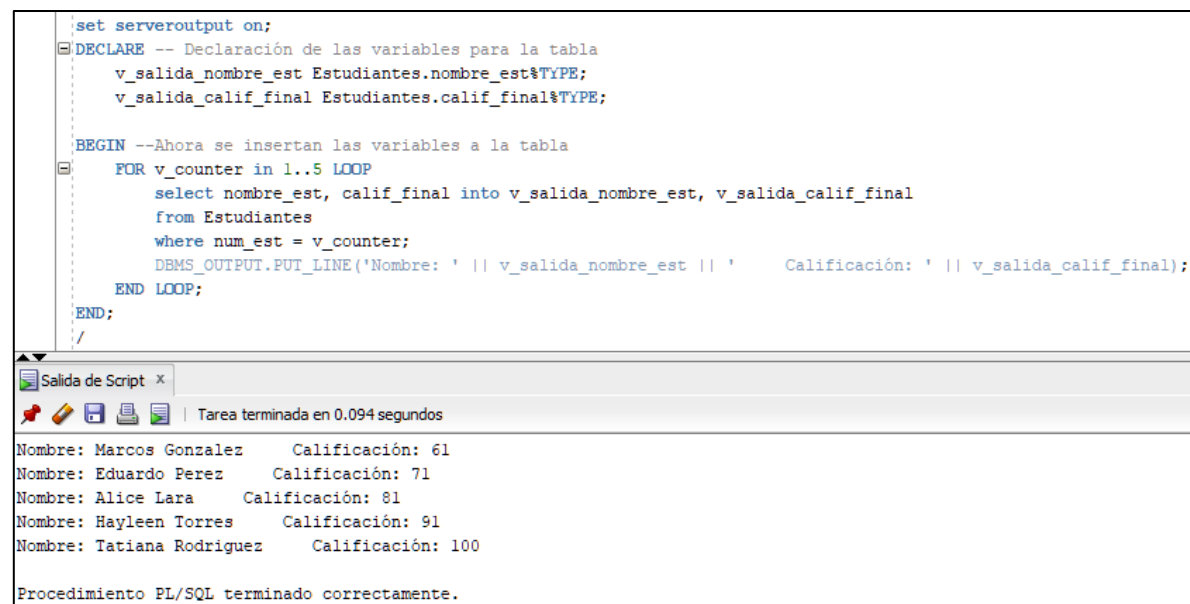
```

set serveroutput on;
DECLARE -- Declaración de las variables para la tabla
    v_salida_nombre_est Estudiantes.nombre_est%TYPE;
    v_salida_calif_final Estudiantes.calif_final%TYPE;

BEGIN --Ahora se insertan las variables a la tabla
    FOR v_counter in 1..5 LOOP
        select nombre_est, calif_final into v_salida_nombre_est, v_salida_calif_final
        from Estudiantes
        where num_est = v_counter;
        DBMS_OUTPUT.PUT_LINE('Nombre: ' || v_salida_nombre_est || '
Calificación: ' || v_salida_calif_final);
    END LOOP;
END;
/

```

## PROGRAMA EN EJECUCIÓN



```

set serveroutput on;
DECLARE -- Declaración de las variables para la tabla
    v_salida_nombre_est Estudiantes.nombre_est%TYPE;
    v_salida_calif_final Estudiantes.calif_final%TYPE;

BEGIN --Ahora se insertan las variables a la tabla
    FOR v_counter in 1..5 LOOP
        select nombre_est, calif_final into v_salida_nombre_est, v_salida_calif_final
        from Estudiantes
        where num_est = v_counter;
        DBMS_OUTPUT.PUT_LINE('Nombre: ' || v_salida_nombre_est || '      Calificación: ' || v_salida_calif_final);
    END LOOP;
END;
/

```

Salida de Script x

Tarea terminada en 0.094 segundos

```

Nombre: Marcos Gonzalez      Calificación: 61
Nombre: Eduardo Perez       Calificación: 71
Nombre: Alice Lara          Calificación: 81
Nombre: Hayleen Torres      Calificación: 91
Nombre: Tatiana Rodriguez   Calificación: 100

Procedimiento PL/SQL terminado correctamente.

```

3. Desarrolle un bloque anónimo que capture un numero entero y determine si este número es primo o no lo es, adicionalmente muestre el resultado en la línea de comando.

DECLARE

V\_Num number;

Div number :=0;

Negative EXCEPTION;

BEGIN

V\_num := '&Numb';

IF V\_Num <=0 THEN

RAISE Negative;

END IF;

FOR i IN 1..V\_Num LOOP

IF mod (V\_Num, i)=0 THEN

Div:=Div+1;

END IF;

END LOOP;

IF Div = 2 THEN

DBMS\_OUTPUT.PUT\_LINE ('Es un numero primo');

ELSE

DBMS\_OUTPUT.PUT\_LINE ('No es un número primo');

END IF;

EXCEPTION

WHEN Negative THEN

DBMS\_OUTPUT.PUT\_LINE ('El número debe ser mayor a 0');

END;

/

## PROGRAMA EJECUTÁNDOSE CON NÚMERO PRIMO

```
SQL> DECLARE
  2  V_Num number;
  3  Div number :=0;
  4  Negative EXCEPTION;
  5  BEGIN
  6  V_num := '&Numb';
  7  IF V_Num <=0 THEN
  8  RAISE Negative;
  9  END IF;
 10  FOR i IN 1..V_Num LOOP
 11  IF mod (V_Num, i)=0 THEN
 12  Div:=Div+1;
 13  END IF;
 14  END LOOP;
 15  IF Div = 2 THEN
 16  DBMS_OUTPUT.PUT_LINE ('Es un numero primo');
 17  ELSE
 18  DBMS_OUTPUT.PUT_LINE ('No es un número primo');
 19  END IF;
 20  EXCEPTION
 21  WHEN Negative THEN
 22  DBMS_OUTPUT.PUT_LINE ('El número debe ser mayor a 0');
 23  END;
 24  /
Enter value for numb: 7
old  6: V_num := '&Numb';
new  6: V_num := '7';
Es un numero primo

PL/SQL procedure successfully completed.
```



## PROGRAMA EJECUTÁNDOSE CON NÚMERO QUE NO ES PRIMO

```
SQL> DECLARE
  2 V_Num number;
  3 Div number :=0;
  4 Negative EXCEPTION;
  5 BEGIN
  6 V_num := '&Numb';
  7 IF V_Num <=0 THEN
  8 RAISE Negative;
  9 END IF;
 10 FOR i IN 1..V_Num LOOP
 11 IF mod (V_Num, i)=0 THEN
 12 Div:=Div+1;
 13 END IF;
 14 END LOOP;
 15 IF Div = 2 THEN
 16 DBMS_OUTPUT.PUT_LINE ('Es un numero primo');
 17 ELSE
 18 DBMS_OUTPUT.PUT_LINE ('No es un número primo');
 19 END IF;
 20 EXCEPTION
 21 WHEN Negative THEN
 22 DBMS_OUTPUT.PUT_LINE ('El número debe ser mayor a 0');
 23 END;
 24 /
Enter value for numb: 4
old 6: V_num := '&Numb';
new 6: V_num := '4';
No es un número primo

PL/SQL procedure successfully completed.
```

## EXCEPTION DEL PROGRAMA

```
SQL> DECLARE
  2 V_Num number;
  3 Div number :=0;
  4 Negative EXCEPTION;
  5 BEGIN
  6 V_num := '&Numb';
  7 IF V_Num <=0 THEN
  8 RAISE Negative;
  9 END IF;
 10 FOR i IN 1..V_Num LOOP
 11 IF mod (V_Num, i)=0 THEN
 12 Div:=Div+1;
 13 END IF;
 14 END LOOP;
 15 IF Div = 2 THEN
 16 DBMS_OUTPUT.PUT_LINE ('Es un numero primo');
 17 ELSE
 18 DBMS_OUTPUT.PUT_LINE ('No es un número primo');
 19 END IF;
 20 EXCEPTION
 21 WHEN Negative THEN
 22 DBMS_OUTPUT.PUT_LINE ('El número debe ser mayor a 0');
 23 END;
 24 /
Enter value for numb: 0
old 6: V_num := '&Numb';
new 6: V_num := '0';
El número debe ser mayor a 0

PL/SQL procedure successfully completed.
```

4. Desarrolle un bloque anónimo que implemente un proceso de repetición para almacenar en una relación de base de datos llamada cumpleaños la identificación que corresponde al contador que controla el ciclo de repetición, nombre y día de cumpleaños de 5 estudiantes de su grupo. Luego un bloque adicional que me permita capturar la identificación y haga una consulta a la relación cumpleaños para conocer el nombre y el día de cumpleaños en línea de comando.

## CREACIÓN DE LA TABLA QUE VAMOS A UTILIZAR

SQL> Create table cumpleaños (

2 ID number not null,

3 nombre varchar2(25) not null,

4 dia date not null,

5 constraint pk\_cumpleanos\_ID primary key (ID)

6 );

```
SQL> Create table cumpleaños (
  2 ID number not null,
  3 nombre varchar2(25) not null,
  4 dia date not null,
  5 constraint pk_cumpleanos_ID primary key (ID)
  6 );
```

Table created.

## CÓDIGO DEL PRIMER BLOQUE ANÓNIMO

Declare

    IDC number;

    nombre cumpleaños.nombre%TYPE;

    dia cumpleaños.dia%TYPE;

Begin

    FOR IDC in 1..5 LOOP

        IF IDC = 1 THEN

            nombre := '&Nombre';

            dia := to\_date('&fecha','DD/MM/YYYY');

            INSERT INTO cumpleaños VALUES(IDC,nombre,dia);

        ELSIF IDC = 2 THEN

            nombre := '&Nombre';

            dia := to\_date('&fecha','DD/MM/YYYY');

            INSERT INTO cumpleaños VALUES(IDC,nombre,dia);

        ELSIF IDC = 3 THEN

            nombre := '&Nombre';

            dia := to\_date('&fecha','DD/MM/YYYY');

            INSERT INTO cumpleaños VALUES(IDC,nombre,dia);

        ELSIF IDC = 4 THEN

            nombre := '&Nombre';

            dia := to\_date('&fecha','DD/MM/YYYY');

            INSERT INTO cumpleaños VALUES(IDC,nombre,dia);

        ELSIF IDC = 5 THEN

            nombre := '&Nombre';

            dia := to\_date('&fecha','DD/MM/YYYY');

            INSERT INTO cumpleaños VALUES(IDC,nombre,dia);

    END IF;

END LOOP;

END;

### PROGRAMA EJECUTÁNDOSE

```
Enter value for nombre: Paola
old 8: nombre := '&Nombre';
new 8: nombre := 'Paola';
Enter value for fecha: 23/10/1999
old 9: dia :=to_date('&fecha','DD/MM/YYYY');
new 9: dia :=to_date('23/10/1999','DD/MM/YYYY');
Enter value for nombre: Reynaldo
old 12: nombre := '&Nombre';
new 12: nombre := 'Reynaldo';
Enter value for fecha: 23/11/1999
old 13: dia :=to_date('&fecha','DD/MM/YYYY');
new 13: dia :=to_date('23/11/1999','DD/MM/YYYY');
Enter value for nombre: Anel
old 16: nombre := '&Nombre';
new 16: nombre := 'Anel';
Enter value for fecha: 18/11/1999
old 17: dia :=to_date('&fecha','DD/MM/YYYY');
new 17: dia :=to_date('18/11/1999','DD/MM/YYYY');
Enter value for nombre: Milagros
old 20: nombre := '&Nombre';
new 20: nombre := 'Milagros';
Enter value for fecha: 06/12/1998
old 21: dia :=to_date('&fecha','DD/MM/YYYY');
new 21: dia :=to_date('06/12/1998','DD/MM/YYYY');
Enter value for nombre: David
old 24: nombre := '&Nombre';
new 24: nombre := 'David';
Enter value for fecha: 09/12/1999
old 25: dia :=to_date('&fecha','DD/MM/YYYY');
new 25: dia :=to_date('09/12/1999','DD/MM/YYYY');

PL/SQL procedure successfully completed.
```

## CÓDIGO DEL SEGUNDO BLOQUE ANÓNIMO

DECLARE

    IDC cumpleanos.ID%TYPE;

    nombre1 cumpleanos.nombre%TYPE;

    dia1 cumpleanos.dia%TYPE;

BEGIN

    IDC := &IDC;

    Select nombre,dia INTO nombre1, dia1

    From cumpleanos

    Where ID=IDC;

    DBMS\_OUTPUT.PUT\_LINE('nombre: ' || nombre1 || 'Dia de nacimiento: ' ||  
dia1);

END;

## CÓDIGO EJECUTÁNDOSE

```
SQL> Set serveroutput on;
SQL> DECLARE
  2     IDC cumpleanos.ID%TYPE;
  3     nombre1 cumpleanos.nombre%TYPE;
  4     dia1 cumpleanos.dia%TYPE;
  5 BEGIN
  6 IDC := &IDC;
  7 Select nombre,dia INTO nombre1, dia1
  8 From cumpleanos
  9 Where ID=IDC;
 10 DBMS_OUTPUT.PUT_LINE('nombre: ' || nombre1 || 'Dia de nacimiento: ' || dia1);
 11 END;
 12 /
Enter value for idc: 3
old 6: IDC := &IDC;
new 6: IDC := 3;
nombre: AnelDia de nacimiento:18-NOV-99
```

***No sabemos que sucedía anteriormente que no imprimía así que colocamos el comando que se ubica antes del 'Declare'.***