

**Universidad Tecnológica de Panamá**

**Facultad de Ingeniería en Sistemas Computacionales**

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

**Licenciatura en Ingeniería en Sistemas de Información Sistemas de  
Bases de Datos II**

**Asignación N°3**

**Facilitador: Ing. Henry Lezcano**

**Integrantes:**

Batista, Johel {8-914-587}

Pinilla, Miguel {8-975-2460}

Riley, Rolando {8-972-1033}

Villarreal, Andrés {8-970-1267}

**Grupo: 1IF-131**

**Fecha de Entrega:** Lunes 26 de Septiembre de 2022

## Indicaciones Generales:

- Realizar las Implementaciones de los Siguietes Bloques Anónimos.
- Trate de incluir la estructura integral de los bloques de ser necesario controlando las excepciones.
- Adicional aplique las guías de estilo de programación del PL/SQL.
- Debe crear las relaciones con las restricciones que sean necesarias para los programas solicitados de ser necesario.

**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.**

### Código del Programa

*DECLARE*

*--Se declara la variable nombre\_ciudad en la que estará almacenado el nombre de la ciudad a la que pertenece un equipo*

*nombre\_ciudad nvarchar2(25);*

*BEGIN*

*--Captura de datos a través de teclado hacia el usuario*

*nombre\_ciudad := '&ciudad';*

*DBMS\_OUTPUT.PUT\_LINE('Ingrese el Nombre del Equipo: ');*

*--Analizaremos la cadena de caracteres ingresada por teclado por el usuario utilizando un CASE*

*CASE nombre\_ciudad*

*WHEN 'Sevilla' THEN*

*DBMS\_OUTPUT.PUT\_LINE('Sevilla F.C.');*

*WHEN 'Salamanca' THEN*

*DBMS\_OUTPUT.PUT\_LINE('Salamandra F.C.');*

*WHEN 'Andalucía' THEN*

*DBMS\_OUTPUT.PUT\_LINE('Andaluz F.C.');*

*ELSE*

*DBMS\_OUTPUT.PUT\_LINE('Ciudad Equivocada, tío');*

*END CASE;*

## Ejecución del programa

```
SQL> DECLARE
2  --Se declara la variable nombre_ciudad en la que estará almacenado el nombre de la ciudad a la que pertenece un equipo
3  nombre_ciudad nvarchar2(25);
4
5  BEGIN
6  --Captura de datos a través de teclado hacia el usuario
7  nombre_ciudad := '&ciudad';
8  DBMS_OUTPUT.PUT_LINE('Ingrese el Nombre del Equipo: ');
9
10 --Analizaremos la cadena de caracteres ingresada por teclado por el usuario utilizando un CASE
11 CASE nombre_ciudad
12 WHEN 'Sevilla' THEN
13 DBMS_OUTPUT.PUT_LINE('Sevilla F.C.');
14 WHEN 'Salamanca' THEN
15 DBMS_OUTPUT.PUT_LINE('Salamandra F.C.');
16 WHEN 'Andalucía' THEN
17 DBMS_OUTPUT.PUT_LINE('Andaluz F.C.');
18 ELSE
19 DBMS_OUTPUT.PUT_LINE('Ciudad Equivocada, tío');
20 END CASE;
21
22 --A continuación, manejaremos las excepciones en el caso de que el usuario no ingrese por teclado ninguna ciudad que esté almacenada en la Base de Datos
23 EXCEPTION
24 WHEN NO_DATA_FOUND THEN
25 DBMS_OUTPUT.PUT_LINE('No se insertó ningún dato, intente denuevo');
26 END;
27 /
```

```
Enter value for ciudad: Sevilla
old   7: nombre_ciudad := '&ciudad';
new   7: nombre_ciudad := 'Sevilla';
Ingrese el Nombre del Equipo:
Sevilla F.C.
```

**2. Desarrolle un bloque anónimo que cargue en una relación o tabla de base de datos llamada estudiante con el número de estudiante, cédula, nombre y calificación final. Luego que realice una consulta a esta tabla de estudiante para mostrar en la línea de comando el nombre del estudiante con la calificación final obtenida.**

**Código del Programa:**

**DECLARE**

-- La siguiente variable guardará el nombre del estudiante y adicional a ello, efectuará la captura del tipo de dato

v\_nombre\_est Estudiantes.nombre\_est%TYPE;

-- La siguiente variable guardará la calificación del estudiante y adicional a ello, capturará el tipo de dato

v\_nota\_final Estudiantes.nota\_final%TYPE;

--La siguiente variable guardará la cédula del estudiante

v\_cedEst Estudiantes.ced\_estudiante%TYPE;

**BEGIN**

-- A continuación, procederemos a insertar los datos a las variables en una tabla temporal

FOR v\_counter IN 1..5

loop

IF v\_counter = 1 THEN

v\_nombre\_est := '&nombre';

v\_cedEst := '&cedulaEst';

v\_nota\_final := '&nota';

INSERT INTO Estudiantes VALUES (v\_counter, v\_cedEst, v\_nombre\_est, v\_nota\_final);

ELSIF v\_counter = 2 THEN

v\_nombre\_est := '&nombre';

v\_cedEst := '&cedulaEst';

v\_nota\_final := '&nota';

```

        INSERT INTO Estudiantes VALUES (v_counter, v_cedEst, v_nombre_est,
v_nota_final);
        ELSIF v_counter = 3 THEN
            v_nombre_est := '&nombre';
            v_cedEst := '&cedulaEst';
            v_nota_final := '&nota';
            INSERT INTO Estudiantes VALUES (v_counter, v_cedEst, v_nombre_est,
v_nota_final);
        ELSIF v_counter = 4 THEN
            v_nombre_est := '&nombre';
            v_cedEst := '&cedulaEst';
            v_nota_final := '&nota';
            INSERT INTO Estudiantes VALUES (v_counter, v_cedEst, v_nombre_est,
v_nota_final);
        ELSIF v_counter = 5 THEN
            v_nombre_est := '&nombre';
            v_cedEst := '&cedulaEst';
            v_nota_final := '&nota';
            INSERT INTO Estudiantes VALUES (v_counter, v_cedEst, v_nombre_est,
v_nota_final);
        END IF;
    end loop;

```

-- Ahora, imprimiremos por pantalla la información solicitada en una concatenación de una cadena de caracteres

```

FOR v_counter IN 1..5
    LOOP

```

--Selección de los datos de la tabla Estudiantes hacia las variables definidas

```

SELECT nombre_est, nota_final INTO v_nombre_est, v_nota_final
FROM Estudiantes
WHERE num_estudiante = v_counter;

```

-- Ahora, imprimiremos por pantalla la información solicitada en una concatenación de una cadena de caracteres

```

        DBMS_OUTPUT.PUT_LINE('Nombre del Estudiante: ' || v_nombre_est || '
Calificación: ' || v_nota_final);
    END LOOP;

```

## EXCEPTION

```

    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('DATA NO ENCONTRADA ERROR');
END;

/

```

## Ejecución del Programa:

```

new 19:          v_nombre_est := 'Miguel P';
Enter value for cedulaest: 6-142-4573
old 20:          v_cedEst := '&cedulaEst';
new 20:          v_cedEst := '6-142-4573';
Enter value for nota: 96
old 21:          v_nota_final := '&nota';
new 21:          v_nota_final := '96';
Enter value for nombre: Andres V
old 24:          v_nombre_est := '&nombre';
new 24:          v_nombre_est := 'Andres V';
Enter value for cedulaest: 2-346-2353
old 25:          v_cedEst := '&cedulaEst';
new 25:          v_cedEst := '2-346-2353';
Enter value for nota: 90
old 26:          v_nota_final := '&nota';
new 26:          v_nota_final := '90';
Enter value for nombre: Rolando R
old 29:          v_nombre_est := '&nombre';
new 29:          v_nombre_est := 'Rolando R';
Enter value for cedulaest: 8-235-9326
old 30:          v_cedEst := '&cedulaEst';
new 30:          v_cedEst := '8-235-9326';
Enter value for nota: 92
old 31:          v_nota_final := '&nota';
new 31:          v_nota_final := '92';
Enter value for nombre: Kevin G
old 34:          v_nombre_est := '&nombre';
new 34:          v_nombre_est := 'Kevin G';
Enter value for cedulaest: 4-1246-4656
old 35:          v_cedEst := '&cedulaEst';
new 35:          v_cedEst := '4-1246-4656';
Enter value for nota: 97
old 36:          v_nota_final := '&nota';
new 36:          v_nota_final := '97';
Nombre del Estudiante: Johel B Calificación: 99
Nombre del Estudiante: Miguel P Calificación: 96
Nombre del Estudiante: Andres V Calificación: 90
Nombre del Estudiante: Rolando R Calificación: 92
Nombre del Estudiante: Kevin G Calificación: 97

PL/SQL procedure successfully completed.

SQL>

```

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

**Código del Programa:**

*--Inicio del Programa, se inicializan las variables a utilizar durante su ejecución*

*DECLARE*

*Var\_Num number;*

*Res number :=0;*

*Negative EXCEPTION;*

*BEGIN*

*Var\_num := '&Number';*

*--Se usa la condicional IF a manera de comprobar si el número ingresado es menor igual que 0, o en su defecto negativo*

*IF Var\_Num <=0 THEN*

*RAISE Negative;*

*END IF;*

*-- Se utiliza un ciclo FOR que inicia desde 1 hasta el valor que pueda tener el número ingresado, al cual se le calcula el Módulo*

*FOR i IN 1..Var\_Num LOOP*

*IF mod (Var\_Num, i)=0 THEN*

*Res:=Res+1;*

*END IF;*

END LOOP;

*--Se usa la condicional IF a manera de desplegar los dos posibles resultados de la operación efectuada*

IF Res = 2 THEN

DBMS\_OUTPUT.PUT\_LINE ('El número ingresado es un Número Primo');

ELSE

DBMS\_OUTPUT.PUT\_LINE ('El número ingresado NO es un Número Primo');

END IF;

*--Se maneja la excepción para imprimir en pantalla cuando el número ingresado es negativo o es menor igual que 0*

EXCEPTION

WHEN Negative THEN

DBMS\_OUTPUT.PUT\_LINE ('El número ingresado debe ser positivo, es decir mayor a 0');

END;

/

**Ejecución del Programa:**



```

SQL> --Inicio del Programa, se inicializan las variables a utilizar durante su ejecución
SQL> DECLARE
  2  Var_Num number;
  3  Res number :=0;
  4  Negative EXCEPTION;
  5
  6  BEGIN
  7  Var_num := '&Number';
  8
  9  --Se usa la condicional IF a manera de comprobar si el número ingresado es menor igual que 0, o en su defecto negativo
 10 IF Var_Num <=0 THEN
 11   RAISE Negative;
 12   END IF;
 13
 14 -- Se utiliza un ciclo FOR que inicia desde 1 hasta el valor que pueda tener el número ingresado, al cual se le calcula el Módulo
 15 FOR i IN 1..Var_Num LOOP
 16   IF mod (Var_Num, i)=0 THEN
 17     Res:=Res+1;
 18   END IF;
 19   END LOOP;
 20
 21 --Se usa la condicional IF a manera de desplegar los dos posibles resultados de la operación efectuada
 22 IF Res = 2 THEN
 23   DBMS_OUTPUT.PUT_LINE ('El número ingresado es un Número Primo');
 24 ELSE
 25   DBMS_OUTPUT.PUT_LINE ('El número ingresado NO es un Número Primo');
 26   END IF;
 27
 28 --Se maneja la excepción para imprimir en pantalla cuando el número ingresado es negativo o es menor igual que 0
 29 EXCEPTION
 30 WHEN Negative THEN
 31   DBMS_OUTPUT.PUT_LINE ('El número ingresado debe ser positivo, es decir mayor a 0');
 32 END;
 33 /

```

```

Enter value for number: 2
old   7: Var_num := '&Number';
new   7: Var_num := '2';
El número ingresado es un Número Primo

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.

#### **Código del Primer Programa (Bloque Anónimo #1):**

--Se crea la tabla Cumpleanos con los atributos id, nombre, fec\_cumple y se establece que su llave primaria es id

```
CREATE TABLE Cumpleanos (  
id number,  
nombre varchar2(15) NOT NULL,  
fec_cumple date NOT NULL,  
CONSTRAINT pk_Cumpleanos PRIMARY KEY (id)  
);
```

--Procedemos a la creación de las variables v\_nombre y v\_fec\_cumple con su respectivo tipo de datos

#### **DECLARE**

```
v_nombre cumpleanos.nombre%TYPE := '';  
v_fec_cumple cumpleanos.fec_cumple%TYPE;
```

#### **BEGIN**

-- Se genera un ciclo for de 5 repeticiones a través de la cual el usuario ingresará los datos en las tuplas de la tabla para las variables v\_nombre y v\_fec\_cumple

```
FOR counter in 1..5  
LOOP  
DBMS_OUTPUT.PUT_LINE('CONTADOR= ' || counter);  
IF counter = 1 THEN  
v_nombre := '&nombre';  
v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');  
INSERT INTO Cumpleanos VALUES (counter, v_nombre, v_fec_cumple);  
ELSIF counter = 2 THEN  
v_nombre := '&nombre';
```

```
        v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
        INSERT INTO Cumpleanos VALUES (counter, v_nombre, v_fec_cumple);
ELSIF counter = 3 THEN
    v_nombre := '&nombre';
    v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
    INSERT INTO Cumpleanos VALUES (counter, v_nombre, v_fec_cumple);
ELSIF counter = 4 THEN
    v_nombre := '&nombre';
    v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
    INSERT INTO Cumpleanos VALUES (counter, v_nombre, v_fec_cumple);
ELSIF counter = 5 THEN
    v_nombre := '&nombre';
    v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
    INSERT INTO Cumpleanos VALUES (counter, v_nombre, v_fec_cumple);
END IF;
END LOOP;
```

**END;**

/

## Ejecución del Primer Programa (Bloque Anónimo #1):

```
29         END IF;
30     end loop;
31
32 END;
33 /
Enter value for nombre: miguel
old 10:          v_nombre := '&nombre';
new 10:          v_nombre := 'miguel';
Enter value for fecha: 03/12/2001
old 11:          v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
new 11:          v_fec_cumple := TO_DATE('03/12/2001', 'dd/mm/yyyy');
Enter value for nombre: Juan
old 14:          v_nombre := '&nombre';
new 14:          v_nombre := 'Juan';
Enter value for fecha: 21/4/1998
old 15:          v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
new 15:          v_fec_cumple := TO_DATE('21/4/1998', 'dd/mm/yyyy');
Enter value for nombre: Pedro
old 18:          v_nombre := '&nombre';
new 18:          v_nombre := 'Pedro';
Enter value for fecha: 10/6/2002
old 19:          v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
new 19:          v_fec_cumple := TO_DATE('10/6/2002', 'dd/mm/yyyy');
Enter value for nombre: Diego
old 22:          v_nombre := '&nombre';
new 22:          v_nombre := 'Diego';
Enter value for fecha: 26/09/1990
old 23:          v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
new 23:          v_fec_cumple := TO_DATE('26/09/1990', 'dd/mm/yyyy');
Enter value for nombre: Fernando
old 26:          v_nombre := '&nombre';
new 26:          v_nombre := 'Fernando';
Enter value for fecha: 13/07/1999
old 27:          v_fec_cumple := TO_DATE('&fecha', 'dd/mm/yyyy');
new 27:          v_fec_cumple := TO_DATE('13/07/1999', 'dd/mm/yyyy');
CONTADOR= 1
CONTADOR= 2
CONTADOR= 3
CONTADOR= 4
CONTADOR= 5

PL/SQL procedure successfully completed.
```

## **Código del Segundo Programa (Bloque Anónimo #2):**

*--Inicio del Segundo Bloque Anónimo*

**SET SERVEROUTPUT ON;**

*--Se declaran las variables v\_idCumpleanos, v\_nombre, v\_fec\_cumple como parte del bloque anónimo*

**DECLARE**

*v\_idCumpleanos Cumpleanos.id%TYPE := 0;*  
*v\_nombre cumpleanos.nombre%TYPE := '';*  
*v\_fec\_cumple cumpleanos.fec\_cumple%TYPE := '';*

*--Se hace una búsqueda de los valores ingresados previamente en el primer bloque anónimo de los atributos nombre, fec\_cumple en las variables v\_nombre y v\_fec\_cumple*

**BEGIN**

*v\_idCumpleanos := '&id';*  
*SELECT nombre, fec\_cumple into v\_nombre, v\_fec\_cumple*  
*FROM Cumpleanos*  
*WHERE id = v\_idCumpleanos;*  
*--Se imprime en pantalla las variables v\_nombre y v\_fec\_cumple con los datos almacenados en sus diferentes tuplas*  
*DBMS\_OUTPUT.PUT\_LINE('Nombre del Cumpleaño: ' || v\_nombre || ' Día de Cumpleaños: ' || v\_fec\_cumple);*

*--Establecemos un manejo de excepciones para el caso en el que la variable v\_idCumpleanos no contenga ningún valor almacenado en alguna de sus tuplas*

**EXCEPTION**

*WHEN NO\_DATA\_FOUND THEN*  
*DBMS\_OUTPUT.PUT\_LINE('ID del cumpleaños no encontrado, intente con otro');*  
*END;*  
*/*

## **Ejecución del Segundo Programa (Bloque Anónimo #2):**

```

SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2     v_idCumpleanos Cumpleanos.id%TYPE := 0;
  3     v_nombre cumpleanos.nombre%TYPE := '';
  4     v_fec_cumple cumpleanos.fec_cumple%TYPE := '';
  5 BEGIN
  6     v_idCumpleanos := '&id';
  7     SELECT nombre, fec_cumple into v_nombre, v_fec_cumple
  8     FROM Cumpleanos
  9     WHERE id = v_idCumpleanos;
 10     DBMS_OUTPUT.PUT_LINE('Nombre del Cumplea ero: ' || v_nombre || ' Dia de Cumplea os: ' || v_
fec_cumple);
 11 exception
 12     when no_data_found then
 13         DBMS_OUTPUT.PUT_LINE('ID NO ENCONTRADO, INTENTE CON OTRO');
 14 END;
 15 /
Enter value for id: 6
old 6:     v_idCumpleanos := '&id';
new 6:     v_idCumpleanos := '6';
ID NO ENCONTRADO, INTENTE CON OTRO

PL/SQL procedure successfully completed.

```