
6. GESTIÓN DE EXCEPCIONES.

- 6.1.- Introducción.
- 6.2.- Excepciones internas de PL/SQL.
- 6.3.- Excepciones definidas por el usuario.
- 6.4.- Funciones SQLCODE y SQLERRM
- 6.5.- Propagación de excepciones.
- 6.6.- RAISE_APPLICATION_ERROR

6.1.- INTRODUCCIÓN

Las excepciones son unas características de PL y que ha heredado del lenguaje ADA, del cual proviene. El objetivo de las excepciones es el tratamiento de los errores que se producen en tiempo de ejecución y no de compilación.

Cuando se produce un error, se genera una excepción. Cuando esto sucede, el control pasa al gestor de excepciones, que es una sección independiente del programa. Una vez aquí no hay forma de volver a la sección ejecutable del bloque.

Las excepciones pueden ser:

- **Implícitas:** que son predefinidas y no predefinidas por el servidor Oracle
- **Explícitas:** que son las definidas por el usuario.

Las excepciones (de usuario) se declaran en la sección declarativa de un bloque, se generan en la sección ejecutable y se tratan en la sección de excepciones. Un bloque termina cuando PL provoca una excepción.

Declare

```
e_nombreExcep EXCEPTION; definidos por el usuario
```

BEGIN

```
---raise e_nombreExcep;
```

EXCEPTION

```
when exception_1 then
```

```
    ordenesPL --- ;
```

```
when e_nombreExcep then
```

```
    ordenesPL ;
```

```
When others then
```

```
    ---;
```

END

Donde OTHERS es la excepción que irrumpe en caso de error de Oracle no controlado por otras excepciones.

6.2.- EXCEPCIONES INTERNAS DE PL/SQL (PREDEFINIDAS POR EL SERVIDOR)

Este tipo de excepciones se disparan ante un determinado error detectado por PL/SQL. No hay que declararlas. Disponemos de las siguientes:

Excepciones	Descripción
NO_DATA_FOUND	Se produce cuando una orden del tipo SELECT INTO no ha devuelto ningún valor.
ZERO_DIVIDE	Sucede cuando se produce cuando se divide un valor por 0.
INVALID_CURSOR	Se produce cuando se produce una operación ilegal en un cursor.
TOO_MANY_ROWS	Se produce cuando una orden SELECT INTO devuelva más de una fila.
INVALID_NUMBER	Sucede cuando hay un fallo de conversión de una cadena a un número.
DUP_VAL_ON_INDEX	Se produce cuando se intenta introducir un duplicado en una tabla.
LOGIN_DENIED	Sucede cuando se introduce un nombre de usuario o contraseña no válido.
CURSOR_ALREADY_OPEN	Se produce cuando se intenta abrir un cursor que ya está abierto.
STORAGE_ERROR	Se produce cuando hay un fallo en memoria.
VALUE_ERROR	Se produce cuando hay un error de operación aritmética de conversión, de truncamiento o de restricción.
TIMEOUT_ON_RESOURCE	Sucede cuando se acaba el tiempo al coger determinados recursos.
ACCESS_INTO_NULL	Se produce cuando se intenta acceder a los atributos de un objeto no inicializado.
COLLECTION_IS_NULL	Se produce cuando se intenta acceder a elementos de una colección no inicializada.
NOT_LOGGED_ON	Se produce cuando se intenta acceder a la base de datos sin estar conectado a Oracle.
PROGRAM_ERROR	Sucede si hay un problema interno en la ejecución de un programa.
ROWTYPE_MISMATCH	Se produce cuando la variable HOST y la variable PL/SQL pertenecen a tipos incompatibles.
SUBSCRIPT_OUTSIDE_LIMIT	Sucede cuando se intenta acceder a una tabla o a un array con un valor de índice ilegal.

Hacer un bloque PL, utilizando excepciones internas, que introduciendo dos números por teclado divida N1 entre N2.

```
de Trabajo | Generador de Consultas
--
DECLARE
    NUM1 NUMBER:=&n1; NUM2 NUMBER:=&n2; DIVIDE NUMBER;
BEGIN
    DIVIDE:= NUM1/NUM2;
EXCEPTION
    WHEN ZERO_DIVIDE THEN
        DBMS_OUTPUT.PUT_LINE('IMPOSIBLE DIVIDIR POR 0');
END;
```

6.3.- EXCEPCIONES DEFINIDAS POR EL USUARIO.

Para poder utilizar o crear una excepción definida a nivel a usuario, lo primero que se debe hacer es declararla en la zona de declaraciones. Siendo su sintaxis en la zona de excepciones idéntica a las anteriores.

Para arrancar una excepción de usuario y pasarle el control a la sección de excepciones utilizaremos la orden RAISE seguida del nombre de la excepción.

Para su utilización hay que dar tres pasos:

1. Se deben declarar en la sección DECLARE de la siguiente forma:

```
nb_excepción EXCEPTION;
```

2. Se disparan o levantan en la sección ejecutable del programa con la orden RAISE.

```
RAISE nb_excepción;
```

3. Se tratan en la sección EXCEPTION según el formato ya conocido.

```
WHEN nb_excepción THEN tratamiento;
```

Ejemplo:

```
DECLARE
    v_salary emp.sal%type;
    e_emp_no_valido EXCEPTION;
    v_id number := &id;
BEGIN
    If v_id <= 0 then
        Raise e_emp_no_valido;
    End If;

    Select e.sal into v_salary from emp e where e.empno = v_id;
    DBMS_OUTPUT.PUT_LINE('Salario :'||v_salary);
EXCEPTION
    WHEN e_emp_no_valido THEN
        DBMS_OUTPUT.PUT_LINE('IDENTIFICADOR NO VALIDO');
END;
```

6.5.- FUNCIONES SQLCODE, SQLERRM

Se utilizan para saber que error Oracle dio lugar a la excepción dentro de una excepción others.

Funciones	Descripción										
SQLCODE	Devuelve el número de error de Oracle de las excepciones internas. Se le asigna una variable NUMBER. Los valores DECODE son los siguientes:										
	<table><tr><th>Valor</th><th>Descripción</th></tr><tr><td>0</td><td>No se encontró ninguna excepción.</td></tr><tr><td>1</td><td>Excepción definida por el usuario.</td></tr><tr><td>+100</td><td>Excepción NO_DATA_FOUND.</td></tr><tr><td>Nº negativo</td><td>Otro número de error del servidor.</td></tr></table>	Valor	Descripción	0	No se encontró ninguna excepción.	1	Excepción definida por el usuario.	+100	Excepción NO_DATA_FOUND.	Nº negativo	Otro número de error del servidor.
	Valor	Descripción									
	0	No se encontró ninguna excepción.									
	1	Excepción definida por el usuario.									
	+100	Excepción NO_DATA_FOUND.									
	Nº negativo	Otro número de error del servidor.									
SQLERRM	Devuelve datos carácter que contienen el mensaje de error asociado.										

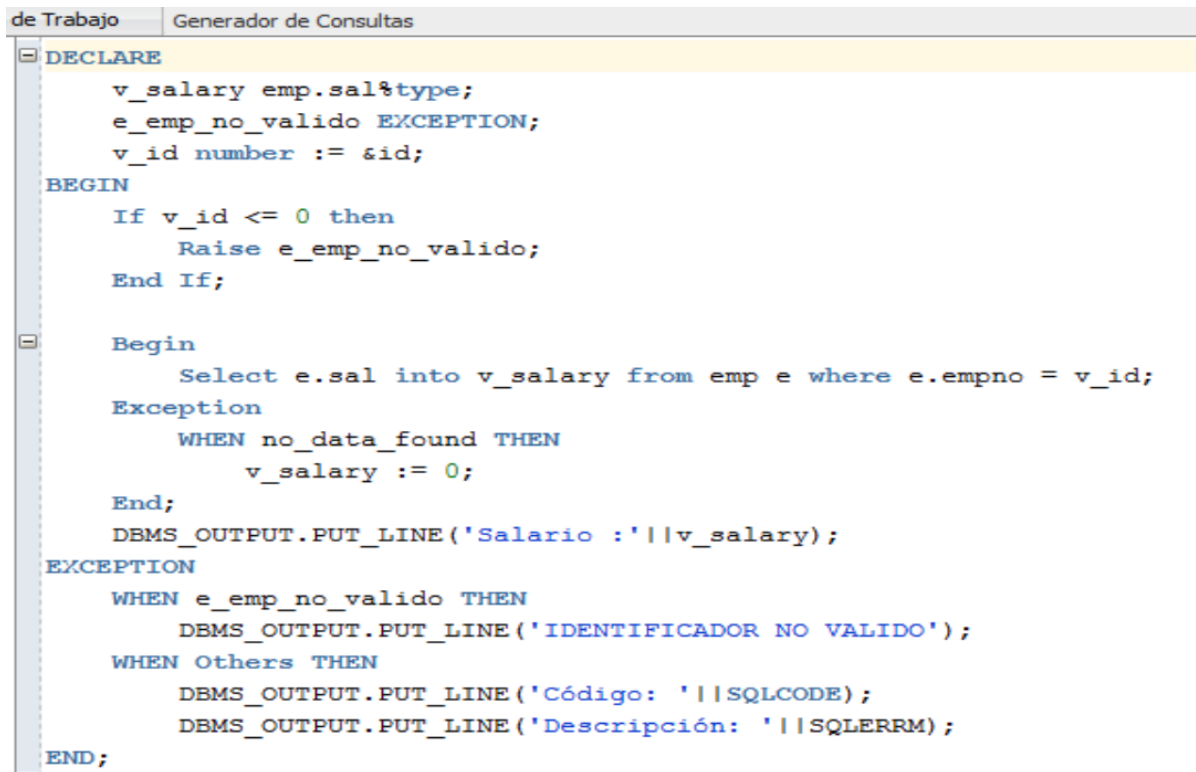
```
DECLARE
    v_salary emp.sal%type;
    e_emp_no_valido EXCEPTION;
    v_id number := &id;
BEGIN
    If v_id <= 0 then
        Raise e_emp_no_valido;
    End If;

    Select e.sal into v_salary from emp e where e.empno = v_id;
    DBMS_OUTPUT.PUT_LINE('Salario :'||v_salary);
EXCEPTION
    WHEN e_emp_no_valido THEN
        DBMS_OUTPUT.PUT_LINE('IDENTIFICADOR NO VALIDO');
    WHEN Others THEN
        DBMS_OUTPUT.PUT_LINE('Código: '||SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Descripción: '||SQLERRM);
END;
```

6.6.- PROPAGACIÓN DE EXCEPCIONES

El ámbito de las excepciones es igual que el de las variables.

Si una excepción se controla en un bloque, no se “propaga” al bloque superior:



```
de Trabajo  Generador de Consultas

DECLARE
    v_salary emp.sal%type;
    e_emp_no_valido EXCEPTION;
    v_id number := &id;
BEGIN
    If v_id <= 0 then
        Raise e_emp_no_valido;
    End If;

    Begin
        Select e.sal into v_salary from emp e where e.empno = v_id;
    Exception
        WHEN no_data_found THEN
            v_salary := 0;
    End;
    DBMS_OUTPUT.PUT_LINE('Salario :'||v_salary);
EXCEPTION
    WHEN e_emp_no_valido THEN
        DBMS_OUTPUT.PUT_LINE('IDENTIFICADOR NO VALIDO');
    WHEN Others THEN
        DBMS_OUTPUT.PUT_LINE('Código: '||SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Descripción: '||SQLERRM);
END;
```

Si una excepción se propaga fuera de su ámbito no podrá ser referenciada por su nombre. Para propagar un error definido por el usuario fuera del bloque lo mejor es definir la excepción dentro de un paquete para que continúe siendo visible fuera del bloque.

6.7.- RAISE APPLICATION ERROR

El paquete DBMS_STANDARD incluye un procedimiento muy útil llamado RAISE_APPLICATION_ERROR que sirve para crear nuestros propios mensajes de error.

Sintaxis:

<code>RAISE_APPLICATION_ERROR(-nº_error, mensaje_error);</code>

Donde número_error: es un número comprendido entre -20000 y -20999 y mensaje_error es una cadena de hasta 512 bytes.

Ejemplo:

```
PROCEDURE SUBIR_SUELDO
    (NUM_EMPL NUMBER, INCREMENTO NUMBER)
IS
    SALARIO_ACTUAL NUMBER;
BEGIN
    SELECT SALARIO INTO SALARIO_ACTUAL
    FROM EMPLEADOS
    WHERE EMP_NO = NUM_EMPL;
    IF SALARIO_ACTUAL IS NULL THEN
        RAISE_APPLICATION_ERROR(-20010, 'SALARIO NULO');
    ELSE
        UPDATE EMPLEADOS
        SET SUELDO = SALARIO_ACTUAL + INCREMENTO
        WHERE EMP_NO = NUM_EMPL;
    END IF;
END SUBIR_SUELDO;
```