Índice

1. INTRODUCCIÓN DE EXCEPCIONES 2
2. DECLARACIÓN DE EXCEPCIONES 2
3. MANEJO DE EXCEPCIONES 4
4. EJERCICIOS PROPUESTOS 10

1. INTRODUCCIÓN DE EXCEPCIONES

Cualquier programa bien escrito debe ser capaz de tratar los errores de manera inteligente y de recuperarse de ellos, si es posible. PL/SQL implementa los mecanismos de tratamiento de errores mediante excepciones.

El objetivo de las excepciones es el tratamiento de los errores que se producen en tiempo de ejecución y no en tiempo de compilación. Los errores que se producen en la fase de compilación son detectados por el motor PL/SQL y comunicados al usuario. El programa no puede tratar estos errores, dado que aún no ha sido ejecutado. Las excepciones y los gestores de excepciones son el método a través del cual el programa reacciona a los **errores de ejecución** y realiza su tratamiento.

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. Esto permite separar la gestión de errores del resto del programa, lo que hace que sea más fácil de entender la lógica de éste, y también asegura que todos los errores serán interceptados.

Tipos de errores PL/SQL		
Tipo de error	**Cuándo se informa de él**	**Como es tratado**
De compilación	Compilador PL/SQL	Interactivamente
De ejecución	Motor de ejecución PL/SQL	Pregramáticamente

2. DECLARACIÓN DE EXCEPCIONES

Las excepciones son declaradas en la sección de declaración, son lanzadas en la sección de ejecución, y resueltas en la sección de excepciones. Existen dos tipos de excepciones las definidas por los usuarios y las predefinidas.

Definidas por el usuario

Es un error cuya definición se realiza en el programa. El error en cuestión no tiene que ser, necesariamente, un error Oracle; podría tratarse de un error relativo a los datos. Las excepciones definidas por el usuario se declaran en la sección declarativa de un bloque PL/SQL. Al igual que las variables, las excepciones tienen un tipo asociado (EXCEPCTION) y un ámbito. La sintaxis es:

Nombre_excepcion **EXCEPTION**;

Excepciones predefinidas

Excepciones predefinidas de Oracle		
Error Oracle	**Excepción Equivalente**	**Descripción**
ORA - 0001	DUP_VAL_ON_INDEX	Violación de una restricción de unidad
ORA - 0051	TIMEOUT_ON_RESOURCE	Se produjo un fin de intervalo mientras se esperaba un cierto recurso
ORA - 0061	TRANSACTION_BACKED_OUT	La transacción fue cancelada debido a un bloqueo
ORA - 1001	INVALID_CURSOR	Operación ilegal con un cursor
ORA - 1012	NOT_LOGGED_ON	No existe conexión con Oracle
ORA - 1017	LOGIN_DENIED	Nombre de usuario o contraseña inválidos
ORA - 1403	NO_DATA_FOUND	No se ha encontrado ningún dato
ORA - 1422	TOO_MANY_ROWS	Hay más de una línea que corresponde a una orden SELECT… INTO
ORA - 1476	ZERO_DIVIDE	División por cero
ORA - 1722	INVALID_NUMBER	Falló la conversión a un número
ORA - 6500	STORAGE_ERROR	Error interno PL/SQL, generado cuando PL/SQL se queda sin memoria
ORA - 6501	PROGRAM_ERROR	Error interno PL/SQL
ORA - 6502	VALUE_ERROR	Error de truncamiento aritmético o de conversión
ORA - 6504	ROWTYPE_MISMATCH	Una variable de cursor del HOST y una variable del cursor PL/SQL tienen tipo de fila incompatibles
ORA - 6511	CURSOR_ALREADY_OPEN	Se ha intentado abrir un cursor que ya estaba abierto
ORA - 6530	ACCESS_INTO_NULL	Se ha intentado asignar valores a los atributos de un objeto que tiene el valor NULL
ORA - 6531	COLLECTION_IS_NULL	Se ha intentado aplicar métodos de colección distintos de EXISTS a una tabla o array PL/SQL con valor NULL
ORA - 6532	SUBSCRIPT_OUTSIDE_LIMIT	Una referencia a una tabla anidada o índice de arrray se encuentra fuera del rango declarado
ORA - 6533	SUBSCRIPT_BEYOND_COUNT	Una referencia a una tabla animada o índice de array es mayor que el número de elementos de la colección.

3. MANEJO DE LAS EXCEPCIONES

Una vez que se haya lanzado una excepción, el manejador de errores pasa a la sección EXCEPTION. Aquí se utiliza la siguiente estructura:

```
EXCEPTION

        WHEN   nombre_excepción1   THEN
                        Conjunto de sentencias 1

        WHEN   nombre_excepción2   THEN
                        Conjunto de sentencias 2
        . . .

        WHEN   OTHERS   THEN
                        Conjunto de sentencias

END;
```

Cada excepción tendrá un **when … then** y un conjunto de sentencias que intentarán subsanar el error acaecido. Además, podemos añadir dos o más excepciones a un mismo conjunto de sentencias.

```
…
WHEN   EXCEPTION  OR  EXCEPTION  THEN

                Conjunto de sentencias
…
```

En la parte **when … other … then** contiene las sentencias que se ejecutarán al ocurrir un error que no tiene excepción declarada.

EXCEPCIONES DEFINIDAS POR EL USUARIO

Las excepciones definidas por el usuario se usan para tratar condiciones de error definidas por el programador.

Para su utilización hay que seguir tres pasos:

1.-Se declaran en la sección **DECLARE** de la forma siguiente:

<nombreexcepcion> **EXCEPTION**;

2.- se disparan en la sección ejecutable del programa con la orden **RAISE**:

RAISE <nombreexcepcion>

3.- se trata en la sección **EXCEPTION** según el formato conocido

When <nombreexcepcion> **then** <tratamiento>;

La instrucción RAISE se puede utilizar varias veces en el mismo bloque con la misma o con distintas excepciones, pro solo puede haber un manejador when para cada excepción.

Ejemplo.

Crear un procedimiento denominado **subir_salario** en el que se pasa como parámetros el numero de un empleado y cuanto se le va a incrementar su sueldo y se actualiza su salario en la tabla **employee**. Utilizar dos excepciones. La excepción **NO_DATA_FOUND** para el caso de introducir un empleado que no esté en la tabla **employee** y otra definida por el usuario denominada **salario_nulo** que no permita subir salario a aquellos que tienen en el campo salary un valor null.

```
create or replace procedure subir_salario
        (num_empleado integer,
        incremento real)
is
        salario_actual real;
        salario_nulo exception;
begin
        select salary into salario_actual
        from employee
        where employee_id=num_empleado;
        if salario_actual is null
           then
             raise salario_nulo;
        end if;
        update employee
        set salary= salary+ incremento
        where employee_id=num_empleado;
        exception
        when no_data_found then
            dbms_output.put_line('error,el empleado numero: '||num_empleado||' no existe');
        when salario_nulo then
           dbms_output.put_line('error el empleado numero: '||num_empleado||' tiene salario nulo');
end subir_salario;
/

execute subir_salario(2089,300)
```

OTRAS EXCEPCIONES

Existen otros errores internos de oracle que no tienen asignada una excepción. No obstante, generan un código de error y un mensaje de error, a los que se accede mediante las funciones **SQLCODE** y **SQLERRM.**

SQLCODE	.- devuelve el código de error de la excepción
SQLERRM	.- muestra el mensaje de error.

Cuando se produce uno de estos errores se transfiere el control a la sección EXCEPTION, donde se tratará el error en la cláusula WHEN OTHERS

Ejemplo

Crear un procedimiento denominado buscar_codigo_dep al que se le pasa el nombre de un departamento y visualiza el codigo de dicho departamento. En el caso de que haya varios departamentos con el mismo nombre dará un mensaje de error hay varios departamentos con este nombre. Y en el caso de que se produzca otro error no ha de salir el codigo del error y el error cometido.

```
create or replace procedure busca_codigo_dep (nombre varchar2) is
        codigo department.department_id%type;
begin
         select department_id into codigo
         from department
        where name = nombre;
         dbms_output.put_line ('el departamento de nombre: '||nombre||' tiene codigo: '||codigo);
        exception
         when TOO_MANY_ROWS then
                dbms_output.put_line ('Coinciden en nombre varios departamentos');
         when OTHERS then
                dbms_output.put_line (to_char(SQLCODE) || '    ' || SQLERRM);
end;
```

```
SQL> execute busca_codigo_dep ('SALES');
Coinciden en nombre varios departamentos

PL/SQL procedure successfully completed.

SQL> execute busca_codigo_dep ('HOLA');
100     ORA-01403: no se ha encontrado ningún dato (no data found)

PL/SQL procedure successfully completed.

SQL> execute busca_codigo_dep ('ACCOUNTING');
El departamento de nombre ACCOUNTING tiene código 10
```

RAISE_APPLICATION_ERROR.- Se usa esta sentencia para crear nuestros propios mensajes de error

```
RAISE_APPLICATION_ERROR (número, mensaje de error);
```

Donde *número* debe estar en el rango comprendido ente -20.000 y -20.999
El *mensaje* no debe ser de más de 512 caracteres

Ejemplo de utilización de raise_application_error

Crear un procedimiento denominado subir_sueldo por el que se pasan dos parámetros número de empleado e incremento y se le sube el sueldo a dicho empleado de la tabla employee siempre que este no sea nulo.

```
create or replace procedure subir_sueldo
        (num_emple integer, incremento real)
is
        salario_actual real;
begin
        select salary into salario_actual
        from employee
        where employee_id=num_emple;
        if salario_actual is null
           then
             raise_application_error(-20010,'el usuario '||num_emple||' tiene salario nulo');
          else
             update employee
             set salary=salario_actual+incremento
             where employee_id=num_emple;
        end if;
end subir_sueldo;

execute subir_sueldo(7919,300)

select * from employee
```

4. EJERCICIOS PROPUESTOS

1º.- Desarrollar un procedimiento que permita insertar nuevos departamentos según las siguientes especificaciones:

- Se pasará al procedimiento el nombre del departamento y la localidad.
- El procedimiento insertará la fila nueva asignando como número de departamento la decena siguiente al número mayor de la tabla.
- Se incluirá gestión de posibles errores.

2º.- Escribir un procedimiento que reciba todos los datos de un nuevo empleado procese la transacción de alta, gestionando los posibles errores.

3º.- Codificar un procedimiento reciba como parámetros un numero de departamento, un importe y un porcentaje; y suba el salario a todos los empleados del departamento indicado en la llamada. La subida será el porcentaje o el importe indicado en la llamada (el que sea más beneficioso para el empleado en cada caso empleado).

4º.- Escribir un procedimiento que suba el sueldo de todos los empleados que ganen menos que el salario medio de su oficio. La subida será del 50% de la diferencia entre el salario del empleado y la media de su oficio. Se deberá asegurar que la transacción no se quede a medias, y se gestionarán los posibles errores.

5º.- Borrar un departamento que no tenga empleados atendiendo a las siguientes premisas:

- si el departamento no existe lanzar una excepción.
- si el departamento tiene empleados asociados también.