Sesión 10

MANEJO DE EXCEPCIONES

En la mayoría de los lenguajes de programación se emplean EXCEPCIONES para tratar errores en tiempo de ejecución. En Oracle también podemos definir excepciones. Estas definiciones hacen que, cuando se produce un error, salte una excepción y pase el control a la sección excepción correspondiente. Las acciones a seguir se incluyen en la sección EXCEPTION definida al final del bloque BEGIN-END.

```
BEGIN

  …

EXCEPTION
   WHEN <nombre_excepción> THEN
      <bloque de sentencias>;
  WHEN <nombre_excepción> THEN
      <bloque de sentencias>;

   ...
   [WHEN OTHERS THEN <bloque de sentencias>;]
END;
```

Existen dos tipos de excepciones:
las excepciones predefinidas y las excepciones definidas por el usuario.

EXCEPCIONES PREDEFINIDAS

Son aquellas que se disparan automáticamente al producirse determinados errores. Las más frecuentes son:

- **too_many_rows**: Se produce cuando SELECT … INTO devuelve más de una fila.
- **no_data_found**: se produce cuando un SELECT… INTO no devuelve ninguna fila.
- **value_error**: se produce cuando hay un error aritmético o de conversión.
- **zero_divide**: se puede cuando hay una división entre 0.
- **dupval_on_index**: se crea cuando se intenta almacenar un valor que crearía duplicados en la clave primaria o en una columna con restricción UNIQUE.
- **invalid_number**: se produce cuando se intenta convertir una cadena a un valor numérico.

Al estar ya definidas en Oracle no se declaran en la sección DECLARE.

EXCEPCIONES DEFINIDAS POR EL USUARIO

Son las que define el usuario. Para poder trabajar con excepciones definidas por el usuario, es necesario que se realicen 3 pasos:

1. Definición del nombre de la excepción en la sección DECLARE. La sintaxis es

   Nombre_de_excepción EXCEPTION

2. Lanzar la excepción. Se hace mediante la orden RAISE ;
3. Definir las acciones a seguir si se lanza la excepción. Esto se hace en la sección EXCEPTION.

   WHEN nombre_de_excepción THEN …

Ejemplo 1:

Crea un procedimiento en la que dado un nif de un empleado se muestre su nombre y teléfono.

```
create or replace PROCEDURE  mostrar_empleado(elnif empleado.nif%type) is
        auxnombre empleado.nombre%type;
        auxtelefono empleado.telefono%type;
BEGIN
  select nombre, telefono into auxnombre, auxtelefono
  from empleado where nif=elnif;
  escribir('Nombre '||auxnombre||'   telefono '||auxtelefono);

  EXCEPTION
      WHEN no_data_found then
        escribir('No existe empleado con ese nif');
END;
```

Ejemplo 2:

Crea una función en la que dado un número de habitación devuelva su categoría.

```
create or replace FUNCTION tipo_habitacion
 (elnumero habitacion.numero%type) RETURN habitacion.categoria%type
is
auxcategoria habitacion.categoria%type;
begin
select categoria into auxcategoria
from habitacion
where numero=elnumero;
return(auxcategoria);
exception
   when no_data_found then
       escribir('No existe ninguna habitación con ese número');
      return null;
end;
```

Ejemplo 3:

Crea un procedimiento en el que dado un nif de un empleado y un valor X (número). Incluya a ese empleado como recepcionista siempre que no existan ya más de X recepcionistas.

Si al darlo de alta como recepcionista vemos que no está dado de alta como empleado, se dará de alta tanto de empleado como de recepcionista y se mostrará un mensaje indicando que falta por completar sus datos.

```
create or replace PROCEDURE
                pon_recepcion(elnif empleado.nif%type, maximo number) is
    auxnombre empleado.nombre%type;
    auxtelefono empleado.telefono%type;
    total number(3);
    pasa_tope exception;

BEGIN
 select count(*) into total from emprecepcion;
 if total>maximo then raise pasa_tope; end if;
 select nombre, telefono into auxnombre, auxtelefono
 from empleado where nif=elnif;
 escribir('Nombre '||auxnombre||'  pasa a tareas de recepcionista');
 insert into emprecepcion values(elnif);

EXCEPTION
   WHEN no_data_found then
      insert into empleado(nif) values(elnif);
      insert into emprecepcion values(elnif);
      escribir('No existía. Se ha dado de alta como recepcionista, falta
            completar  sus datos en empleado');
   WHEN pasa_tope then
     escribir('Se supera el tope asignado para recepcionistas');

END;
```

FUNCIONES PARA IDENTIFICAR EXCEPCIONES

SQLCODE

Devuelve el código numérico del error producido

SQLERRM

Devuelve el mensaje asociado al error que se ha producido. Es varchar.

Ejemplo:

```
CREATE PROCEDURE …
.
.
.
EXCEPTION
WHEN NO_DATA_FOUND THEN
escribir(' No se encuentra información para los datos de entrada')
WHEN OTHERS
escribir('Se ha producido error ' || SQLCODE||' consiste en ' ||SQLERRM);
END;
```

Os recordamos que **raise_application_error** sirve para generar un error mostrando un mensaje.