ÍNDICE

Contenido

Antolín·Muñoz·Chaparro¶
Jefe·de·Proyecto·Sistemas·Informáticos·¶
División·III·de·Aplicaciones·de·Costes·de·Personal·y·Pensiones·Públicas¶
Oficina·de·Informática·Presupuestaria·(DIR3:·EA0027952)¶
Intervención·General·de·la·Administración·del·Estado¶

Antolín·Muñoz·Chaparro¶
Jefe·de·Proyecto·Sistemas·Informáticos·¶
División·III·de·Aplicaciones·de·Costes·de·Personal·y·Pensiones·Públicas¶
Oficina·de·Informática·Presupuestaria·(DIR3:·EA0027952)¶
Intervención·General·de·la·Administración·del·Estado¶

TRATAMIENTO DE ERRORES EN PL/SQL

PL/SQL implementa mecanismos para el control de errores mediante 2 métodos:

- Excepciones.
- Gestores de excepciones.

Las excepciones se asocian a:

- Errores de Oracle.
- Errores definidos por el usuario.

Los errores tratados en este capítulo son los que se producen en tiempo de ejecución.

Los errores de compilación se muestran durante la fase de compilación pero no se pueden tratar.

Cuando se produce un error, se genera lo que se llama una "EXCEPCION".

Cuando se lanza un error de tipo EXCEPTION se pasa el control al gestor de excepciones. Es la sección de un bloque que se denomina EXCEPTION, en la cual se ejecutarán las operaciones necesarias para controlar ese error.

Sintaxis:

```
BEGIN
.........
EXCEPTION
      /* Gestión de excepciones */
END;
```

Antolín·Muñoz·Chaparro¶
Jefe·de·Proyecto·Sistemas·Informáticos·¶
División·III·de·Aplicaciones·de·Costes·de·Personal·y·Pensiones·Públicas¶
Oficina·de·Informática·Presupuestaria·(DIR3:·EA0027952)¶
Intervención·General·de·la·Administración·del·Estado¶

Declaración de excepciones

Toda excepción ha de seguir un proceso de declaración, generación y tratamiento:

- La declaración se realiza dentro de la sección DECLARE de un bloque.
- La generación se producen en la sección BEGIN…END.
- El tratamiento se realiza en la sección EXCEPTION.

Excepciones definidas por el usuario

Son errores que se definen dentro del programa en la parte declarativa DECLARE de un bloque.

No tienen por qué ser errores de ORACLE.

Sintaxis:

```
<nombre_excepcion> EXCEPTION;
```

Ejemplo:

```
DECLARE
      Demasiados_estudiantes     EXCEPTION;
```

Excepciones predefinidas

En este apartado se definen las excepciones que se han definido internamente junto con el lenguaje PL/SQL y que tienen asignado un nombre concreto con el que se pueden referenciar.

Código Error	Excepción	Descripción
ORA-0001	DUP_VAL_ON_INDEX	Violación de una restricción de unicidad.
ORA-0051	TIMEOUT_ON_RESOURCE	Se produjo un fin de intervalo mientras se esperaba un cierto recurso.
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 fila que corresponde a una orden SELECT..INTO.
ORA-1476	ZERO_DIVIDE	División por cero.
ORA-1722	INVALID_NUMBER	Falló la conversión a un número; por ejemplo, 'IA' no es válido.
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 cursos del host y una variable de cursor PL/SQL tienen tipos 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 varray PL/SQL con valor NULL.
ORA-6532	SUBSCRIPT_OUTSIDE_LIMIT	Una referencia a una tabla anidada o índice de varray se encuentra fuera del rango declarado (ej: -1).
ORA-6533	SUBSCRIPT_BEYOND_COUNT	Una referencia a una tabla anidada o índice de varray es mayor que el número de elementos de la colección.
ORA-6592	CASE_NOT_FOUND	Se produce cuando la cláusula CASE…END, recibe un valor que no está contemplado en las condiciones WHEN, y además no se ha indicado la cláusula ELSE.
ORA-6548	NO_DATA_NEEDED	Excepción que se provoca cuando se manejan tablas PIPELINED. Si se incluye a una excepción OTHER en un bloque que incluye una sentencia PIPE ROW. Si el código que alimenta una sentencia PIPE ROW no va seguido de un procedimiento de limpieza.
ORA-1410	SYS_INVALID_ROW	Ocurre cuando la conversión de un valor ROWID en formato texto a formato universal ROWID, falla porque el resultado de la conversión no representa una fila ROWID válida.

PROVOCAR EXCEPCIONES

Las excepciones normalmente se generan automáticamente como consecuencia de un error en la ejecución del código, pero también es posible forzar que una excepción definida por el usuario se provoque.

La sintaxis es la siguiente:

```
RAISE nombre_excepcion;
```

Ejemplo

```
DECLARE
      Demasiados_estu       EXCEPTION;
      V_registro            estudiantes%ROWTYPE;
      Cuenta                NUMBER;
BEGIN
      SELECT COUNT(*) INTO cuenta FROM estudiantes
      WHERE depto = 'VENTAS';

      SELECT * INTO V_registro FROM estudiantes
      WHERE depto = 'VENTAS';

      IF cuenta > 1 THEN
            RAISE demasiados_estu;
      END IF;
EXCEPTION
      WHEN demasiados_estu THEN
            INSERT INTO temporal VALUES …;
      WHEN OTHERS
            INSERT INTO  otroerror VALUES …;
END;
```

SINTAXIS DE LA SECCIÓN EXCEPTION

Esta sección es la encargada de gestionar los errores que se produce en la sección ejecutable de un bloque.

La sintaxis es la siguiente:

```
EXCEPTION
      WHEN nombre_excepcion THEN
            Ordenes;
      WHEN OTHERS THEN
            Ordenes;
END;
```

La cláusula WHEN nos permite gestionar las excepciones predefinidas y las del usuario, permitiendo realizar operaciones cuando las detecte.

El gestor OTHERS controla todas aquellas excepciones generadas y no tratadas con anterioridad. Este gestor debe aparecer en último lugar y se activa cuando los gestores anteriores no satisfacen la excepción que se ha provocado en cuyo caso se activa el gestor OTHERS para resolverlo.

Ejemplo

```
DECLARE
      V_registro            estudiantes%ROWTYPE;
BEGIN
      SELECT * INTO v_registro
      FROM estudiantes
      WHERE depto = 'VENTAS';
EXCEPTION
      WHEN NO_DATA_FOUND OR TOO_MANY_ROWS THEN
            INSERT INTO temporal ……;
      WHEN OTHERS
            INSERT INTO otroerror……;
END;
```

USO DE SQLCODE Y SQLERRM

Son variables predefinidas por el lenguaje PL/SQL que tratan el código y el texto del error que se produce en una excepción y su significado es el siguiente:

- SQLCODE: Almacena el código del error en formato NUMBER.
- SQLERRM: Almacena el texto del error en formato VARCHAR2(512).

Ejemplo

```
DECLARE
      V_error          NUMBER;
BEGIN
      ……
EXCEPTION
      WHEN OTHERS
            V_error := SQLCODE;
            V_texto := SQLERRM;
END;
```

UTILIZACIÓN DE RAISE_APPLICATION_ERROR

Permite al usuario crear y almacenar sus propios mensajes de error que luego podrá mostrarlos en pantalla.

La sintaxis es la siguiente:

```
RAISE_APPLICATION_ERROR(numero_error, mensaje
                              [,preserver_errores]);
```

Donde:

- NUMERO_ERROR: Puede ser un valor entre -20.000 y -20.999.
- MENSAJE: Puede ser un texto de un máximo de 512 caracteres.
- PRESERVAR_ERRORES: Si se indica TRUE para este parámetro, el error se añade a la lista. Si se indica FALSE (que es el valor predeterminado), el error sustituye a toda la lista de errores anteriormente almacenados por el usuario.

Ejemplo

```
BEGIN
      ……
      IF valor > 50
            RAISE_APPLICATION_ERROR(-2000, 'Hay un error');
      END IF;
EXCEPTION
      ……
END;
```

UTILIZACIÓN DE EXCEPTION_INIT

Permite realizar una asociación entre los errores definidos por el usuario con un código de error de Oracle.

La sintaxis es la siguiente:

```
PRAGMA EXCEPTION_INIT(nombre_excepcion,
                      numero_error_Oracle);
```

Ejemplo

```
DECLARE
      V_error          VARCHAR2(512);
      E_sindicatos     EXCEPTION;
      PRAGMA EXCEPTION_INIT (e_sindicatos, -1400);
BEGIN
      INSERT INTO estudiantes (id) VALUES (null);
EXCEPTION
      WHEN e_sindicatos THEN
            BEGIN
                  V_error := SQLERRM;
                  INSERT INTO log_table(cod, message)
                  VALUES (-1400, v_error);
            END;
END;
```

PROPAGACIÓN DE EXCEPCIONES

Hasta el momento se han planteado en este capítulo los errores que ocurrían en la sección ejecutable, pero ¿Qué ocurre con errores que se producen en otras secciones? ¿Cómo se tratan?

Un error al poderse producir en cualquiera de las secciones de un bloque ha de estudiarse pues, que ocurre en cada una de ellas de manera individual.

Tratamiento de errores en la sección ejecutable

Si se produce un error en dicha sección, Oracle realiza las siguientes operaciones:

1. Busca la sección EXCEPTION del bloque actual. Si la encuentra realiza las acciones pertinentes y devuelve el control al bloque superior.

2. Si no existe sección EXCEPTION en el bloque actual, se propaga el error al bloque superior y se vuelve a realizar la operación anterior.

EJEMPLO

```
DECLARE
      A           EXCEPTION;
BEGIN
      ……
      BEGIN
            RAISE A;          -- provocamos el error.
      EXCEPTION
            WHEN A THEN       -- tratamos el error.
                  ……
      END;
      ……                      -- se devuelve el control.
```

EJEMPLO

```
DECLARE
      A           EXCEPTION;
      B           EXCEPTION;
BEGIN
      ……
      BEGIN
            RAISE B;          -- provocamos el error.
      EXCEPTION
            WHEN A THEN       -- no podemos tratar el error.
                  ……
      END;
      ……                      -- se devuelve el control.
EXCEPTION
      WHEN B THEN
                  ……          -- se propaga al bloque
                              -- superior y se trata.
END;
```

EJEMPLO

```
DECLARE
      A           EXCEPTION;
      B           EXCEPTION;
      C           EXCEPTION;
BEGIN
      ……
      BEGIN
            RAISE C;          -- provocamos el error.
      EXCEPTION
            WHEN A THEN       -- no podemos tratar el error.
                  ……
      END;
      ……                      -- se devuelve el control.
EXCEPTION
      WHEN B THEN
                  ……          -- No se trata y se propaga al
                              -- bloque superior.
END;                          -- Se propaga al final del
                              -- programa y produce un error.
```

Tratamiento de errores en la sección declarativa

Si se produce un error en dicha sección, Oracle realiza las siguientes operaciones:

1. Propaga el error al bloque superior si es que existe.

2. Trata el error en la sección EXCEPTION del bloque superior al que se ha producido el error, si es que existe.

EJEMPLO

```
BEGIN
      ……
      DECLARE
            V_number   NUMBER(3) := 'ABC';
            /* provocamos un error de tipo value_error */
      BEGIN
            ……
      EXCEPTION
            WHEN OTHERS THEN
                  ……
                  /* En este bloque no se trata el error y
                     se propaga al bloque superior */
      END;
EXCEPTION
      WHEN OTHERS THEN      -- se trata el error.
            ……
END;
```

Antolín·Muñoz·Chaparro¶
Jefe·de·Proyecto·Sistemas·Informáticos·¶
División·III·de·Aplicaciones·de·Costes·de·Personal·y·Pensiones·Públicas¶
Oficina·de·Informática·Presupuestaria·(DIR3:·EA0027952)¶
Intervención·General·de·la·Administración·del·Estado¶

Tratamiento de errores en la sección de excepciones

Si se produce un error en dicha sección, Oracle realiza las siguientes operaciones:

1. Propaga el error al bloque superior si es que existe.

2. Trata el error en la sección EXCEPTION del bloque superior al que se ha producido el error, si es que existe.

EJEMPLO

```
DECLARE
      A     EXCEPTION;
      B     EXCEPTION;
BEGIN
      RAISE A;              -- se produce un primer error.
EXCEPTION
      WHEN A THEN           -- se trata el primer error
            RAISE B         -- se produce un segundo error
      WHEN B THEN
            ……
            /* El segundo error no se trata aquí y se
               propaga al bloque superior.
               Como no hay bloque superior se aborta la
               ejecución del programa con un error */
END;
```

EJEMPLO

```
BEGIN
      DECLARE
            A     EXCEPTION;
            B     EXCEPTION;
      BEGIN
            RAISE A;          -- se produce un primer error.
      EXCEPTION
            WHEN A THEN       -- se trata el primer error
                  RAISE B     -- se produce un segundo error
            WHEN B THEN
                  ……
                  /* No se trata el error y se propaga al
                     al bloque superior */
      END;
EXCEPTION
      WHEN OTHERS THEN        -- se trata el segundo error
            ……
END;
```

Antolín·Muñoz·Chaparro¶
Jefe·de·Proyecto·Sistemas·Informáticos·¶
División·III·de·Aplicaciones·de·Costes·de·Personal·y·Pensiones·Públicas¶
Oficina·de·Informática·Presupuestaria·(DIR3:·EA0027952)¶
Intervención·General·de·la·Administración·del·Estado¶

Ámbito de las excepciones

El ámbito de las excepciones es igual que el de las variables. Si una excepción definida por el usuario se propaga fuera de su ámbito, no podrá ser referenciada por su nombre. Para poderlo remediar se utilizan paquetes.

Por ejemplo:

```
CREATE OR REPLACE PACKAGE global
     B          EXCEPTION;
END;

SQL> DECLARE
           ……
           BEGIN
                RAISE global.B;
           END;
     EXCEPTION
           WHEN global.B THEN
                ……
     END;
```