Capítulo 5
PL/SQL en objetos de la base de datos

1. Introducción

Además de los bloques PL/SQL anónimos utilizados por SQL*Plus o por las herramientas de desarrollo (Oracle*Forms, Oracle*Reports...), se puede emplear código PL/SQL en determinados objetos de la base de datos, como los procedimientos almacenados (PROCEDURE, FUNCTION, PACKAGE) y los triggers de base de datos.

2. Los triggers de bases de datos

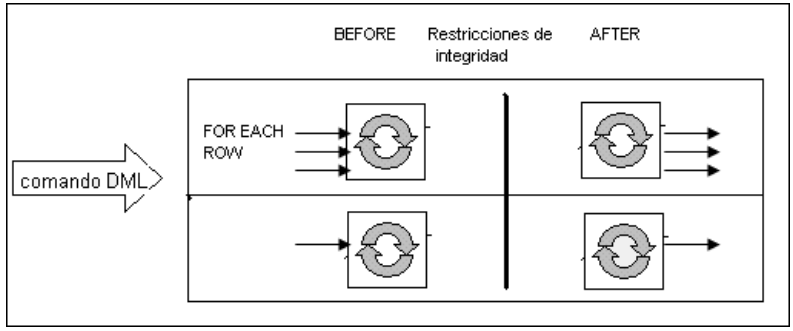
Un trigger es un bloque PL/SQL asociado a una tabla. Este bloque se ejecutará cuando se aplique a la tabla una instrucción DML (INSERT, UPDATE, DELETE).

El bloque PL/SQL que constituye el trigger puede ejecutarse antes o después de la actualización y, por lo tanto, antes o despues de la verificación de las restricciones de integridad.

Los triggers ofrecen una solución procedimental para definir restricciones complejas o que tengan en cuenta datos procedentes de varias filas o de varias tablas, como por ejemplo para garantizar el hecho de que un cliente no pueda tener más de dos pedidos no pagados. Sin embargo, los triggers no deben emplearse cuando sea posible establecer una restricción de integridad.

En efecto, las restricciones de integridad se definen en el nivel de tabla y forman parte de la estructura de la propia tabla, por lo que la verificación de estas restricciones es mucho más rápida. Además, las restricciones de integridad garantizan que todas las filas de las tablas respetan dichas restricciones, mientras que los triggers no tienen en cuenta los datos ya contenidos en la tabla en el momento de definirlos.

El bloque PL/SQL asociado a un trigger se puede ejecutar para cada fila afectada por la instrucción DML (opción FOR EACH ROW), o una única vez para cada instrucción DML ejecutada (opción predeterminada).



Ejecución antes o después de la comprobación de las restricciones de integridad para cada fila o cada sentencia

En los triggers BEFORE y FOR EACH ROW se pueden modificar los datos que van a insertarse en la tabla, de modo que respeten las restricciones de integridad. También es posible ejecutar consultas de tipo SELECT sobre la tabla a la que se aplica la instrucción DML, aunque únicamente en el marco de un trigger BEFORE INSERT. Todas estas operaciones no se pueden realizar en los triggers AFTER, ya que después de la verificación de las restricciones de integridad no es posible modificar los datos y, dado que la modificación (adición o eliminación) de la fila no se ha terminado, no es posible ejecutar consultas de tipo SELECT sobre la tabla.

También se pueden incluir triggers en las vistas (VIEW) con el fin de capturar las instrucciones DML que se pueden ejecutar sobre ellas. Estos triggers permiten controlar todas las operaciones realizadas sobre las vistas y, para el usuario final, la vista es completamente similar a una tabla, ya que puede realizar sobre ella operaciones INSERT, UPDATE y DELETE. Estos triggers son de tipo INSTEAD OF, es decir, que su ejecución va a reemplazar a la instrucción DML a la que estén asociados. Este tipo de trigger solo puede definirse en vistas y es el único tipo de trigger que puede implementarse en ellas.

Principio de funcionamiento de los triggers instead of

Sintaxis

```
CREATE [OR REPLACE] TRIGGER nombre_trigger
{BEFORE/AFTER/INSTEAD OF}
{INSERT/UPDATE[OF col,...]/DELETE}
ON Nombre_tabla [FOR EACH ROW]
[FOLLOWS nombre_otro_trigger[,...]]
[ENABLE/DISABLE]
[WHEN (condition)]
Bloc PL/SQLv
```

`OR REPLACE`

Reemplaza la descripción del trigger si ya existe.

`BEFORE`

El bloque PL/SQL se ejecuta antes de la verificación de las restricciones de tabla y de actualizar los datos almacenados en la misma.

`AFTER`

El bloque PL/SQL se ejecuta después de la actualización de los datos contenidos en la tabla.

`INSTEAD OF`

El bloque PL/SQL siguiente reemplaza el procesamiento estándar asociado a la instrucción que ha activado al trigger (solo por una vista).

`INSERT/UPDATE [OF col,...]/DELETE`

Instrucción asociada a la activación del trigger. Varias instrucciones pueden activar un mismo trigger y se combinan mediante el operador OR.

`FOR EACH ROW`

El trigger se ejecuta para cada fila tratada por la instrucción asociada.

`FOLLOWS nombre_otro_trigger[,...]`

Oracle permite definir varios triggers para la misma tabla y el mismo evento. En este caso, el orden relativo de ejecución de estos triggers es indeterminado. Si el orden de ejecución de estos triggers es importante en su aplicación, puede utilizar la cláusula FOLLOWS disponible a partir de la versión 11. Esta cláusula permite indicar que el trigger debe ejecutarse después de los triggers enumerados.

`ENABLE/DISABLE`

Esta cláusula permite indicar si el trigger está o no activo desde el momento de su creación; por defecto, un trigger de nueva creación está activo. Crear un trigger desactivado permite verificar que se compila correctamente antes de ponerlo realmente en servicio. Un trigger creado desactivado puede activarse más adelante utilizando una sentencia ALTER TRIGGER...ENABLE.

`WHEN (condición)`

La condición especificada debe cumplirse para que se ejecute el código.

Los datos de la tabla a la que está asociado el trigger son inaccesibles desde las instrucciones del bloque. Solo la fila que se está modificando es accesible a través de dos variables de tipo RECORD: OLD y NEW, las cuales poseen la estructura de la tabla o de la vista asociada. Estas variables pueden utilizarse en la cláusula WHEN del trigger o en el bloque de instrucciones. En este último caso se referencian como variables host mediante el prefijo ":" (:OLD.nombre_campo, :NEW.nombre_campo).

La palabra OLD permite conocer qué fila se va a eliminar en un trigger DELETE o la fila que se va a modificar en un trigger UPDATE. La palabra NEW permite conocer cuál es la nueva fila insertada en un trigger INSERT o la fila tras su modificación en un trigger UPDATE.

Los nombres OLD y NEW están definidos de manera predeterminada, aunque es posible utilizar otros nombres empleando la cláusula `REFERENCING OLD AS nuevo_nombre NEW AS nuevo_nombre`. Esta cláusula se incluye justo antes de la cláusula FOR EACH ROW (si existe) en la definición del trigger.

Ejemplo

Ejecución de un bloque PL/SQL antes de una eliminación en la tabla CLIENTES por parte del usuario FLORENCIO:

```
CREATE TRIGGER antes_elim_cli
      BEFORE    DELETE
      ON FLORENCIO.CLIENTES
      DECLARE
         ...
      BEGIN
         ...
      END;
```

Ejecución de un bloque PL/SQL después de actualizar cada fila de la tabla ARTICULOS cuando el precio antiguo es mayor que el nuevo:

```
create or replace trigger post_actprecio
  after update of precio
  on articulos
  for each row
  when (old.precio > new.precio)
declare
  ...
begin
  ...
end;
```

Para cada pedido, se desea conocer el nombre del usuario de Oracle que lo ha introducido. La primera etapa consiste en añadir una nueva columna a la tabla de pedidos. Esta columna debe aceptar el valor NULL ya que, para las filas de los pedidos existentes, el nombre del usuario de Oracle es desconocido.

Modificación de la tabla PEDIDOS:

```
SQL> alter table pedidos
  2    add (usuario varchar2(30));

Tabla modificada.

SQL>
```

En el trigger se cambia el nombre de la nueva fila, y el trigger se ejecuta antes de la verificación de las restricciones de integridad para cada fila insertada en la tabla de pedidos.

Definición del trigger:

```
SQL> create or replace trigger bf_ins_pedidos
  2     before insert
  3     on pedidos
  4     referencing new as nuevo
  5     for each row
  6  begin
  7     select user into :nuevo.usuario from dual;
  8  end;
  9  /

Trigger creado.

SQL>
```

Se desea saber el número de pedidos introducido por el usuario de Oracle. Para evitar escribir una consulta que recorra la tabla de pedidos completa, operación que puede resultar muy pesada, el trigger se limita a actualizar una tabla de estadísticas.