TEMA 8. BASES DE DATOS ACTIVAS

Las reglas que especifican acciones que son activadas de forma automática por determinados eventos han sido considerados en las bases de datos, principalmente para la definición de restricciones (constrains). El modelo que se ha implementado para especificar las reglas de las bases de datos activas se denomina Evento-Condición-Acción (ECA). Estas reglas se denominan disparadores (triggers), siendo su finalidad convertir una base de datos pasiva en activa.

Un evento son operaciones de actualización de la base de datos que se aplican explícitamente a la base de datos (podrían considerarse los eventos temporales, una consideración del tiempo en la activación de la regla).

Una condición es una declaración que determina si la regla debe o no ejecutarse. Una vez que se ha producido el evento activador, puede evaluarse una condición opcional. Si no se especifica una condición, la acción se ejecuta una vez que se produzca el evento. Si se especifica una condición, primero se evalúa ésta y, si su evaluación es verdadera, se ejecuta la acción asociada a la regla.

Una acción es una operación que se realiza cuando se evalúa afirmativamente la regla. Suele ser un procedimiento SQL, aunque puede ser una función de otro lenguaje, programa externo, …

Las características de los disparadores son:

- Son reglas simples.
- No consideran un orden de evaluación.
- No incluyen conocimiento.
- No son escalables.
- Un número excesivo genera confusión.
- Pueden generarse ciclos de ejecución.
- Los eventos están limitados a insert, update y delete.
- Se pueden definir 2 niveles: filas y sentencias.
- No soportan la consideración del tiempo.
- Sobrecargan al gestor de bases de datos.
- Se ejecutan antes, después o en lugar de un evento.
- No son estándar para diferentes DBMS.
- No existe conocimiento externo de su ejecución.
- Complejidad para diseñar y verificar la consistencia.
- Pueden invocar a otros procedimientos y disparar otros triggers, pero no admiten parámetros y no pueden ser invocados desde otros procedimientos.

Los tipos de disparadores son: auto-generados (son generados de forma automática por el gestor de bases de datos cuando se define el esquema) o generados por el DBA (correspondientes a las reglas de negocio).

Los disparadores son usados:

- Cuando los datos de una tabla son generados desde otro tipo de procedimientos y se necesita controlar los valores que toman algunos campos determinados de la tabla en cuestión.
- Para duplicar los contenidos de una tabla automáticamente y en tiempo real.
- Para implementar complejas restricciones sobre los valores que pueden tomar los campos en una tabla Oracle.
- Para controlar las modificaciones de los valores de los campos de una tabla (auditorías).
- Para incrementar automáticamente los valores de un campo.
- Para realizar actualizaciones de una tabla en cascada.
- Para modificar campos o registros de una tabla que un usuario no puede modificar directamente.

Las restricciones de los disparadores son:

- Un disparador no puede emitir ninguna orden de control de transacciones. El disparador se activa como parte de la ejecución de la orden que provocó el disparo, y forma parte de la misma transacción que dicha orden.
- Ningún procedimiento o función llamado por el disparador puede emitir órdenes de control de transacciones.
- El cuerpo de un disparador no puede contener ninguna declaración de variables long o long raw.
- Existen restricciones acerca de a qué tablas puede acceder el cuerpo de un disparador. Dependiendo del tipo de disparador y de las restricciones que afecten a las tablas.
- INSTEAD OF es una cláusula válida sólo para vistas. Si una vista tiene un disparador INSTEAD OF, cualquier vista creada sobre ésta debe tener a su vez un disparador INSTEAD OF.

-- Formato general de los Disparadores en Oracle 10g

```
create [or replace] trigger [schema .] trigger
{ [FOLLOWS nombre-otro-trigger] }
{ before | after | instead of }
{ dml_event_clause
         | { ddl_event [or ddl_event]...
         | database_event [or database_event]…}
on { [schema .] schema | database }
[referencing_clause] [for each row | for each statement]

[when ( condition ) ]
{ pl/sql_block | call_procedure_statement }

-- La sintaxis depende del tipo de trigger

{ delete | insert | update [of column [, column]...] }
[or { delete | insert | update [of column [, column]...] }]...
on { [schema .] table | [nested table nested_table_column of]
[schema .] view }
```

-- Habilitando y deshabilitando triggers

```
alter trigger BOOKSHELF_BEF_UPD_INS_ROW enable;

alter table BOOKSHELF enable all triggers;

alter trigger BOOKSHELF_BEF_UPD_INS_ROW disable;

alter table BOOKSHELF disable all triggers;

-- Borrando triggers
drop trigger BOOKSHELF_BEF_UPD_INS_ROW;
```