Eventos en MySQL. Implementación de los Triggers.

Implementación de los triggers en MySQL

Introducción a los disparadores de MySQL

Resumen: en este tutorial, aprenderá sobre la implementación de los triggers MySQL. Además, le mostraremos cómo MySQL almacena los disparadores y las limitaciones de los disparadores en MySQL.
En MySQL, un trigger es un conjunto de instrucciones SQL que se invoca automáticamente cuando se realiza un cambio en los datos de la tabla asociada. Se puede definir un trigger para que se invoque antes o después de que los datos se cambien mediante la instrucción INSERT, UPDATE o DELETE. Antes de MySQL versión 5.7.2, puede definir un máximo de seis disparadores para cada tabla.

- BEFORE INSERT: se activa antes de que los datos se inserten en la tabla.
- AFTER INSERT: se activa después de insertar los datos en la tabla.
- BEFORE UPDATE: se activa antes de que se actualicen los datos de la tabla.
- AFTER UPDATE: se activa después de actualizar los datos de la tabla.
- BEFORE DELETE: se activa antes de que se eliminen los datos de la tabla.
- AFTER DELETE: se activa después de eliminar los datos de la tabla.

Sin embargo, desde MySQL versión 5.7.2+, puede definir múltiples disparadores para el mismo evento de disparo y tiempo de acción.
Cuando utiliza una sentencia que no utilice las instrucciones INSERT, DELETE o UPDATE para cambiar datos en una tabla, no se invocan los disparadores asociados con la tabla. Por ejemplo, la instrucción TRUNCATE elimina todos los datos de una tabla pero no invoca el trigger asociado con esa tabla.
Hay algunas sentencias que usan la orden INSERT detrás de escena, como la declaración REPLACE o la declaración LOAD DATA. Si usa estas declaraciones, se invocan los disparadores correspondientes asociados con la tabla.
Debe usar un nombre único para cada disparador asociado con una tabla. Sin embargo, puede tener el mismo nombre de trigger definido para diferentes tablas.
Debe nombrar los disparadores utilizando la siguiente convención de nomenclatura:

```
(BEFORE|AFTER)_tableName_(INSERT|UPDATE|ELIMINAR)
```

Por ejemplo, before_order_update es un trigger invocado antes de que se actualice una fila en la tabla de pedidos.
La siguiente convención de nomenclatura es tan buena como la anterior.

```
nombre_tabla_(BEFORE|AFTER)_(INSERT|UPDATE|ELIMINAR)
```

Por ejemplo, order_before_update es el mismo que el disparador before_order_update anterior.

Almacenamiento de disparador en MySQL

MySQL almacena los disparadores en un directorio de datos, por ejemplo, *data/classicmodels* con los archivos llamados tablename.TRG y triggername.TRN:

- El archivo tablename.TRG asigna el trigger a la tabla correspondiente.
- El archivo triggername.TRN contiene la definición del trigger.

Puede hacer una copia de seguridad de los disparadores de MySQL copiando los archivos de los disparadores a la carpeta de copia de seguridad. También puede hacer una copia de seguridad de los disparadores utilizando la herramienta mysqldump.

Limitaciones de los disparadores de MySQL

Los triggeres de MySQL cubren todas las características definidas en el SQL estándar. Sin embargo, hay algunas limitaciones que debe conocer antes de usarlas en sus aplicaciones.
Los desencadenadores de MySQL no pueden:

- Utilice las instrucciones SHOW, LOAD DATA, LOAD TABLE, BACKUP DATABASE, RESTORE, FLUSH y RETURN.
- Usar declaraciones que confirmen o reviertan implícita o explícitamente, como COMMIT, ROLLBACK, START TRANSACTON, LOCK/UNLOCK TABLES, ALTER, CREATE, DROP, RENAME.
- Usar declaraciones preparadas como PREPARE y EXECUTE.
- Usar sentencias SQL dinámicas.

Desde MySQL versión 5.1.4, un disparador puede llamar a un procedimiento almacenado o función almacenada, lo que era una limitación en las versiones anteriores.