Laboratorio de Bases de Datos (EBB)

Unidad V – Procedimientos almacenados/triggers

Departamento de Electricidad, Electrónica y Computación
Facultad de Ciencias Exactas y Tecnología
Universidad Nacional de Tucumán

Primer Cuatrimestre 2024

Contenido

Procedimientos almacenados

- Definición
- Creación
- Recomendaciones
- Modificación y borrado

Definición

Procedimiento Almacenado (Stored Procedure):

- Colección de sentencias SQL, con un nombre, almacenadas en el servidor dentro de la BD

Finalidad:

- Encapsular tareas repetitivas

Definición

Características:

- Soportan declaración de variables, ejecución condicional y otras características de programación
- Aceptan parámetros de entrada y devuelven valores
- Devuelven un valor de estado que indica éxito o falla
- Pueden llamar a otros procedimientos almacenados

Definición

Características:

- Encapsulan la lógica de negocio permitiendo su reusabilidad
- Ocultan la complejidad subyacente de la BD
- Proveen mecanismos de seguridad
- Reducen el tráfico en la red

Definición

Características:

- Pueden mantener la integridad de datos y reforzar políticas de la BD
- Si se modifican no se necesita hacer un despliegue
- Facilitan los cambios en entornos variados
- Resulta difícil su refactorización

Definición

Características:

- Se requiere de código externo a la BD para hacerles pruebas unitarias
- A veces no mejoran el rendimiento
- Reducen la vulnerabilidad (ejemplo: SQL injection)

Definición

Ejemplo de SQL injection:

1. Se lee de un formulario un campo de texto llamado IDUsuario:

```
txtIDUsuario = getRequestString("IDUsuario");
```

2. Se arma la sentencia SQL:

```
txtSQL="SELECT * FROM Usuarios
    WHERE IDUsuario=" + txtIDUsuario;
```

Definición

Ejemplo de SQL injection:

3. Si se ingresa “105 or 1=1” en el campo de texto, la sentencia queda:

```
SELECT * FROM Usuarios
    WHERE IDUsuario = 105 or 1=1;
```

4. Como 1=1 siempre es True, se muestra todo el contenido de la tabla Usuarios

Definición

Plan de ejecución:

- Es el resultado del cálculo realizado por el optimizador de consultas para determinar el modo más eficiente de implementar una consulta

- Para ver un plan de ejecución, además de la interfaz gráfica, se puede usar:

```
EXPLAIN FORMAT = TREE <Consulta>
```

Visualización de un plan de ejecución

Definición

Procesos que se realizan cuando se ejecuta una consulta:

- Análisis sintáctico (parsing)
- Normalización
- Compilación
- Optimización

Definición

Análisis sintáctico (parsing):

- Controla que la consulta esté bien escrita (bien formada)
- No se controlan los nombres de los objetos o columnas
- La salida de este proceso se llama “árbol de consulta”

Definición

Normalización:

- Verifica los nombres de objetos y columnas, los tipos de datos, la presencia de funciones de agrupamiento, el uso de alias o sinónimos que se deban resolver, etc

- La salida de este proceso es la entrada del proceso de compilación

Definición

Compilación:

- Construye planes de ejecución teniendo en cuenta la salida del proceso anterior

- La salida de este proceso es la entrada del proceso de optimización

Definición

Optimización:

- Evalúa los distintos planes de ejecución y elige el de menor costo (el plan que ejecutará la consulta más rápidamente usando la menor cantidad de recursos)

- Para la elección del plan tiene en cuenta las estadísticas, los operadores en la cláusula WHERE, el uso de JOIN, UNION, GROUP BY y ORDER BY

Definición

Optimización:

- Para determinar el mejor plan, el optimizador utiliza las estadísticas y recorre los distintos planes, prueba distintas formas de implementar los JOINs, modifica el orden de los JOINs, prueba distintos índices, etc

- A cada paso de un plan le asigna un número, que representa el tiempo estimado para ejecutarlo. La suma de estos tiempos es el tiempo estimado (costo) de un plan

Definición

Optimización:

- Una vez elegido el plan de ejecución, el mismo se guarda en el cache de procedimientos

- Con el plan de ejecución elegido y guardado, se ejecuta la consulta (según el plan)

Definición

Primera ejecución de un procedimiento:

- Se arma su plan de ejecución y se lo guarda en el cache de procedimientos

Siguientes ejecuciones:

- Reutilizan el plan de ejecución que ya está en memoria (evitan tener que volver a crearlo)

Definición

Permanencia de un plan de ejecución en memoria:

- Depende del tiempo requerido para su recompilación (costo)

- Depende de su frecuencia de uso (edad)

Creación

Consideraciones:

- Se emplea la sentencia CREATE PROCEDURE
- Hay que asignarle un nombre (único dentro de la BD)
- Su definición (cuerpo) está comprendida en un bloque delimitado por las sentencias BEGIN y END

Creación

Consideraciones:

- Cada una de las sentencias que forman el cuerpo del procedimiento termina en el delimitador de fin de línea predeterminado de MySQL (;)

- Para que se pueda pasar toda la definición del procedimiento a MySQL, se debe redefinir temporalmente el delimitador predeterminado

Creación

Ejecución:

- Se emplea la sentencia CALL seguida del nombre del procedimiento y sus parámetros

- Si el procedimiento no lleva parámetros, también se lo puede llamar sin los mismos

Ejecución de un procedimiento almacenado

Creación

Anidamiento:

- Los procedimientos pueden anidarse (un procedimiento llama a otro)

- El nivel de anidamiento es limitado

- Si Proc1 llama a Proc2, Proc2 puede acceder a todos los objetos definidos localmente en Proc1

Anidamiento de un procedimiento almacenado

Creación

Información:

- Para ver información sobre un procedimiento:

  SHOW CREATE PROCEDURE <nombre>

- Para obtener una lista de procedimientos:

  SHOW PROCEDURE STATUS

Información sobre los procedimientos almacenados

Recomendaciones

- Realizar un procedimiento por cada tarea (cohesión)

- Crear, probar y corregir los procedimientos en el servidor, luego hacer la prueba en los clientes

Modificación y borrado

ALTER PROCEDURE:

- Sólo permite modificar algunas características de un procedimiento (por ejemplo, el comentario), no su cuerpo ni parámetros (MySQL)

- Sólo se modifica un procedimiento (el externo)

- Se necesita tener determinados permisos

Modificación y borrado

DROP PROCEDURE:

- Permite borrar un procedimiento:

  DROP PROCEDURE [IF EXISTS] <nombre>

Parametrización

- Parámetros
- Manejo de errores

Parámetros

- **Entrada IN (por defecto)**: quien llame al procedimiento debe proporcionarlos. El procedimiento trabaja con una copia de los mismos
- **Salida OUT**: el valor de los parámetros se puede cambiar en el procedimiento. El procedimiento no puede acceder al valor inicial de este parámetro cuando empieza
- **Entrada/Salida INOUT**: combinación de las 2 opciones anteriores

Parámetros

Consideraciones:

- Tienen un ámbito local al procedimiento
- En el cuerpo del procedimiento también se pueden definir variables (son locales al mismo)
  - Las variables tienen precedencia sobre los parámetros o columnas de una tabla
  - Los parámetros tienen precedencia sobre las columnas de una tabla
  - Una variable en un bloque interno tiene precedencia sobre otra en un bloque externo

Parámetros

Parámetros de entrada:

- Verificar al principio del procedimiento los valores de entrada para detectar posibles valores inválidos

- Los valores de los parámetros se pasan en el mismo orden en el que fueron definidos en el procedimiento

- El máximo número de parámetros depende del SGBDR

Procedimiento con parámetros de entrada

Parámetros

Parámetros de salida:

- Se emplea la cláusula OUT

Procedimiento con parámetros de salida

Manejo de errores

- Al momento de ejecutar un procedimiento, por ejemplo para hacer un alta, se quiere también tener información sobre el resultado de la operación

- Una posible solución es emplear un parámetro de salida

Mensajes de error con parámetros de salida

Manejo de errores

- La implementación mostrada funciona, pero tiene algunas limitaciones:
  - Si se llama al procedimiento desde otro programa, el mismo devolverá éxito (o por lo menos no generará un error) aún cuando no se pueda ejecutar la operación
  - Quien llama al procedimiento, deberá examinar la variable de salida (el procedimiento se diseñó para que dependa de quien lo llama)

Manejo de errores

- Cuando se llama a un procedimiento se recibe un código con la indicación sobre el éxito o fracaso de la llamada, llamado **SQLSTATE**

- Este error se puede devolver a quien llame al procedimiento o lo puede gestionar el propio procedimiento

- Para que un procedimiento devuelva este código se emplea la sentencia SIGNAL

Manejo de errores

- Una forma de usar la sentencia SIGNAL es:

  SIGNAL SQLSTATE <código_SQLSTATE>

Manejo de errores

- Se pueden definir códigos SQLSTATE propios (cumpliendo ciertas reglas) o usar los existentes:

Clases		
Valor	**Clase**	**Significado**
00	S	Exito
01	W	Advertencia
02	N	Sin datos
Todas las otras clases	X	Excepción

Manejo de errores

- Para definir un código SQLSTATE propio, el mismo no debe empezar con '00' ya que este valor está reservado para indicar éxito y no es válido para indicar un error

- Si se especifica un valor inválido, se genera un error

- El valor '45000' es un valor genérico (significa excepción definida por el usuario no tratada)

Manejo de errores

- Para brindar más información sobre el error, se emplea además la cláusula SET:

SIGNAL SQLSTATE <código_sqlstate>
[SET <item_de_información> = <valor1>,
<item_de_información> = <valor2>,
...]

Manejo de errores

- Otra posibilidad para SIGNAL es no especificar un valor de SQLSTATE, sino el nombre de una condición:

  SIGNAL <condición>

- Para definir la condición se emplea la sentencia DECLARE CONDITION con un valor de SQLSTATE:

  DECLARE <condición> CONDITION FOR <valor>

Triggers

- Definición
- Administración

Definición

Trigger:

- Tipo de procedimiento almacenado que se ejecuta automáticamente al modificarse (inserción, borrado, modificación) los datos de la tabla sobre la que se lo creó

- En MySQL no se puede incluir cualquier sentencia SQL dentro de un trigger

Definición

Características:

- Se usan para mantener la integridad de datos
- No devuelven valores
- Su principal ventaja es que manejan complejidad de procesamiento

Definición

Características:

- Si sobre la columna de una tabla hay una restricción, un trigger BEFORE y uno AFTER, primero se ejecuta el trigger BEFORE, luego se evalúa la restricción (si pasa el trigger) y finalmente el trigger AFTER (si pasa la restricción)

- En MySQL se pueden disparar antes o después de la ejecución de sentencias INSERT, UPDATE o DELETE

Definición

Características:

- Una tabla puede tener múltiples triggers del mismo tipo, sin orden de ejecución

- Los dueños de tablas deben tener permisos para crear, modificar y borrar triggers

- No se pueden crear triggers en vistas ni en tablas temporales, pero sí las pueden referenciar

Definición

Características:

- No admiten parámetros

Administración

Los triggers se crean con la sentencia CREATE TRIGGER:

- Nombre del trigger
- Momento en que se dispara el trigger (BEFORE o AFTER)
- Evento que dispara el trigger (INSERT, UPDATE o DELETE)
- Tabla sobre la que se dispara el trigger
- Cuerpo del trigger

Administración

En una tabla se pueden definir varios triggers para el mismo evento y momento de disparo:

- Por defecto, los triggers se disparan según el orden en que fueron creados

- Para cambiar el orden, se pueden emplear las cláusulas FOLLOWS o PRECEDES especificando el trigger con el mismo evento y momento de disparo

Administración

Sobre el momento en que se dispara un trigger:

- **BEFORE**: se dispara antes de modificar la fila (al intentar modificar la fila, al margen que la fila luego se pueda modificar o no). Si falla, no se ejecuta la operación sobre la fila correspondiente

- **AFTER**: se dispara después de modificar la fila (se dispara sólo si se ejecutaron todos los triggers BEFORE y la operación sobre la fila correspondiente)

Administración

Sobre la ejecución de un trigger:

- Si se produce un error durante la ejecución de un trigger BEFORE o AFTER, implica que falla toda la sentencia que disparó el trigger

- En tablas transaccionales (InnoDB) la falla de una sentencia implica un rollback de todos los cambios realizados por la misma. La falla de un trigger hace que falle la sentencia que lo disparó, por lo que también genera un rollback

Administración

Consideraciones:

- Se requieren determinados permisos
- Algunas sentencias no soportadas dentro de un trigger:
  - Todas las sentencias CREATE, DROP y ALTER
  - GRANT y REVOKE
  - START TRANSACTION, COMMIT o ROLLBACK
  - RETURN
  - CALL

Administración

Trigger para inserción:

- Se dispara al insertar filas a la tabla en la que está creado
- Cuando se dispara agrega filas a una tabla llamada NEW (insensible a mayúsculas/minúsculas)
- La tabla NEW mantiene una copia de las filas que se agregaron
- Sólo se pueden referenciar las filas de la tabla NEW desde el trigger

Administración

Problema:

- Se quiere llevar la auditoría de los trabajos que se crean, de manera tal que cada vez que se cree un trabajo nuevo se registre la información necesaria

Solución:

- Crear una tabla de auditoría con los datos necesarios que se quieren auditar

- Definir un trigger de inserción en la tabla Trabajos para auditar la creación de un nuevo trabajo

Administración

Funcionamiento del trigger para inserción:

Trabajos					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...

Administración

Funcionamiento del trigger para inserción:

- Se hace una inserción en la tabla Trabajos:

Trabajos					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...
74	**Sis...**	**6**	**2019-03-15**	**2019-03-28**	**NULL**

Administración

Funcionamiento del trigger para inserción:

- Se dispara el trigger registrando la inserción en NEW:

NEW					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
74	Sis...	6	2019-03-15	2019-03-28	NULL

Administración

Funcionamiento del trigger para inserción:

- El trigger puede referenciar las filas de NEW:

Administración

Trigger para borrado:

- Se dispara al borrar filas de la tabla en la que está creado
- Cuando se dispara agrega filas a una tabla llamada OLD (insensible a mayúsculas/minúsculas)
- La tabla OLD mantiene una copia de las filas que se borraron
- Se pueden referenciar las filas de la tabla OLD desde el trigger

Administración

Problema:

- Se quiere llevar la auditoría de los trabajos que se borran, de manera tal que cada vez que se borre un trabajo se registre la información necesaria

Solución:

- Crear una tabla de auditoría con los datos necesarios que se quieren auditar

- Definir un trigger de borrado en la tabla Trabajos para auditar el borrado de un trabajo

Administración

Funcionamiento del trigger para borrado:

Trabajos					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...
74	Sis...	6	2019-03-15	2019-03-28	NULL

Administración

Funcionamiento del trigger para borrado:

- Se borra una fila en la tabla Trabajos:

Trabajos					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...
~~74~~	~~Sis...~~	~~6~~	~~2019-03-15~~	~~2019-03-28~~	~~NULL~~

Administración

Funcionamiento del trigger para borrado:

- Se dispara el trigger registrando el borrado en OLD:

OLD					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
74	Sis...	6	2019-03-15	2019-03-28	NULL

Administración

Funcionamiento del trigger para borrado:

- El trigger puede referenciar las filas de OLD:

Administración

Problema:

- Se quiere evitar que se borren filas de la tabla AuditoriaTrabajos

Solución:

- Definir un trigger de borrado en la tabla AuditoriaTrabajos para que se dispare al intentar borrar una fila en esta tabla

Administración

Problema:

- Probar el funcionamiento de 2 triggers de borrado sobre una tabla

Administración

Trigger para modificación:

- Se dispara al modificar filas en la tabla en la que está creado
- Funciona como un borrado e inserción, por lo que las filas originales se mueven a la tabla OLD, y las filas modificadas se insertan a la tabla NEW
- Se pueden referenciar las filas de estas tablas desde el trigger

Administración

Problema:

- Se quiere llevar la auditoría de los trabajos que se modifican, de forma tal que cada vez que se modifique un trabajo se registre la información necesaria

Solución:

- Crear una tabla de auditoría con los datos necesarios que se quieren auditar

- Definir un trigger de modificación en la tabla Trabajos para auditar la modificación de un trabajo

Administración

Funcionamiento del trigger para modificación:

Trabajos					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...
74	Sis...	6	2019-03-15	2019-03-28	NULL

Administración

Funcionamiento del trigger para modificación:

- Se modifica una fila en la tabla Trabajos:

Trabajos					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...
74	Sis...	6	2019-03-15	2019-03-28	**2019-12-15**

Administración

Funcionamiento del trigger para modificación:

- El trigger puede referenciar las filas de NEW y de OLD:

OLD					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
74	Sis...	6	2019-03-15	2019-03-28	NULL

NEW					
IDTrabajo	**Titulo**	**Duracion**	**FechaPresentacion**	**FechaAprobacion**	**FechaFinalizacion**
74	Sis...	6	2019-03-15	2019-03-28	2019-12-15

Administración

Consideraciones de rendimiento:

- Los triggers se ejecutan rápidamente porque las tablas NEW y OLD están en el cache

- El tiempo de ejecución está determinado por:
  - Número de tablas referenciadas
  - Número de filas afectadas

Administración

Borrado de un trigger:

- Se emplea la sentencia DROP TRIGGER

  DROP TRIGGER [IF EXISTS] <nombre>

- Cuando se borra una tabla, se borran todos sus triggers

- Sólo el dueño de una tabla tiene permiso para borrar un trigger de la misma

Administración

- Información sobre triggers:

  SELECT * FROM information_schema.TRIGGERS

Información sobre triggers

Resumen

- Definición de procedimiento almacenado
- Creación, modificación y borrado
- Recomendaciones
- Procedimientos almacenados con parámetros de entrada y salida
- Manejo de errores
- Definición de trigger
- Creación, modificación y borrado
- Ejemplos

Otros recursos

Plan de ejecución:

- https://www.simple-talk.com/sql/performance/execution-plan-basics/

Explain:

- http://www.sitepoint.com/using-explain-to-write-better-mysql-queries/

Otros recursos

Optimización:

- https://stackoverflow.com/questions/226859/disadvantage-of-stored-procedures
- http://www.spoiledtechie.com/post/Whats-up-with-Stored-Procedures-these-days.aspx
- https://www.seguetech.com/advantages-and-drawbacks-of-using-stored-procedures-for-processing-data/
- https://stackoverflow.com/questions/15142/what-are-the-pros-and-cons-to-keeping-sql-in-stored-procs-versus-code

Otros recursos

Manejo de errores:

- http://www.devshed.com/c/a/mysql/using-the-signal-statement-for-error-handling/
- https://www.tutorialspoint.com/How-can-we-use-SIGNAL-statement-with-MySQL-triggers
- http://www.brenkoweb.com/tutorials/mysql/mysql-advanced/raising-error-conditions-with-signal-resignal-statements
- https://www.databasejournal.com/features/mysql/mysql-error-handling-using-the-signal-and-resignal-statements.html

Otros recursos

Manejo de errores:

- http://www.mysqltutorial.org/mysql-signal-resignal/

Otros recursos

Triggers:

- https://dev.mysql.com/doc/refman/5.6/en/declare-condition.html
- https://dev.mysql.com/doc/refman/5.6/en/signal.html
- https://stackoverflow.com/questions/6634093/is-possible-to-do-a-rollback-in-a-mysql-trigger
- https://dba.stackexchange.com/questions/168191/mysql-rollback-and-triggers
- https://en.wikipedia.org/wiki/SQLSTATE