Técnico en

DESARROLLO DE SOFTWARE

Bases de Datos II

Bases de Datos II

Unidad IV

Procedimientos almacenados

1. Procedimientos almacenados en MySQL

Los procedimientos almacenados son nuevas funcionalidades de la versión de MySQL 5.0. Un procedimiento almacenado es un conjunto de comandos SQL que pueden almacenarse en el servidor. Una vez que se hace, los clientes no necesitan relanzar los comandos individuales, pero pueden en su lugar referirse al procedimiento almacenado.

Algunas situaciones en que los procedimientos almacenados pueden ser particularmente útiles: Cuando múltiples aplicaciones cliente se escriben en distintos lenguajes o funcionan en distintas plataformas, pero necesitan realizar la misma operación en la base de datos. Cuando la seguridad es muy importante. Los bancos, por ejemplo, usan procedimientos almacenados para todas las operaciones comunes. Esto proporciona un entorno seguro y consistente, y los procedimientos pueden asegurar que cada operación se loguea apropiadamente. En tal entorno, las aplicaciones y los usuarios no obtendrían ningún acceso directo a las tablas de la base de datos, sólo pueden ejecutar algunos procedimientos almacenados.

Los procedimientos almacenados pueden mejorar el rendimiento ya que se necesita enviar menos información entre el servidor y el cliente. El intercambio que hay es que

aumenta la carga del servidor de la base de datos ya que la mayoría del trabajo se realiza en la parte del servidor y no en el cliente. Considere esto, si muchas máquinas cliente (como servidores Web) se sirven a sólo uno o pocos servidores de bases de datos.

Los procedimientos almacenados le permiten tener bibliotecas o funciones en el servidor de base de datos. Esta característica es compartida por los lenguajes de programación modernos que permiten este diseño interno, por ejemplo, usando clases. Usando estas características del lenguaje de programación cliente es beneficioso para el programador incluso fuera del entorno de la base de datos.

Los procedimientos almacenados requieren la tabla proc en la base de datos MySQL. Esta tabla se crea durante la instalación de MySQL 5.0. Si está actualizando a MySQL 5.0 desde una versión anterior, asegúrese de actualizar sus tablas de permisos para asegurar que la tabla proc existe.

Definición general de procedimientos y funciones en MySQL

```
CREATE PROCEDURE sp_name ([parameter[,...]])
    [characteristic ...] routine_body

CREATE FUNCTION sp_name ([parameter[,...]])
    RETURNS type
    [characteristic ...] routine_body

parameter:
    [ IN | OUT | INOUT ] param_name type

type:
    Any valid MySQL data type

characteristic:
    LANGUAGE SQL
  | [NOT] DETERMINISTIC
  | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
  | SQL SECURITY {DEFINER | INVOKER}
  | COMMENT 'string'

routine_body:
    procedimientos almacenados o comandos SQL válidos
```

La lista de parámetros entre paréntesis debe estar siempre presente. Si no hay parámetros, se debe usar una lista de parámetros vacía (). Cada parámetro es un parámetro IN por defecto. Para especificar otro tipo de parámetro, use la palabra clave OUT o INOUT antes del nombre del parámetro. Especificando IN, OUT, o INOUT sólo es válido para una PROCEDURE.

Ejemplo de definición (creación, eliminación y uso) de un procedimiento con parámetros de entrada

Creación de un procedimiento en general

Query | Query Builder | Schema Designer

Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List Matching Tags. [Ctrl+Enter]->List All Tags.

```
1  CREATE PROCEDURE procedure1 /* nombre */ (IN parameter1 INTEGER) /* parametros */
2  BEGIN /*inicio del la programación*/
3   DECLARE variable1 CHAR(10);  /* Declarar variables */
4   IF parameter1 = 17 THEN      /* uso del IF */
5    SET variable1 = 'grande';   /* asignacion  */
6   ELSE
7    SET variable1 = 'pequeño';  /* asignacion */
8   END IF;                      /* fin del IF */
9   INSERT INTO table1 VALUES (variable1);  /* sentencia SQL */
10 END
```

El anterior procedimiento consiste en guardar en una tabla información dependiendo del valor del parámetro. Adaptaremos el ejemplo: supongamos que en una tabla llamada dimensión que tiene dos atributos, se pretende que, dado un atributo, se inserte información en la tabla generando el segundo atributo. Para probar el ejemplo en MySQL, debemos crear una base de datos y la tabla de ejemplo dimensión con dos atributos así:

```
-- creación de la base de datos
CREATE DATABASE prueba;
USE prueba;
-- Creación de la tabla
CREATE TABLE DIMENSION (
 Altura INT NOT NULL,
 tamano VARCHAR(15) NOT NULL
);
-- Creacion del procedimiento almacenado
DELIMITER $$
 CREATE PROCEDURE Decidirdimension(IN altura INTEGER)/*Este parámetro es de entrada*/
 BEGIN
 DECLARE variable1 CHAR(15);
 IF altura >10 THEN
  SET variable1 = 'grande';
 ELSE IF  altura >5 THEN
  SET variable1 = 'mediano';
 ELSE SET variable1 = 'pequeno';
 END IF;
END IF;
INSERT INTO DIMENSION VALUES (altura, variable1);
END $$
DELIMITER ;
```

Porque es importante el uso del DELIMITER: por defecto MySQL usa como DELIMITER el punto y coma (;) , es decir, cada vez que encuentre punto y como(;) ejecuta hasta ahí, debido a que los procedimientos y funciones son varias líneas de códigos y algunas de ellas terminan con este delimiter se ejecutaría solo hasta ahí, lo que ocasionaría un error, es por esto que se hace necesario indicarle a MySQL que utilice otro DELIMITER que puede ser cualquiera para el ejemplo usamos $$ y al finalizar la creación del procedimiento o función volvemos a cambiarlo por ;

Llamar a un procedimiento almacenado

Para llamar un procedimiento se utiliza el comando CALL así:

CALL nombre_de_procedimiento(parametros);

Para el ejemplo seria así:

CALL Decidirdimension(5);

CALL Decidirdimension(8);

CALL Decidirdimension(25);

Eliminar procedimiento

Drop procedure nombre_de_procedimiento

Descargo de responsabilidad

La información contenida en este documento descargable en formato PDF o PPT es un reflejo del material virtual presentado en la versión online del curso. Por lo tanto, su contenido, gráficos, links de consulta, acotaciones y comentarios son responsabilidad exclusiva de su(s) respectivo(s) autor(es) por lo que su contenido no compromete al área de e-Learning del Departamento GES o al programa académico al que pertenece.

El área de e-Learning no asume ninguna responsabilidad por la actualidad, exactitud, obligaciones de derechos de autor, integridad o calidad de los contenidos proporcionados y se aclara que la utilización de este descargable se encuentra limitada de manera expresa para los propósitos educacionales del curso.