Bases de Datos

Unidad 8:

Programación de bases de datos

Sesión 1

Introducción

Todo SGBD permite que los usuarios puedan desarrollar:

RUTINAS
formadas por una serie de instrucciones que permiten realizar una tarea.

Almacenada la rutina, ésta podrá ser invocada o llamada a ejecución en cualquier momento.

Las rutinas que se pueden desarrollar en MySQL y, en general, en cualquier SGBD relacional son:

- ❑ Funciones.
- ❑ Procedimientos.
- ❑ Disparadores o Triggers.

Ventajas de usar rutinas almacenadas:

- Se automatizan procesos que constan de varias instrucciones. No hay que reescribir esas instrucciones.
- Desde los clientes se tiene que enviar muchísima menos información al servidor. El servidor ya tiene las rutinas almacenadas.
- Si están perfectamente comprobadas las rutinas, hay mayor seguridad de que los procesos se realicen correctamente.

Desventajas de usar rutinas almacenadas

- Portabilidad. Hay bastantes diferencias en el lenguaje SQL para crear rutinas en los diferentes SGBD por lo que una base de datos con rutinas creadas en un SGBD puede no ser portable a otro SGBD por esas rutinas.
- Pueden producirse errores de ejecución de una rutina que sean difícilmente detectables.

Introducción

Para el desarrollo de rutinas se usa un lenguaje de programación. En MySQL, el lenguaje de programación incluye una serie de instrucciones para:

- Crear el tipo de rutina.
- Declarar variables.
- Manejar variables.
- Establecer el comportamiento de los parámetros.
- Realizar control de flujo.
- Manejar cursores.
- Controlar eventos.
- Devolver valores.

2.- Variables de usuario y de sistema

En MySQL podemos usar dos tipos de variables:

- **Variables de sistema:**
  - Las crea el servidor cuando se inicia y/o cuando se inicia una sesión.
  - El valor que tengan estas variables configuran el comportamiento del servidor y de las sesiones.
  - Por ejemplo, la variable **autocommit**, que hemos visto al estudiar las transacciones, es una variable de sistema y de sesión.
  - Sólo los usuarios con los privilegios adecuados podrán modificar los valores de estas variables.

- **Variables de usuario:**
  - Las declara o crea un usuario para usarlas y modificarlas dentro de la sesión.
  - Cuando se cierra una sesión, todas las variables de usuario que se hubieran creado en la sesión desaparecen.

2.- Variables de usuario y de sistema

• El servidor crea y mantiene varias **variables de sistema** que indican cómo está configurado. Todas ellas tienen valores por defecto. Puede cambiarse el valor al arrancar el servidor usando opciones en la línea de comandos o en ficheros de configuración. En la mayoría de ellas (**las dinámicas**) podemos modificar su valor en tiempo de sesión usando el comando **SET**.

• Las variables de sistema pueden ser globales o de sesión. Las **variables globales** establecen configuraciones globales del servidor. Las **variables de sesión** configuran las sesiones o para conexiones individuales de clientes. Muchas de ellas son tanto globales como de sesión (realmente tienen un valor global y tienen un valor para cada sesión).

• Iniciado el servidor, se puede modificar el valor de las **variables globales** que sean **dinámicas** ejecutando el comando **SET GLOBAL variable=valor**

• El valor de las **variables de sesión** que son **dinámicas** se puede cambiar mediante un comando **SET SESSION variable=valor**

2.- Variables de usuario y de sistema

- Se puede consultar las variables de sistema y sus valores usando el comando **SHOW VARIABLES**.

Variable_name	Value
character_set_connection	utf8
character_set_database	latin1
character_set_filesystem	binary
character_set_results	utf8
character_set_server	latin1
character_set_system	utf8
character_sets_dir	c:\wamp\bin\mysql\mysql5.0.45\share\charsets\
collation_connection	utf8_general_ci
collation_database	latin1_swedish_ci
collation_server	latin1_swedish_ci
completion_type	0
concurrent_insert	1
connect_timeout	5
datadir	c:\wamp\bin\mysql\mysql5.0.45\data\
date_format	%Y-%m-%d

2.- Variables de usuario y de sistema

• Ejemplos de modificación de variables mediante **SET GLOBAL** o **SET SESSION**. Si se escribe solo SET, es equivalente a SET SESSION. LOCAL, @@SESSION. y @@LOCAL. son equivalentes a SESSION.

EJEMPLOS:

```
SET SESSION autocommit=0;
SET autocommit=0;
SET GLOBAL max_connections=5;
SET GLOBAL character_set_results=utf8;
SET SESSION character_set_results=utf8;
SET @@SESSION.character_set_results=utf8;
SET  character_set_results=utf8;
SET LOCAL character_set_results=utf8;
SET @@local.character_set_results=utf8;
```

Por lo tanto, cuando ejecutábamos SET autocommit = 0,

¿Estábamos modificando una variable GLOBAL o de SESSION?

¿Cómo podíamos estar seguros?

2.- Variables de usuario y de sistema

• Se pueden consultar las variables de sistema cuyo nombre coincide con un patrón. Por ejemplo, si queremos ver los valores de las variables de sesión cuyo nombre comienza por **auto_**, ejecutaríamos:

SHOW SESSION VARIABLES LIKE 'auto_%';

¿Es correcta la expresión anterior?

Variable_name	Value
auto_increment_increment	1
auto_increment_offset	1

• El resultado nos da el valor de dos variables que indican cual es el valor inicial o de arranque para una columna **autoincrement** y cuanto se incrementa el valor de esa columna cada vez que se inserta un nuevo registro.

2.- Variables de usuario y de sistema

- También se pueden obtener los valores de las variables de sesión usando SELECT.
- Para consultar su valor se debe escribir el nombre de la variable precedido de @@

EJEMPLO:

```
SELECT @@autocommit, @@max_connections, @@character_set_results;
```

@@autocommit	@@max_conn...	@@character_set...
1	255	utf8

2.- Variables de usuario y de sistema

ALGUNAS VARIABLES DE SISTEMA IMPORTANTES:

- **autocommit:** Vale 0 si está activado el estado transaccional.
- **basedir:** Ruta del directorio raíz de MySQL. No es dinámica.
- **character_set_server:** Conjunto de caracteres utilizado en el servidor.
- **collation_server:** Colación por defecto del servidor.
- **datadir:** Directorio donde se guardan las bases de datos. No es dinámica.
- **init_file:** Nombre del archivo de configuración del servidor, por defecto MY.INI. No dinámica.
- **log:** Vale true si se activa el registro de consultas. No dinámica
- **log_updates:** Vale true si se ha activado el registro de actualizaciones. No dinámica.

¿Qué característica define a una variable como dinámica?

2.- Variables de usuario y de sistema

ALGUNAS VARIABLES IMPORTANTES:

- **max_connections:** Máximo número de conexiones permitidas de forma simultánea.
- **max_user_connections:** Máximo número de sesiones que puede tener iniciadas un usuario.
- **port:** Número de puerto que usa MySQL para escuchar conexiones TCP/IP. No dinámica.
- **skip_networking:** Vale true si el servidor sólo admite conexiones locales. No dinámica.
- **table_type:** Tipo de tabla predeterminado al crear tablas sin especificar su tipo. No dinámica.

2.- Variables de usuario y de sistema

VARIABLES DE ESTADO

- El servidor mantiene muchas variables de estado que proveen de información sobre sus operaciones. Puede ver estas variables y sus valores utilizando la sentencia **SHOW STATUS.**

- El valor de estas variables lo modifica el servidor en función del estado en que se encuentra o de las acciones realizadas por los usuarios.

- Nunca puede un usuario modificar directamente el valor de estas variables.

2.- Variables de usuario y de sistema

ALGUNAS VARIABLES DE ESTADO

- **questions:** El número de consultas que han sido enviadas al servidor.
- **uptime:** El número de segundos que el servidor ha estado funcionando.
- **slow_queries:** El número de consultas que han tardado más de long_query_time segundos
- **max_used_connections:** El número máximo de conexiones que han sido utilizadas simultáneamente desde que el servidor ha sido iniciado.
- **innodb_rows_inserted:** El número de registros insertados en tablas InnoDB.
- **connections:** El número de intentos de conexión (con éxito o no) al servidor MySQL.
- **aborted_connects:** El número de intentos de conexión al servidor MySQL que han fallado.
- **com_select:** Número de instrucciones select ejecutadas en la sesión.
- **com_insert:** Número de instrucciones insert ejecutadas en la sesión.

2.- Variables de usuario y de sistema

VARIABLES DE USUARIO

Como ya vimos en el tema 5 (sesión 7) Un usuario puede crear variables propias o de usuario.

Las variables de usuario se declaran con **@nombre_var**, donde el nombre de variable **nombre_var** puede consistir de caracteres alfanuméricos y los caracteres '**.**', '**_**', y '**$**'.

Una forma de establecer una variable de usuario es empleando una instrucción **SET**:

SET @*nombre_var* = *expresion*;

Otra forma de crear variables de usuario y/o asignarles valores es hacerlo asignándoles un valor devuelto por una consulta. Por ejemplo;

```
SELECT @maximo:=MAX(numcontrato) FROM contratos;
SELECT * FROM contratos WHERE numcontrato=@maximo;
--
SELECT max(numcontrato) INTO @nummayor FROM contratos;
SELECT fini INTO @fecha FROM contratos WHERE numcontrato=@nummayor;
```