TEMA 09 – GESTIÓN DE USUARIOS Y PERMISOS

Creación de usuarios en MySQL

* CREATE USER user IDENTIFIED BY ‘password’;



* Opciones para gestionar la caducidad de la contraseña:
  + ‘PASSWORD EXPIRE’: obliga a cambiar la contraseña en el próximo inicio de sesión
  + ‘PASSWORD EXPIRE INTERVAL n DAY’: obliga a cambiar la contraseña cada n días
  + ’PASSWORD EXPIRE NEVER’: la contraseña nunca caduca
  + ‘PASSWORD EXPIRE DEFAULT’: la contraseña caduca en el nº de días que indica la variable del sistema default\_password\_lifetime
    - 360 días en las versiones inferiores a 5.7.11 y no caduca a partir de esa
  + ‘ACCOUNT LOCK’: podemos crear un usuario con la cuenta bloqueada
* Podemos limitar los recursos con las siguientes opciones:
  + ‘MAX\_QUERIES\_PER\_HOUR’
  + ‘MAX\_UPDATES\_PER\_HOUR’
  + ‘MAX\_CONNECTIONS\_PER\_HOUR’
  + ‘MAX\_USER\_CONNECTIONS’: máximo de conexiones simultáneas que se permiten a cada usuario, por defecto ilimitadas



Modificación de usuarios

* ‘ALTER USER [IF EXISTS] user;’
* Para bloquear usuario: ‘ALTER USER [IF EXISTS] user ACCOUNT LOCK’
* Para modificar la contraseña: ‘ALTER USER [IF EXISTS] user IDENTIFIED BY ‘NuevaPass’’
* Para modificar el nombre: ‘RENAME USER user@localhost YO usuario@localhost;’
* Para borrar usuario: ‘DROP USER usuario@localhost;’

Gestión de permisos

* Permisos más habituales que se pueden asignar a los usuarios:
  + ALL PRIVILEGES: acceder a todas las bases de datos asignadas en el sistema
  + CREATE: crear nuevas tablas o bases de datos
  + DROP: borrar tablas o bases de datos
  + DELETE: eliminar registros de las tablas
  + INSERT: insertar registros en las tablas
  + SELECT: leer registros en las tablas
  + UPDATE: actualizar registros en las tablas
  + GRANT OPTION: permite remover privilegios de usuarios
  + CREATE USER: permite gestionar los usuarios
  + EXECUTE: permite ejecutar procedimientos y funciones
* Orden GRANT. Asignar permisos
  + ‘GRANT permiso ON [nombreBaseDatos].[nombreTabla] TO usuario’
  + \*.\* hace referencia a todas las bases de datos y tablas del sistema
  + Basedatos.\* hace referencia a totas las tablas de la base de datos indicada
  + Basedatos.tabla solo hacer referencia a la tabla de la base de datos
  + Una vez asignados los privilegios no es necesario ejecutar la orden FLUSH PRIVILEGES
  + Solo es necesaria esta instrucción si modificamos los valores utilizando un update, delete o insert
* Mostrar privilegios
  + ‘SHOW GRANTS’
  + Los cambios a nivel global tienen efecto cuando el usuario vuelve a conectarse
  + Los cambios a nivel de base de datos cuando el usuario vuelve a seleccionarla
  + Los cambios a nivel de tabla y columna tienen un efecto inmediato
* Orden REVOKE. Quitar privilegios
  + ‘REVOKE permiso ON [nombreBaseDatos].[nombredetabla] FROM usuario’
* Opción WITH GRAN OPTION
  + Habilitamos a ese usuario para que pueda otorgar ese mismo permiso a otros usuarios

A close-up of a sign

Description automatically generated

Gestión de roles

* Conjunto de privilegios que se pueden otorgar a un usuario o a otro Rol
* Simplifica el trabajo de DBA
* Pueden ser de nivel global, de base de datos o de tabla y columnas
* ‘CREATE ROLE [IF NOT EXISTS] nombrerol;’
* Para asignar y retirar privilegios a los roles utilizamos la misma orden GRANT
* REVOKE para eliminarlos
* Un usuario puede tener diferentes roles asignados
* En el último paso seleccionamos cual es el rol por defecto que va a utilizar con la instrucción SET DEFAULT ROLE



TEMA 10 – GESTIÓN DE TRANSACCIONES

Definición

* Conjunto de sentencias SQL que se ejecutan formando una unidad lógica de trabajo
  + Forma indivisible o atómica (LUW)
  + Finaliza con un ‘commit’
  + MySQL permite realizar transacciones si usamos el motor InnoDB
  + Permite realizar operaciones de forma segura y recuperar datos si se produce algún fallo en el servidor durante la transacción
  + Pueden aumentar el T de ejecución de las transacciones
  + Deben cumplir las 4 propiedades ACID

Propiedades ACID

* Garantizan que las transacciones se puedan realizar en una BBDD de forma segura
* Atomicidad: una transacción es indivisible (o todas o ninguna sentencia)
* Consistencia: después de una transacción la BBDD estará en un formato válido y consistente
* Aislamiento: cada transacción está aislada del resto
* Durabilidad: los cambios que realiza una transacción son permanentes

Autocommit

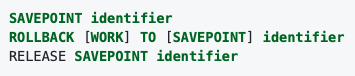
* Activado por defecto en InnoDB
* Automática- se aceptan todos los cambios realizados después de una sentencia SQL y no es posible deshacerlos
* Consultar el valor: ‘SELECT @@AUTOCOMMIT;’
* Desactivar: ‘SET AUTOCOMMIT = 0;’
* Activar: ‘SET AUTOCOMMIT = 1;’

Start transaction, commit y rollback

* Pasos
  + Indicar que vamos a realizar una transacción
    - ‘START TRANSACTION’, ‘BEGIN’ o ‘BEGIN WORK’
  + Realizar las operaciones de manipulación de datos (insertar, actualizar, borrar)
  + ‘COMMIT’: si se han realizado correcta- y queremos que los cambios se apliquen permanente-
    - ‘ROLLBACK’: si ocurre algún error y no queremos aplicarlos

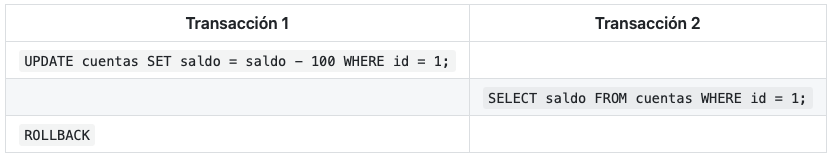
Savepoint, rollback to savepoint y release savepoint

* Con InnoDB
* ‘SAVEPOINT’: permite establecer un punto de recuperación dentro de la transacción
  + Usando un identificador
  + Si existen 2 savepoints con el mismo nombre solo se tendrá en cuenta el último
* ‘ROLLBACK TO SAVEPOINT’: hace rollback hasta el savepoint indicado
* ‘RELEASE SAVEPOINT’: elimina un savepoint

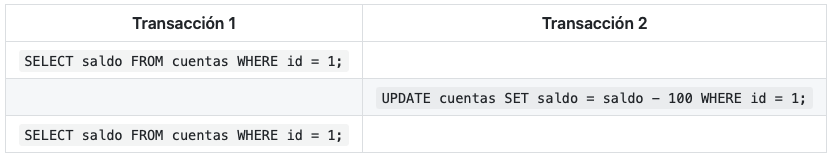


Acceso concurrente a los datos

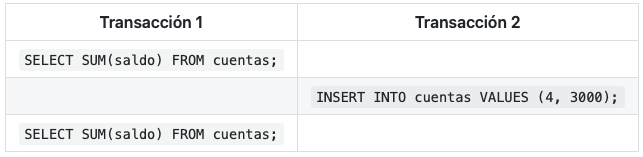
* Dirty read (lectura sucia):cuando una segunda transacción lee datos que están siendo modificados por una transacción antes de que haga un commit



* Non-Repeatable Read (lectura no repetible): cuando una transacción consulta el mismo dato dos veces durante la ejecución de la transacción y la segunda vez el valor del dato ha sido modificado por otra transacción

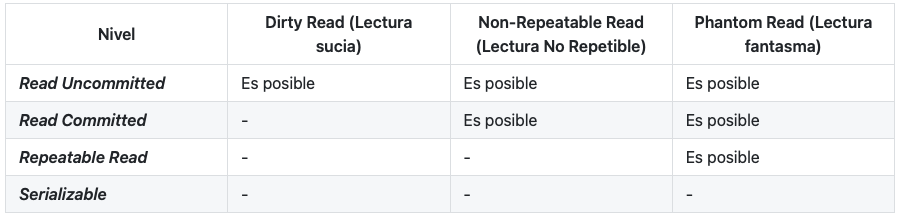


* Phantom Read (lectura fantasma): cuando una transacción ejecuta dos veces una consulta que devuelve un conjunto de filas y en la segunda aparecen nuevas filas en el conjunto que no existían



Niveles de aislamiento

* Red Uncommitted: no se realiza ningún bloqueo
* Read Committed: los datos leídos por una transacción pueden ser modificados por otras transacciones
* Repeatable Read: ningún registro leído con un SELECT puede ser modificado por otra transacción
* Serializable: las transacciones se ejecutan unas detrás de otras, sin posibilidad de concurrencia



* Consultarlo: @@transaction\_isolation
* Variable global: “**SELECT** @@GLOBAL.transaction\_isolation;”
* Variable de sesión: “**SELECT** @@SESSION.transaction\_isolation;” o “**SELECT** @@transac

Cómo realizar transacciones con procedimientos almacenados