VII. TRANSACCIONES EN SQL 2008

Las transacciones proporcionan un mecanismo para agrupar una serie de cambios de base de datos en una operación lógica. Una vez realizados, los cambios en la base de datos se pueden confirmar o cancelar como una única unidad. Microsoft SQL Server es compatible con las transacciones. Las transacciones se pueden utilizar de varias maneras:

- Mediante programación utilizando ADO .NET o las funciones de API de OLE DB
- Mediante el uso del Analizador de consultas de SQL Server Compact Edition en un dispositivo
- Mediante el uso de SQL Server Management Studio en un escritorio

Las transacciones garantizan que se siguen las propiedades ACID (atomicidad, coherencia, aislamiento y duración) de forma que los datos se confirmen correctamente en la base de datos. Para obtener más información acerca de las propiedades ACID de las transacciones, vea "Transacciones" en los Libros en pantalla de SQL Server.

CONTROLAR TRANSACCIONES

Las aplicaciones controlan las transacciones especificando el momento en que una transacción se inicia y termina. Los usuarios controlan las transacciones mediante:

- Instrucciones SQL de SQL Server Management Studio.
- Funciones API de base de datos ADO .NET y OLE DB.

Importante Se puede administrar cada transacción utilizando únicamente una de las API. El uso de ambas API en la misma transacción puede producir resultados indefinidos.

La aplicación debe poder gestionar correctamente los errores que se producen cuando una transacción termina antes de haberse completado.

Utilizar instrucciones SQL

Puede iniciar y finalizar las transacciones mediante instrucciones SQL.

Fuente: Libros en pantalla de SQL Server 2008 de Microsoft Corporation (Edición de Enero de 2009)

Iniciar transacciones

Puede iniciar las transacciones del Database Engine (Motor de base de datos) de SQL Server como explícitas o de confirmación automática.

- Transacciones explícitas
  Inicie explícitamente una transacción ejecutando una instrucción **BEGIN TRANSACTION**.
- Transacciones autoconfirmadas
  Se trata del modo predeterminado en SQL Server Compact Edition. Una transacción auto confirmada se inicia cuando se inicia la instrucción de la operación y se confirma cuando finaliza la instrucción.

Finalizar transacciones

Puede finalizar las transacciones con una instrucción **COMMIT** o **ROLLBACK**.

- **COMMIT**
  Una instrucción **COMMIT** garantiza que todas las modificaciones de la transacción se convierten en una parte permanente de la base de datos. Una instrucción **COMMIT** también libera recursos utilizados por la transacción, por ejemplo bloqueos.
- **ROLLBACK**
  Si se produce un error en una transacción, o bien si el usuario decide cancelar la transacción, una instrucción **ROLLBACK** deshace la transacción. Una instrucción **ROLLBACK** deshace todas las modificaciones realizadas en la transacción devolviendo los datos al estado en el que se encontraban al iniciar la transacción. Una instrucción **ROLLBACK** también libera algunos recursos retenidos por la transacción.

Fuente: Libros en pantalla de SQL Server 2008 de Microsoft Corporation (Edición de Enero de 2009)

Secuencia de comandos Transact-SQL

Una secuencia de comandos es una serie de instrucciones Transact-SQL almacenadas en un archivo. El archivo se puede utilizar como entrada para el editor de código de SQL Server Management Studio o las utilidades **sqlcmd** y **osql**. Las utilidades ejecutarán, entonces, las instrucciones SQL almacenadas en el archivo.

Las secuencias de comandos Transact-SQL constan de uno o varios lotes. El comando GO señala el final de un lote. Si una secuencia de comandos Transact-SQL no contiene ningún comando GO, se ejecutará como un único lote.

Se pueden utilizar secuencias de comandos Transact-SQL para hacer lo siguiente:

- Mantener una copia permanente de los pasos usados para crear y rellenar las bases de datos en el servidor, como mecanismo de copia de seguridad.
- Transferir las instrucciones de un equipo a otro, cuando sea necesario.
- Formar rápidamente a los nuevos empleados al permitirles encontrar problemas en el código, entenderlo o cambiarlo.

SQLCMD

La utilidad **sqlcmd** permite escribir instrucciones Transact-SQL, procedimientos del sistema y archivos de secuencia de comandos en el símbolo del sistema, en el **Editor de consultas** en modo SQLCMD, en un archivo de secuencia de comandos de Windows o en un paso de trabajo del sistema operativo (Cmd.exe) de un trabajo del Agente SQL Server. Esta utilidad utiliza OLE DB para ejecutar lotes de Transact-SQL.

Opciones importantes del comando SQLCMD

-U *login_id*

Es el identificador de inicio de sesión del usuario.

Si no se especifica la opción **-U** ni la opción **-P**, **sqlcmd** intenta conectarse mediante el modo de autenticación de Microsoft Windows. La autenticación se basa en la cuenta de Windows del usuario que está ejecutando **sqlcmd**.

Fuente: Libros en pantalla de SQL Server 2008 de Microsoft Corporation (Edición de Enero de 2009)

-P *password*

Es una contraseña especificada por el usuario. En las contraseñas se distingue entre mayúsculas y minúsculas. Si se utiliza la opción -U y no la opción **-P** y además no se ha establecido la variable de entorno SQLCMDPASSWORD, **sqlcmd** solicita al usuario una contraseña. Si se utiliza la opción **-P** al final del símbolo del sistema sin una contraseña, **sqlcmd** utiliza la contraseña predeterminada (NULL).

-S *server_name* [*****instance_name*]

Especifica la instancia de SQL Server a la que hay que conectarse. Establece la variable de secuencia de comandos de **sqlcmd** SQLCMDSERVER.

Especifique *server_name* para conectarse a la instancia predeterminada de SQL Server en ese equipo servidor. Especifique *server_name* [*****instance_name*] para conectarse a una instancia con nombre de SQL Server en ese equipo servidor. Si no se especifica ningún equipo servidor, **sqlcmd** se conecta a la instancia predeterminada de SQL Server en el equipo local. Esta opción es necesaria si se ejecuta **sqlcmd** desde un equipo remoto conectado a la red.

Opciones de entrada o salida

-i *input_file*[**,***input_file2*...]

Identifica el archivo que contiene un lote de instrucciones SQL o procedimientos almacenados. Se pueden especificar varios archivos que se leerán y se procesarán en orden. No utilice espacios en blanco entre los nombres de archivo. **sqlcmd** comprobará primero si todos los archivos especificados existen. Si uno o más archivos no existen, **sqlcmd** se cerrará. Las opciones -i y -Q/-q se excluyen mutuamente.

Ejemplos de rutas de acceso:

-i C:\<nombreDeArchivo>

-i \\<Servidor>\<recursoCompartido$>\<nombreDeArchivo>

-i "C:\Carpeta\<nombreDeArchivo>"

Las rutas de acceso que contienen espacios deben incluirse entre comillas.

Fuente: Libros en pantalla de SQL Server 2008 de Microsoft Corporation (Edición de Enero de 2009)

Esta opción se puede utilizar más de una vez: **-i** *input_file* **-i** *I input_file.*

-o *output_file*

Identifica el archivo que recibe la salida de **sqlcmd**.

Si se especifica **-u**, el archivo *output_file* se almacena en formato Unicode. Si el nombre de archivo no es válido, se genera un mensaje de error y **sqlcmd** se cierra. **sqlcmd** no admite la escritura simultánea de varios procesos de **sqlcmd** en el mismo archivo. El archivo de salida estará dañado o será incorrecto. Vea el modificador **-f** para obtener más información acerca de formatos de archivo. Este archivo se creará si no existe. Se sobrescribirá cualquier archivo con el mismo nombre que pertenezca a una sesión de **sqlcmd** anterior. El archivo que se especifica aquí no es el archivo **stdout**. Si se especifica un archivo **stdout**, este archivo no se utilizará.

Ejemplos de rutas de acceso:

-o C:\< nombreDeArchivo>

-o \\<Servidor>\<recursoCompatido$>\<nombreDeArchivo>

-o "C:\Carpeta\<nombreDeArchivo>"

Las rutas de acceso que contienen espacios deben incluirse entre comillas.