Transacciones en SQL con Ejemplos para la Base de Datos de Gestión Médica

Una **transacción en SQL** es un conjunto de operaciones que se ejecutan como una unidad, asegurando la consistencia y confiabilidad de los datos según las propiedades ACID.

* Atomicidad: La transacción se ejecuta completamente o no se ejecuta en absoluto. Si una parte falla, toda la transacción se revierte.
* Consistencia: Las transacciones llevan la base de datos de un estado válido a otro.
* Aislamiento: Los cambios que pueden generar una transacción no se pueden ver hasta que esta finalice.
* Durabilidad: Una vez que se ha confirmado (commit) una transacción, los cambios persisten, incluso si el sistema falla.

Operaciones básicas en una Transacción

Las transacciones se gestionan con operaciones fundamentales:

* **BEGIN TRANSACTION**: Inicia la transacción.
* **COMMIT**: Finaliza la transacción y confirma todos los cambios realizados en la base de datos.
* **ROLLBACK**: Revierte todos los cambios realizados desde el inicio de la transacción o desde un punto de guardado (savepoint).
* **SAVEPOINT**: Crea un punto dentro de una transacción al cual puedes volver en caso de fallo sin deshacer toda la transacción.

**Ejemplo**:

Supongamos que insertamos un nuevo paciente, una cita asociada y luego actualizamos su ficha médica. Si cualquiera de estos pasos falla, todos los cambios se revierten:

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Funcionamiento de Transacciones Anidadas.

Las transacciones anidadas permiten iniciar una nueva transacción dentro de una transacción existente.

Esto es muy eficiente y útil cuando se tratan de procesos que involucran operaciones complejas, debido a esto podemos controlar y revertir algunas de estas operaciones de manera independiente. Sin perder el control de la transacción completa.

Hablando del motor que nosotros utilizamos “SQL SERVER” las transacciones anidadas se manejan a través de contadores. Cada “BEGIN TRANSACTION” incrementa el contador, mientras que “COMMIT” y “ROLLBACK” lo decrementan

En estas situaciones es donde mas se ven reflejados los Puntos de Guardado (Savepoints), ya que nos sirve para marcar puntos intermedios dentro de una transacción. Gracias a estos puntos un “ROLLBACK” nos permite volver a un estado intermedio sin deshacer toda la transacción.

Como SQL Server no permite revertir a un SAVEPOINT después de un error en el mismo bloque transaccional; cualquier excepción marca la transacción como "abortada" y requiere una reversión completa. Sin embargo, una forma alternativa es hacer la inserción del paciente en una transacción separada o antes del bloque transaccional de la inserción de la cita.

Ejemplo:

Para ello podemos simular un error después del primer INSERT en Paciente. Por ejemplo, provocaremos un error en INSERT de Cita.

Texto

Descripción generada automáticamente

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Lo que ocurrió en este ejemplo es que la separación de transacciones nos permitieron que se ejecute la inserción del paciente y que permanezca en la base, mientras que la segunda transacción fallo y ni un dato perteneciente a esta segunda haya provocado cambios en la base.

Esto deshará solo los cambios realizados después de ese SAVEPOINT.

También se puede optar para usar el ROLLBACK total ya que esto nos permite verificar que la transacción se revierta completamente, evitando inserciones parciales.

Conclusiones basadas en pruebas.

1. Atomicidad e Integridad: La transacción garantiza que todas las operaciones se ejecuten como una unidad, si algún paso falla revierte todos los cambios. Pudiendo así preservar la base de datos sin registros incompletos.
2. Efectividad de la Revisión ante Fallos: Al provocar un error, la transacción se revierte y ni un dato es insertado o actualizado, manteniendo la consistencia de datos
3. Uso de SAVEPOINT: Aun que fueron savepoints como tal, lo tratamos como transacciones anidadas, en un ejemplo sencillo para que se vea su funcionamiento, esto nos permiten tener un control de las transacciones, pudiendo así revertir partes específicas.