**LICENCIATURA EN SISTEMAS DE INFORMACIÓN**

**Bases de datos I**

PROYECTO DE ESTUDIO E INVESTIGACIÓN

**Tema: Manejo de transacciones y transacciones anidadas**

**Profesores:**

* Cuzziol, Juan José
* Sambrana, Ivan

**Grupo n° 4:**

* Duete, Juan Pablo
* Bartolucci, Gastón

Contenido

[CAPÍTULO I: INTRODUCCIÓN 3](#_Toc149126957)

[CAPITULO II: MARCO CONCEPTUAL O REFERENCIAL 3](#_Toc149126958)

[Transacciones 3](#_Toc149126959)

[Transacciones anidadas 4](#_Toc149126960)

# CAPÍTULO I: INTRODUCCIÓN

El presente trabajo de investigación tiene como objetivo entender el funcionamiento de las transacciones en SQL Server, así como los distintos tipos de opciones que tenemos en su aplicación. La elección del tipo de transacción que escogemos para realizar depende concretamente del dominio del problema a resolver.

Para empezar a trabajar con transacciones debemos entender que una transacción es una unidad única de trabajo. Si una transacción tiene éxito, todas las modificaciones de los datos realizadas durante la transacción se confirman y se convierten en una parte permanente de la base de datos. Si una transacción encuentra errores y debe cancelarse o revertirse, se borran todas las modificaciones de los datos.

Cuando enviamos una sentencia al SQL Server se escribe en el fichero de transacciones lo que va a ocurrir y a continuación realiza los cambios necesarios en la base de datos. Si hay algún tipo de problema al hacer esta operación el SQL Server puede leer en el fichero de transacciones lo que se estaba haciendo y si es necesario puede devolver la base de datos al estado en el que se encontraba antes de recibir la sentencia.

Por supuesto este tipo de transacciones no requieren de nuestra intervención puesto que el sistema se encarga de todo. Sin embargo, si hay que realizar varias operaciones y queremos que sean tratadas como una unidad tenemos que crear esas transacciones de manera explícita.

En un conjunto de operaciones marcadas como transacción, todas las operaciones que la conforman tienen éxito o todas fracasan.

En ciertas ocasiones también necesitamos llamar a un procedimiento dentro de una transacción, y que éste a su vez tenga otra transacción. Con lo cual tendríamos una transacción dentro de otra. Esto es lo que se conoce como transacciones anidadas.

# CAPITULO II: MARCO CONCEPTUAL O REFERENCIAL

## 

## Transacciones

Una transacción es un conjunto de operaciones que van a ser tratadas como una única unidad.

Estas transacciones deben cumplir 4 propiedades fundamentales comúnmente conocidas como ACID (atomicidad, consistencia, asilamiento y durabilidad):

1. **Atomicidad:** Es la propiedad de las transacciones que permite observarlas como operaciones atómicas: ocurren totalmente o no ocurren.
2. **Consistencia:** La ejecución aislada de la transacción conserva la consistencia de la base de datos.
3. **Aislamiento:** Las transacciones son independientes entre sí.
4. **Durabilidad:** El sistema gestor de bases de datos asegura que perduren los cambios realizados por una transacción que termina con éxito.

Una transacción puede estar en distintos estados:

• Activa: Durante su ejecución.

• Parcialmente comprometida: Después de ejecutar su última instrucción.

• Fallida: Imposible de continuar su ejecución normal.

• Abortada: Transacción retrocedida y base de datos restaurada al estado anterior a su

ejecución. Se puede reiniciar o cancelar.

**La transacción más simple en SQL Server es una única sentencia SQL.**

La sentencia que se utiliza para indicar el comienzo de una transacción es 'BEGIN TRAN'. Si alguna de las operaciones de una transacción falla hay que deshacer la transacción en su totalidad para volver al estado inicial en el que estaba la base de datos antes de empezar. Esto se consigue con la sentencia 'ROLLBACK TRAN'.

Si todas las operaciones de una transacción se completan con éxito hay que marcar el fin de una transacción para que la base de datos vuelva a estar en un estado consistente con la sentencia 'COMMIT TRAN'.

## Transacciones anidadas

Otra de las posibilidades que nos ofrece el SQL Server es utilizar transacciones anidadas. Esto quiere decir que podemos tener transacciones dentro de transacciones, es decir, podemos empezar una nueva transacción sin haber terminado la anterior.

Asociada a esta idea de anidamiento existe una variable global @@TRANCOUNT que tiene valor 0 si no existe ningún nivel de anidamiento, 1 si hay una transacción anidada, 2 si estamos en el segundo nivel de anidamiento. y así sucesivamente.

La dificultad de trabajar con transacciones anidadas está en el comportamiento que tienen ahora las sentencias 'COMMIT TRAN' y 'ROLLBACK TRAN'

* ROLLBACK TRAN: Dentro de una transacción anidada esta sentencia deshace todas las transacciones internas hasta la instrucción BEGIN TRANSACTION más externa.
* COMMIT TRAN: Dentro de una transacción anidada esta sentencia únicamente reduce en 1 el valor de @@TRANCOUNT, pero no "finaliza" ninguna transacción ni "guarda" los cambios. En el caso en el que @@TRANCOUNT=1 (cuando estamos en la última transacción) COMMIT TRAN hace que todas las modificaciones efectuadas sobre los datos desde el inicio de la transacción sean parte permanente de la base de datos, libera los recursos mantenidos por la conexión y reduce @@TRANCOUNT a 0.