**TP1-T3 (Teoría Gestión Transacciones)**

Jorge Arturo Aliaga Valencia 218166141

1. **¿Cuál es el concepto de Base de Datos Consistente?**

﻿Se refiere a la exigencia que cualquier ﻿transacción de bases de datos sólo debe cambiar datos afectados de formas permitidas.

Los datos escritos en la base de datos deben ser válidos según todas las reglas definidas, incluyendo ﻿limitaciones, ﻿Cascadas, ﻿factores desencadenadores y cualquier combinación de éstos. Esto no garantiza la exactitud de la operación de todas maneras hubiera querido programador de la aplicación (que es la responsabilidad del código de nivel de aplicación) pero simplemente que los errores de programación no pueden resultar en la violación de las reglas definidas.

1. **Al ejecutar una o más instrucciones SQL ¿Cuáles son las posibles razones para que las instrucciones SQL NO terminen de ejecutarse?**

* Situación excepcional detectada que hace que el programa no pueda continuar
* Falla del programa
* Falla del software de la base de datos
* Falla del sistema operativo
* Falla del hardware
* Falla de energía eléctrica
* Control de concurrencia ha encontrado un conflicto
* Control de concurrencia ha detectado un deadlock

1. **Al ejecutar dos o más instrucciones SQL. Defina el concepto de Ejecución Serializable.**

Idealmente, deberían ser transacciones *serializable*. Las transacciones se dice que son serializable si los resultados de ejecutar al mismo tiempo las transacciones son los mismos que los resultados de su ejecución en serie: es decir, una tras otra. No es importante qué transacción se ejecuta en primer lugar, solo que el resultado no refleja cualquier combinación de las transacciones.

1. **Al ejecutar dos o más instrucciones SQL. Defina el concepto de Atomicidad.**

La atomicidad previene que las actualizaciones a la base ocurren de forma parcial, lo cual podría ocasionar mayores problemas que rechazar la transacción entera. En otras palabras, la atomicidad significa indivisibilidad e irreductibilidad.

La Atomicidad requiere que cada transacción sea "todo o nada": si una parte de la transacción falla, todas las operaciones de la transacción fallan, y por lo tanto la base de datos no sufre cambios. Un sistema atómico tiene que garantizar la atomicidad en cualquier operación y situación, incluyendo fallas de alimentación eléctrica, errores y caídas del sistema.

1. **¿Qué es una Transacción?**

Es un conjunto de instrucciones que se ejecutan formando una unidad de trabajo, es decir, en forma indivisible o atómica.

1. **¿Las Transacciones resuelven el problema de Seriabilidad y Atomicidad?**

Si para ello deben cumplir las 4 propiedades que garantizan que las transacciones en las bases de datos se realicen de forma confiable (atomicidad, coherencia, asilamiento y durabilidad).

1. **¿Cuándo usar Transacciones?**

Cuando queremos que las operaciones Transact SQL que se ejecutan como un único bloque, es decir, si falla una operación Transact SQL fallan todas.

1. **¿Cuáles son los beneficios de usar Transacciones?**

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

1. **¿Cuáles es el Rol de las Transacciones en una Base de Datos?**

* Proteger los datos de las fallas del software, hardware y potencia eléctrica.
* Permitir el aislamiento de datos de tal forma que varios usuarios pueden acceder simultáneamente a los datos sin interferencia.

1. **¿Cómo se inicia una transacción?**

La transacción más simple en SQL Server es una única sentencia SQL. una transacción ‘autocommit’, una transacción autocompletada.

**UPDATE** clientes **SET** sexo='F' **WHERE** sexo ='FEMENINO'

Cuando enviamos esta 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.

1. **¿Defina el concepto de COMMIT en las transacciones?**

Confirma las instrucciones de modificación de la base de datos.

1. **¿Defina el concepto de ROLLBACK en las transacciones?**

Cancela las instrucciones de modificación de la base de datos.

1. **¿Describa las propiedades ACID de las transacciones?**

**Atomicidad**

La Atomicidad requiere que cada transacción sea "todo o nada": si una parte de la transacción falla, todas las operaciones de la transacción fallan, y por lo tanto la base de datos no sufre cambios. Un sistema atómico tiene que garantizar la atomicidad en cualquier operación y situación, incluyendo fallas de alimentación eléctrica, errores y caidas del sistema.

**Consistencia**

La propiedad de Consistencia se asegura que cualquier transacción llevará a la base de datos de un estado válido a otro estado válido. Cualquier dato que se escriba en la base de datos tiene que ser válido de acuerdo a todas las reglas definidas, incluyendo (pero no limitado a) los constraints, los cascades, los triggers, y cualquier combinación de estos.

**AIslamiento**

El aislamiento ("Isolation" en inglés) se asegura que la ejecución concurrente de las transacciones resulte en un estado del sistema que se obtendría si estas transacciones fueran ejecutadas una atrás de otra. Cada transacción debe ejecutarse en aislamiento total; por ejemplo, si T1 y T2 se ejecutan concurrentemente, luego cada una debe mantenerse independiente de la otra.

**Durabilidad**

La durabilidad significa que una vez que se confirmó una transacción (commit), quedará persistida, incluso ante eventos como pérdida de alimentación eléctrica, errores y caídas del sistema. Por ejemplo, en las bases de datos relacionales, una vez que se ejecuta un grupo de sentencias SQL, los resultados tienen que almacenarse inmediatamente (incluso si la base de datos se cae inmediatamente luego)

1. **¿Describa los diferentes estados de las Transacciones?**

**Activa (estado inicial)**: la transacción permanece en este estado durante su ejecución.

**Parcialmente Comprometida**: la transacción pasa a este estado cuando acaba de realizar la última instrucción.

**Fallida**: la transacción pasa a este estado tras descubrir que no puede continuar la ejecución normal.

**Abortada**: la transacción pasa a este estado después de haber restablecido la base de datos a su estado anterior.

**Comprometida**: la transacción pasa a este estado tras completarse con éxito

1. **¿Describa el concepto de BITACORA en una Base de Datos?**

Es utilizada para la recuperación del sistema después de una falla y para garantizar las propiedades ACID de las transacciones.

1. **¿Cuál es el rol de la BITACORA en una Base de Datos?**

Permite registrar, analizar detectar y notificar eventos que suceden en cualquier sistema de información utilizado en las organizaciones. La estructura más ampliamente usada para grabar las modificaciones de la BD.

La importancia de las bitácoras es la de recuperar información ante incidentes de seguridad, detección de comportamiento inusual, información para resolver problemas, evidencia legal, es de gran ayuda en las tareas de cómputo forense.

Es fundamental que siempre se cree un registro en la bitácora cuando se realice una escritura antes de que se modifique la base de datos.

También tenemos la posibilidad de deshacer una modificación que ya se ha escrito en la base de datos, esto se realizará usando el campo del valor antiguo de los registros de la bitácora.

Los registros de la bitácora deben residir en memoria estable como resultado el volumen de datos en la bitácora puede ser exageradamente grande.

Una bitácora puede registrar mucha información acerca de eventos relacionados con el sistema que la genera los cuales pueden ser:

•          Fecha y hora.

•          Host origen.

•          Usuario.

•          Actividad realizada.

1. **Escriba las instrucciones básicas que debe incluir una Transacción.**

BEGIN TRAN[SACTION]

Instrucción 1

...

SAVE TRAN[SACTION] sp

...

ROLLBACK [TRAN[SACTION] sp]

...

Instrucción n COMMIT [TRAN[SACTION]]