TEXT CLASS REVIEW

TEMAS A TRATAR EN EL CUE:

- Transaccionalidad.
- La integridad de datos
- ¿Cuándo usar transacciones?
- Instrucciones Begin, Commit y Rollback.
- La propiedad ACID (Atomicidad, Consistencia, Aislamiento, Permanencia).
- Journaling y bloqueos.

TRANSACCIONALIDAD

¿Qué es una transacción de base de datos?

Una transacción es una colección de acciones que hacen transformaciones consistentes de los estados de un sistema, preservando la consistencia de éste.

La ejecución de un programa que incluye operaciones de acceso a la base de datos se denomina “transacción de base de datos”, o simplemente transacción. Mientras que, si las operaciones de base de datos en una transacción no actualizan datos, sino que sólo los leen, se habla de una transacción sólo de lectura.

Una transacción es una unidad de ejecución de un programa, y ésta puede consistir en varias operaciones de acceso a la base de datos.

¿Por qué se necesita control de transacciones?

Éste impone la integridad de base de datos, garantizando que los lotes de operaciones SQL se ejecuten completamente, o no se ejecuten.

Una transacción es una secuencia de operaciones, que llevan la base de datos desde un estado de consistencia, a otro estado de consistencia; por ello, también suele decirse que es una unidad lógica de integridad. Cuando múltiples transacciones son introducidas en el sistema por varios usuarios, es necesario evitar que interfieran entre ellas de forma tal que provoquen que la BD quede en un

estado no consistente; desde este punto de vista, podemos ver una transacción como una unidad lógica de concurrencia. Cuando ocurre un fallo que provoca la caída del sistema, justo en el momento en el que había varias transacciones en curso de ejecución, probablemente dejará erróneos los datos en la BD (estado inconsistente); en estas circunstancias, se debe garantizar que la BD pueda ser recuperada a un estado en el cual su contenido sea consistente, y por esto, una transacción también es considerada una unidad lógica de recuperación.

La idea clave es que una transacción debe ser atómica, es decir, que las operaciones que la componen se ejecuten en su totalidad, o no se ejecuten en lo absoluto.

LA INTEGRIDAD DE DATOS

Este concepto se refiere a la precisión, la coherencia, y la integridad de los datos de una organización. La integridad de un dato alude a un atributo o cualidad inherente a la información que se considera exacta, completa, homogénea, sólida, coherente, y confiable, con base en la intención de los creadores de las bases de datos.

También guarda relación con la seguridad y la normatividad. Cuando es segura, los datos estarán a prueba de fuerzas externas, pues las bases de datos son completas, precisas y fiables durante el tiempo, independientemente del número de veces que sean accedidas. Esta cualidad va ligada al propio dato, y no al lugar donde se almacena. Esta evita: datos duplicados, datos faltantes, datos alterados, datos incorrectos.

¿CUÁNDO USAR TRANSACCIONES?

Las bases de datos transaccionales son una de las primeras que se implementan en los sistemas de las empresas u organizaciones, puesto que dan apoyo a las tareas operativas de las mismas.

Es así que, entre las diferentes aplicaciones de las bases de datos transaccionales, encontramos las siguientes:

- Nos permiten obtener datos almacenados de manera muy rápida que, además, ofrecen una imagen actual de los procesos de la empresa. Por lo tanto, una de las aplicaciones que se da a éstas es la de obtener información para su posterior análisis y toma de decisiones tácticas.

- A través de ellas se pueden organizar los almacenes de datos de las empresas, puesto que permiten dotarles de un esquema común, y optimizarlos para el procesamiento de consultas complejas.
- Permiten contextualizar las transacciones llevadas a cabo por aplicaciones operativas, lo que otorga una visión más completa al análisis de la información.
- Cuando se integran con bases de datos analíticas en una sola plataforma, se consigue una mayor consistencia del procesamiento de transacciones y, por lo tanto, la obtención de información útil, como informes sobre tendencias de ventas, o el comportamiento de los clientes.

Ejemplos de uso común de transacciones

Uno de los ejemplos más comunes lo encontramos en las transferencias bancarias. Las bases de datos transaccionales se emplean aquí para validar, o deshacer la transferencia. Puesto que las transferencias bancarias se desarrollan en dos operaciones distintas: primero se descuenta el dinero en la cuenta origen, y luego se suma en la cuenta de destino; si la segunda operación falla, el dinero se devuelve a la cuenta de origen, y la transferencia no tiene lugar y, por lo tanto, no se registra en la base de datos. Para garantizar la atomicidad del sistema (es decir, para que no aparezca o desaparezca dinero), las dos operaciones deben ser atómicas, es decir, el sistema debe garantizar que, bajo cualquier circunstancia (incluso una caída del sistema), el resultado final es que, o se han realizado las dos operaciones, o no se ha realizado ninguna.

Otro ejemplo lo encontramos en los sistemas de venta en línea: como en el caso anterior, éstos registran una transacción, una venta y un ingreso de dinero para que ésta se produzca. Si hay algún error durante el proceso (por ejemplo, en la pasarela de pago), la transacción no se completa, es decir, no se produce la venta ni el descuento del dinero, y por lo tanto, no se registra en la base de datos.

INSTRUCCIONES BEGIN, COMMIT Y ROLLBACK

Como se ha definido, una transacción es una unidad atómica de trabajo que se realiza por completo, o no se efectúa en lo absoluto. Para fines de recuperación, el sistema necesita mantenerse al tanto

de cuando la transacción se inicia, termina, y se confirma o aborta. Es así como el gestor de recuperación se mantiene al tanto de las siguientes operaciones:

- Inicio de transacción (BEGIN): marca el principio de la ejecución de la transacción.
- Confirmar transacción (COMMIT): señala que la transacción culminó con éxito, y que cualquier cambio (actualizaciones) ejecutado se puede confirmar sin que esto se considere un peligro para la base de datos, y no se cancelará.
- Revertir (ROLLBACK): indica que la transacción culminó sin éxito, y que cualquier cambio o efecto que se pueda haber aplicado a la base de datos se debe cancelar. Es decir, que se ha alcanzado un fallo, y se debe restablecer la base de datos al punto de integridad.

Algorítmica habitual de una transacción

```
BEGIN;
// Conjunto de sentencias SQL
[COMMIT | ROLLBACK];
```

LA PROPIEDAD ACID (ATOMICIDAD, CONSISTENCIA, AISLAMIENTO, PERMANENCIA)

PROPIEDADES (deseables) DE UNA TRANSACCIÓN:

Se suele hacer referencia a éstas como las propiedades ACID (por sus iniciales en inglés).

1. Atomicidad: todas las operaciones de la transacción son ejecutadas por completo, o no se ejecuta ninguna de ellas (si se ejecuta la transacción, se hace hasta el final).

2. Consistencia: una transacción T transforma un estado consistente de la base de datos, en otro estado consistente, aunque T no tiene por qué preservar la consistencia en todos los puntos intermedios de su ejecución. Ejemplo: la transferencia de una cantidad de dinero entre dos cuentas bancarias.

3. Aislamiento (Isolation): una transacción está aislada del resto de transacciones. Aunque existan muchas transacciones ejecutándose a la vez, cualquier modificación de datos que

realice T está oculta para el resto hasta que sea confirmada (COMMIT). Es decir, para cualesquiera T1 y T2, se cumple que - T1 ve las actualizaciones de T2 después de que T2 realice COMMIT, o - T2 ve las modificaciones de T1, después de que T1 haga un COMMIT; pero nunca se cumplen ambas cosas al mismo tiempo.

4. Durabilidad: una vez que se confirma una transacción, sus actualizaciones sobreviven cualquier fallo del sistema. Las modificaciones ya no se pierden, aunque el sistema falle justo después de realizar dicha confirmación.

JOURNALING Y BLOQUEOS

El journaling es un mecanismo que permite implementar transacciones. También se conoce como «registro por diario», y se basa en llevar un journal o registro en el que se almacena la información necesaria para restablecer los datos afectados por una transacción por si esta falla.

Las aplicaciones más frecuentes se utilizan para implementar transacciones de sistemas de bases de datos, y para evitar la corrupción de estructuras de datos en las que se basan los sistemas de archivos modernos. Lo que persigue el journaling de sistemas de archivos, es evitar los engorrosos y largos chequeos de disco que efectúan los sistemas al apagarse bruscamente, ya que éste al arrancar solo deberá deshacer el journal para tener un sistema coherente de nuevo.

El bloqueo es el proceso por el cual un Sistema Gestor de Base de Datos, restringe el acceso a una fila en un entorno de varios usuarios. Cuando una fila o columna está bloqueada exclusivamente, no se permite que otros usuarios accedan a sus datos hasta que sea liberada. Esto garantiza que dos usuarios no puedan actualizar simultáneamente la misma columna en una fila.

Los bloqueos pueden ser muy costosos, desde la perspectiva de los recursos, y solo se deben usar cuando sea necesario para conservar la integridad de los datos. En una base de datos en la que cientos, o miles de usuarios, podrían estar intentando acceder a un registro cada segundo, el bloqueo innecesario podría provocar un rendimiento más lento en la aplicación.

En cualquier base de datos, generalmente la falta de administración de las transacciones produce problemas de contención y de rendimiento en sistemas con muchos usuarios. A medida que

aumenta el número de usuarios que obtienen acceso a los datos, adquiere importancia el hecho de que las aplicaciones utilicen las transacciones eficazmente.

Cada transacción solicita diferentes tipos de bloqueo en los recursos como, por ejemplo: filas, páginas, o tablas de los que depende la transacción. Estos bloqueos impiden que otras transacciones puedan modificar los recursos, de forma que esto provoque problemas para la transacción que solicita el bloqueo, y cada transacción los libera cuando ya no depende de los recursos bloqueados.