Administración de las transacciones.

**Propiedades de las transacciones**

No importa el cuidado con que se diseñe y cree una base de datos, es fácil que se dañe o destruya, a menos que haya controles apropiados de la concurrencia y técnicas de recuperación.

**Recuperación** de la base de datos es el proceso de restaurarla a su estado correcto en caso de falla. **El control de la concurrencia** es la capacidad de administrar procesos simultáneos que acceden a la base de datos sin que interfieran entre sí.

La **transacción** se puede considerar una unidad lógica de trabajo en la base de datos. Puede ser un programa completo, fragmento de programa o comando único. Involucra cualquier número de operaciones en la base de datos.

Hay dos formas de finalizar o terminar una transacción.

Si se ejecuta hasta el final con éxito, se dice que la transacción fue comprometida (commit), y la base de datos es llevada a un nuevo estado consistente.

La otra posibilidad es que la transacción no se ejecute con éxito. En este caso, la transacción es abortada. Si se aborta una transacción, es esencial que la base de datos se restaure al estado consistente en que estaba antes de comenzar la transacción. Es decir, dicha transacción fue deshecha, esto quiere decir que se retrocede y al mismo tiempo se deshace (rollback) cada una de las operaciones realizadas por la transacción hasta su inicio.

Una transacción comprometida no se puede abortar. Si se decide que la transacción comprometida fue un error, se debe ejecutar otra transacción compensadora para revertir sus efectos. Sin embargo, una transacción abortada que haya sido deshecha puede ser reiniciada en algún momento futuro y, en función de cuál haya sido la causa de la falla, podría ejecutarse exitosamente y ser comprometida.

Para garantizar que la base de datos mantiene un estado correcto a pesar de una falla de concurrencia o del sistema, todas las transacciones deben mostrar cuatro propiedades importantes, por lo general llamadas **ACID:**

* Atomicidad. La transacción es una sola unidad de “todo o nada”. Se ejecuta todo el conjunto de acciones o ninguna. Para garantizar esta propiedad, el DBMS debe ser capaz de cancelar transacciones que no terminen con éxito, y anular sus efectos sobre la base de datos. El subsistema de recuperación del DBMS mantiene un registro, llamado bitácora, de todas las transacciones escritas en la base de datos, el cual se utiliza al deshacer la transacción.
* Consistencia. El usuario es responsable de asegurar que su transacción, si se ejecutara por sí sola, deje a la base de datos en un estado consistente. Es trabajo del subsistema de control de concurrencia del DBMS garantizar la consistencia cuando se ejecutan al mismo tiempo transacciones múltiples.
* Aislamiento. Es posible que varias transacciones se ejecuten al mismo tiempo con sus operaciones intercaladas. La propiedad de aislamiento requiere que el efecto final sea como si las transacciones se hubieran ejecutado una después de otra en vez de ejecutarse en forma concurrente. Como cada transacción de manera aislada deja a la base de datos en un estado consistente, el resultado en conjunto de todas las transacciones también será un estado consistente. El sistema de control de concurrencia tiene que garantizar el aislamiento.
* Durabilidad. Si una transacción ha sido comprometida, el DBMS debe asegurar que sus efectos se registren de manera permanente en la base de datos, aun si el sistema fallara antes de que se hubieran escrito en la base de datos los registros involucrados por la transacción. El subsistema de recuperación es responsable de garantizar la persistencia de los datos, para lo que utiliza la bitácora de las transacciones.

El objetivo del control de la concurrencia es maximizar el caudal de procesamiento de transacciones y, al mismo tiempo, evitar la interferencia entre varios usuarios. El caudal de procesamiento de transacciones, el número de transacciones procesadas por unidad de tiempo, es una importante medida del trabajo que realiza un DBMS. Por lo regular, el caudal de procesamiento se reporta en transacciones por minuto.

*Problemas de interferencia*

* Actualización perdida*:* Una actualización perdida es el problema de interferencia más serio porque los cambios a una base de datos se pierden sin que nadie se dé cuenta. En una actualización perdida, la actualización de un usuario sobrescribe la de otro.
* Dependencia sin realización: Una dependencia sin realización sucede cuando una transacción lee los datos que escribe otra antes de que la otra transacción se realice. Una dependencia sin realización también se conoce como lectura sucia porque la provoca una transacción que lee datos sucios (sin realizar).
* Problemas que comprenden recuperaciones inconsistentes: El último problema comprende situaciones en las que la interferencia provoca inconsistencias entre varias recuperaciones de un subconjunto de datos. Todos los problemas de recuperaciones inconsistentes comprenden una transacción que lee y una segunda transacción que cambia la misma parte de la base de datos. El problema de resumen incorrecto es el más importante entre los que comprenden recuperaciones inconsistentes. Un resumen incorrecto ocurre cuando una transacción, mientras calcula la función de resumen, lee algunos valores antes de que otra transacción los cambie, pero lee otros valores después de que otra transacción los cambió. Un segundo problema que comprende recuperaciones inconsistentes se conoce como problema de lectura fantasma. Un problema de lectura fantasma se presenta cuando una transacción realiza una consulta con condiciones de registro. Luego, otra transacción inserta o modifica los datos que la consulta recuperaría. Un problema de lectura no repetible ocurre cuando una transacción lee el mismo valor más de una vez. Entre una y otra lectura de datos, otra transacción los modifica. La segunda recuperación contiene un valor diferente que la primera por el cambio que realiza otra transacción.

Los problemas de lectura no repetible y fantasma son ligeramente diferentes. Un problema de lectura no repetible ocurre si otro usuario cambia el valor de una columna de una fila de consulta, de modo que la consulta regresa un valor diferente en la siguiente ejecución. Un problema de lectura fantasma ocurriría al insertar una nueva fila que coincida con una condición en una consulta, de modo que esta última recupera una fila adicional en la siguiente ejecución. La diferencia clave es el requisito de la condición de la fila para el problema de lectura fantasma.