

Bases de Datos I

CLASE 6

Transacciones



Transacciones

➤ Transacción: colección de operaciones que forman una única unidad lógica de trabajo.

❖ Propiedades **ACID**

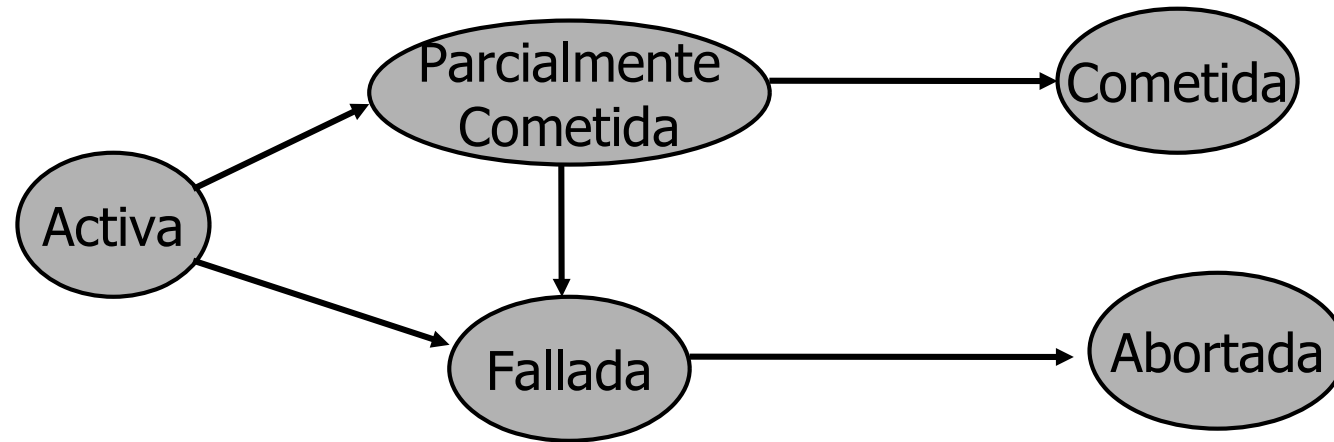
- Atomicidad: todas las operaciones de la transacción se ejecutan o no lo hacen ninguna de ellas
- Consistencia: la ejecución aislada de la transacción conserva la consistencia de la BD
- Aislamiento (isolation): cada transacción ignora el resto de las transacciones que se ejecutan concurrentemente en el sistema, actúa c/u como única.
- Durabilidad: una transacción terminada con éxito realiza cambios permanentes en la BD, incluso si hay fallos en el sistema.

Transacciones

- Estados de una transacción
 - ❖ Activa: estado inicial, estado normal durante la ejecución.
 - ❖ Parcialmente Cometida: después de ejecutarse la última instrucción
 - ❖ Fallada: luego de descubrir que no puede seguir la ejecución normal
 - ❖ Abortada: después de haber retrocedido la transacción y restablecido la BD al estado anterior al comienzo de la transacción.
 - ❖ Cometida: tras completarse con éxito.

Transacciones

➤ Diagrama de estado de una transacción



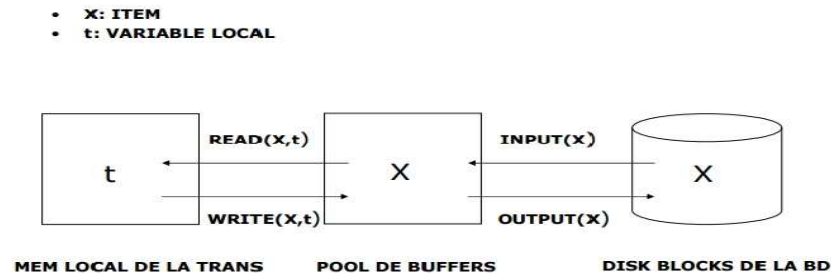
Transacciones

- Transacción abortada: que hacer
 - ❖ Reiniciar la transacción: cualquier error que no dependa de la lógica de la transacción (hardware o software)
 - ❖ Cancelar la transacción: error interno lógico en el programa que ejecuta la transacción.

Transacciones

➤ Modelo de transacción (Diferencia entre READ, WRITE y INPUT, OUTPUT)

```
READ ( A, a1)
a1 := a1 - 100;
WRITE( A, a1)
READ (B, b1)
b1 := b1 + 100;
WRITE(B, b1)
```



➤ Uso de transacciones:

- ❖ En sistemas monousuario
- ❖ En sistemas concurrentes
- ❖ En sistemas distribuidos

Fallos

➤ Primer Clasificación:

- ❖ Con pérdida de Información
- ❖ Sin pérdida de Información

➤ Tipos de fallos con perdida:

- ❖ Fallo en la transacción: dos errores
 - Lógicos: interno a la transacción
 - Del sistema: bloqueos (deadlock)
- ❖ Caída del sistema: (hardware, software (SO, DBMS))
- ❖ Fallo de disco

Fallos

- Algoritmos de tratamiento de fallos
 - ❖ Acciones llevadas a cabo durante el procesamiento normal de la transacción que permiten la recuperación ante fallos
 - ❖ Acciones llevadas a cabo después de ocurrir el fallo para restablecer el contenido de la BD a un estado que asegure ACID.

Fallo

➤ Estructura de almacenamiento

- ❖ Almacenamiento volátil

- ❖ Almacenamiento no volátil

- ❖ Almacenamiento estable

- Replicar la información en varios medios no volátiles independiente y actualizar controladamente para asegurar que se mantienen los datos ante todo caso.

Recuperación en caso de Fallo

- Ante un error, como proceder?
 - ❖ Re-ejecutar: no sirve
 - ❖ No Re-ejecutar: no sirve
 - ❖ Problema: modificar la BD sin seguridad que la transacción se va a cometer.
 - ❖ Solución: indicar las modificaciones antes de hacerlas efectivas → permite recuperar
- Métodos de recuperación
 - ❖ Basado en bitácora (log)
 - ❖ Doble Paginación.

Recuperación en caso de Fallo

- Recuperación basada en bitácora
 - ❖ Registro histórico: secuencia de actividades realizadas sobre la BD.
 - ❖ Contenido de la bitácora
 - <T iniciada>
 - <T, E, Va, Vn>
 - Identificador de la transacción
 - Identificador del elemento de datos
 - Valor anterior
 - Valor nuevo
 - <T Commit>
 - <T Abort>

Recuperación en caso de Fallo

- Operaciones sobre la BD deben almacenarse antes en la Bitácora!!
- Dos técnicas de bitácora
 - ❖ Modificación diferida de la BD
 - ❖ Modificación inmediata de la BD
- Modificación diferida
 - ❖ Las operaciones write se aplazan hasta que la transacción esté parcialmente cometida, en ese momento se actualiza la bitácora y la BD

Recuperación en caso de Fallo

- Dada la siguiente transacción
 - < T0 Start >
 - < T0, A, 900 >
 - < T0, B, 2100 >
 - < T0 Commit >
- Recién con T0 parcialmente cometida, entonces se actualiza la BD.
 - ✓ No se necesita valor viejo, se modifica todo al final o no se modifica.
- Ante un fallo, y luego de recuperarse:
 - ✓ REDO (Ti), para todo Ti que tenga un Start y un Commit en la Bitácora.
 - ✓ Si no tiene Commit entonces se ignora, dado que no llegó a hacer algo en la BD.

Recuperación en caso de Fallo

➤ Modificación inmediata:

- ❖ La actualización de la BD se realiza mientras la transacción está activa y se va ejecutando.
- ❖ Se necesita el valor viejo, pues los cambios se fueron efectuando.
- ❖ Ante un fallo, y luego de recuperarse:
 - REDO(T_i), para todo T_i que tenga un Start y un Commit en la Bitácora.
 - UNDO(T_i), para todo T_i que tenga un Start y no un Commit.

Recuperación en caso de Fallo

➤ Buffers de Bitácora

- ❖ Grabar en disco c/registro de bitácora insume gran costo de tiempo → se utilizan buffer, como proceder?
 - Transacción está parcialmente cometida después de grabar en memoria no volátil el Commit en la Bitácora.
 - Un Commit en la bitácora en memoria no volátil, implica que todos los registros anteriores de esa transacción ya están en memoria no volátil.
 - **Siempre** graba primero la Bitácora y luego la BD.

Recuperación en caso de Fallo

➤ Puntos de verificación:

❖ Ante un fallo, que hacer

- REDO, UNDO: según el caso

❖ Revisar la bitácora:

- Desde el comienzo?: probablemente gran porcentaje esté correcto y terminado.
- Lleva mucho tiempo.

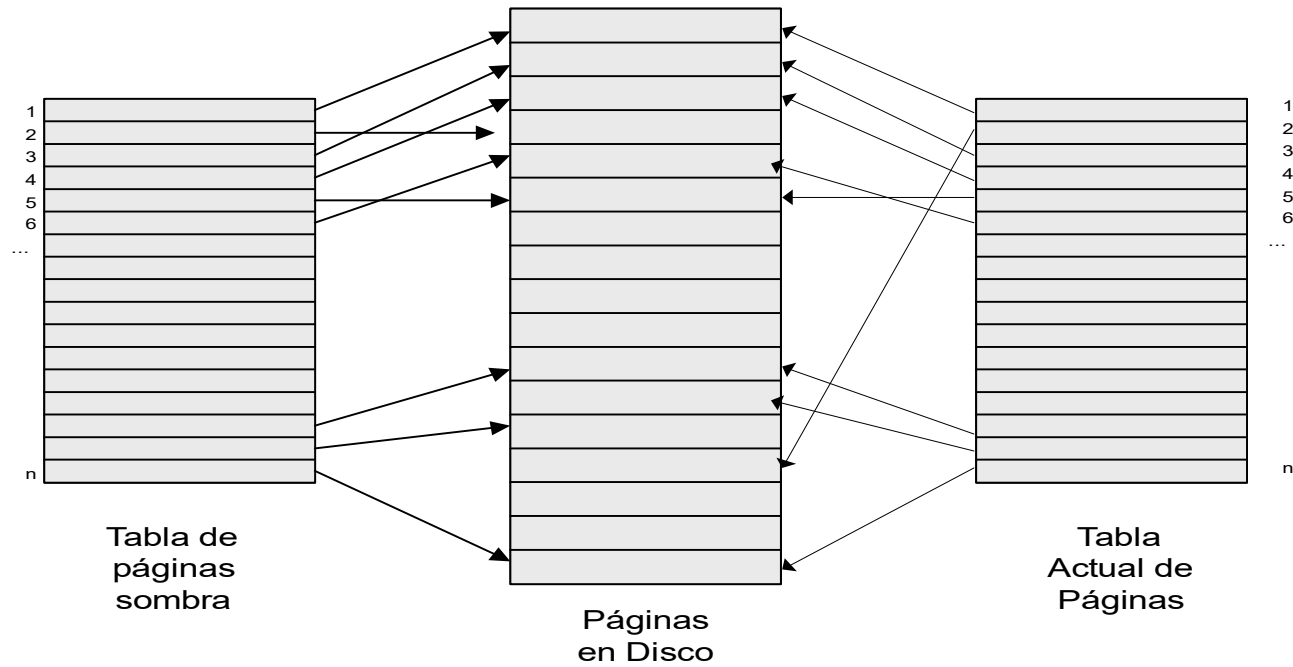
❖ Checkpoints (monousario)

- Se agregan periódicamente indicando desde allí hacia atrás todo OK.
- Cuan periódico?

Recuperación en caso de Fallo

- Doble Paginación (paginación en la sombra):
 - ❖ Ventaja: menos accesos a disco
 - ❖ Desventaja: complicada en un ambiente concurrente/distribuido.
 - ❖ N páginas equivalente a páginas del SO.
 - Tabla de páginas actual
 - Tabla de páginas sombra

Recuperación en caso de Fallo



Recuperación en caso de Fallo

- Ejecución de la operación *escribir*
 - ❖ Ejecutar **entrada**(X) si página i-ésima no está todavía en memoria principal.
 - ❖ Si es la primer escritura sobre la página i-ésima, modificar la tabla actual de páginas así:
 - Encontrar una página en el disco no utilizada
 - Indicar que a partir de ahora está ocupada
 - Modificar la tabla actual de página indicando que la i-ésima entrada ahora apunta a la nueva página

Recuperación en caso de Fallo

- En caso de fallo y luego de la recuperación
 - ❖ Copia la tabla de páginas sombra en memoria principal.
 - ❖ Abort automáticos, se tienen la dirección de la página anterior sin las modificaciones.

- Ventajas:
 - ❖ Elimina la sobrecarga de escrituras del log
 - ❖ Recuperación más rápida (no existe el REDO o UNDO).

Recuperación en caso de Fallo

➤ Desventajas:

- ❖ Sobrecarga en el compromiso: la técnica de paginación es por cada transacción.
- ❖ Fragmentación de datos: cambia la ubicación de los datos continuamente.
- ❖ Garbage Collector: ante un fallo queda una página que no es mas referenciada.

Recuperación en caso de Fallo

➤ Fallos con pérdida de memoria no volátil

❖ Qué pasa si el DR se rompe?. BACKUPS

- Características de los Backups.

❖ Almacenamiento de datos en memoria estable.

- Proceder con la bitácora igual que antes pero ahora entre DR y Memoria Estable.
- Otra opción: discos espejos.

Fin Clase 6