MODULO 7

ARCHIVOS DE LOG, DEFINICIÓN Y USOS
MÉTODOS DE RECUPERACIÓN

Marcela Russo
Laboratorio IV

UP Universidad de Palermo

Resguardo y recuperación

- **Un DBMS debe proporcionar los medios para la recuperación ante fallas de hardware o de software.**
- **El subsistema de respaldo y recuperación del DBMS es el responsable de la recuperación.**
- **Si en medio de una transacción ocurre una falla, el subsistema de recuperación debe asegurar que la base de datos sea restaurada al estado en el que se encontraba, antes del inicio de la transacción.**
- **También es necesario realizar copias de seguridad de los archivos físicos para los casos en los que ocurra una falla catastrófica.**
- **Decimos que una base de datos está en un estado consistente si obedece a todas las restricciones de integridad definidas sobre ella.**

PROTECCION DE DATOS ...Transacciones

- **Como ya hemos visto, existen las *UNIDADES LÓGICAS DE EJECUCIÓN* llamadas *TRANSACCIONES,* con las siguientes caracterísiticas:**

  - ✓ **Las transacciones terminan con éxito y son grabadas en la base de datos.**
  - ✓ **Las transacciones fallan y la base de datos debe ser restaurada para regresar al estado en que se encontraba antes del inicio de la transacción.**
  - ✓ **Una transacción incluye una o más operaciones de acceso a la base de datos.**
  - ✓ **Transacciones de lectura-escritura (read-write) son las que involucran operaciones de inserción, eliminación, modificación (actualización) o recuperación.**
  - ✓ **Transacciones de solo lectura (read only) son las que involucran solo operaciones de lectura.**

- **Las transacciones pueden declararse en los programas de aplicación utilizando begin transaction y end transaction. Un programa de aplicación puede contener más de una transacción.**
- **Las transacciones pueden ejecutarse desde un lenguaje de alto nivel como el lenguaje SQL.**
- **El DBMS es el responsable de asegurar que todas las operaciones en la transacción se completen con éxito, es decir que alcancen el commit, y su efecto se registre de forma permanente en la base de datos, o que la transacción no tenga ningún efecto sobre la base de datos (transacción abortada)**

PROTECCION DE DATOS ...Transacciones

- **Para fines de recuperación, el sistema necesita realizar un seguimiento de cada transacción, para saber cuando comienza, cuando finaliza y como lo hizo (commit o abort) . Por lo tanto, el subsistema de recuperación del DBMS debe realizar un seguimiento de las siguientes operaciones:**

  - ✓ **BEGIN_TRANSACTION: Indica el comienzo de la ejecución de la transacción.**

  - ✓ **READ o WRITE: especifican operaciones de lectura o escritura en la base de datos.**

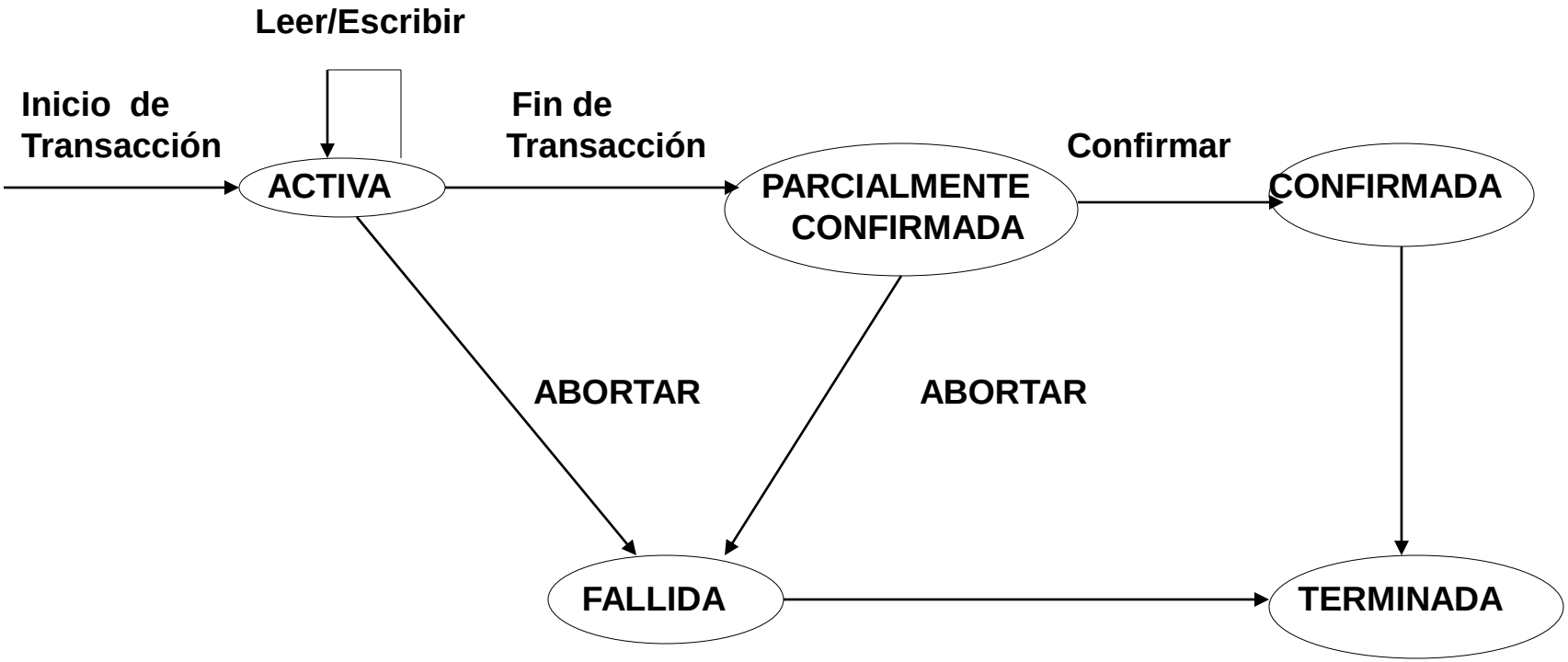
  - ✓ **END_TRANSACTION: indica que las operaciones de transacciones lectura y escritura han finalizado y marca el final de la ejecución de la transacción. Sin embargo, en este punto puede ser necesario comprobar si los cambios introducidos por la transacción se puede aplicar permanentemente a la base de datos (commit) o si la transacción tiene que ser abortada (abort) por alguna falla.**

  - ✓ **COMMIT_TRANSACTION: indica un final exitoso de la transacción para que cualquier cambio ejecutado por la transacción quede confirmado de forma segura en la base de datos y no se deshará.**

  - ✓ **ROLLBACK (ó ABORT): indica que la transacción ha finalizado sin éxito, para deshacer los cambios o efectos que la misma haya realizado sobre la base de datos.**

UP Universidad de Palermo

Diagrama de transición de Estados en la ejecución de una transacción



- **Una transacción pasa a un estado activo inmediatamente después de que comience la ejecución, donde puede ejecutar sus operaciones de LECTURA y ESCRITURA.**

- **Cuando finaliza la transacción, pasa al estado PARCIALMENTE CONFIRMADO. En este punto:**

    - **Algunos tipos de protocolos de control de concurrencia pueden hacer comprobaciones adicionales para ver si la transacción puede alcanzar, o no, el commit.**
    - **Algunos protocolos de recuperación necesitan asegurarse de que una falla del sistema no resulte en la imposibilidad de registrar los cambios de la transacción de forma permanente (normalmente registrando los cambios en el sistema de log).**
    - **Si estas comprobaciones tienen éxito, la transacción ha alcanzado su punto de compromiso y entra en el estado CONFIRMADO. Esto implica que la transacción ha concluido su ejecución con éxito y todos sus cambios deben ser registrados permanentemente en la base de datos, incluso si ocurre una falla en el sistema.**
    - **Sin embargo, una transacción puede pasar al estado FALLIDA si una de las comprobaciones falla o si la transacción es abortada durante su estado ACTIVA. Es posible que la transacción tenga que revertirse para deshacer el efecto de sus operaciones WRITE en la base de datos.**

- **El estado TERMINADA corresponde a la transacción saliendo del sistema.**

- **La información de la transacción que se mantiene en las tablas del sistema mientras la transacción se ha estado ejecutando y se elimina cuando termina la transacción.**

- **Las transacciones fallidas o abortadas se pueden reiniciar más tarde, ya sea automáticamente o después de que el usuario las haya vuelto a enviar, como transacciones nuevas.**

UP Universidad de Palermo

Fallas que pueden ocurrir

- **En los sistema pueden ocurrir fallas que ponen en riesgo la integridad y la existencia misma de la BD, provocando la perdida de los datos.**

- **Las fallas pueden ocurrir en las transacciones, en el sistema o en los medios, como por ejemplo:**

  - ✓ **Fallas de la computadora, provocando la caída del sistema (Ej. Falla en la memoria)**
  - ✓ **Fallas por error de la transacción o del sistema (Ej. División por 0)**
  - ✓ **Fallas por excepciones no programadas (Ej. datos inexistentes para realizar la transacción).**
  - ✓ **Fallas en el control de concurrencia (Ej. Para evitar abrazos mortales)**
  - ✓ **Fallas de disco (Ej. bloqueo del cabezal de lectura/escritura del disco)**
  - ✓ **Fallas por catástrofes o problemas físicos (Ej. fallas de energía o aire acondicionado, incendio, robo, sabotaje, sobreescritura discos o cintas por error, etc.)**

- **Los problemas aparecen en las operaciones de actualización de la base de datos, la LECTURA NO ES causal de problemas.**

- **El sistema debe mantener suficiente información para una rápida recuperación ante la ocurrencia de una falla.**

UP Universidad de Palermo

Propiedades deseables de una transacción y los subsistemas del DBMS:

- **Las transacciones deben poseer varias propiedades, a menudo denominadas propiedades ACID; deben hacerse cumplir mediante el control de concurrencia y los métodos de recuperación del SGBD**

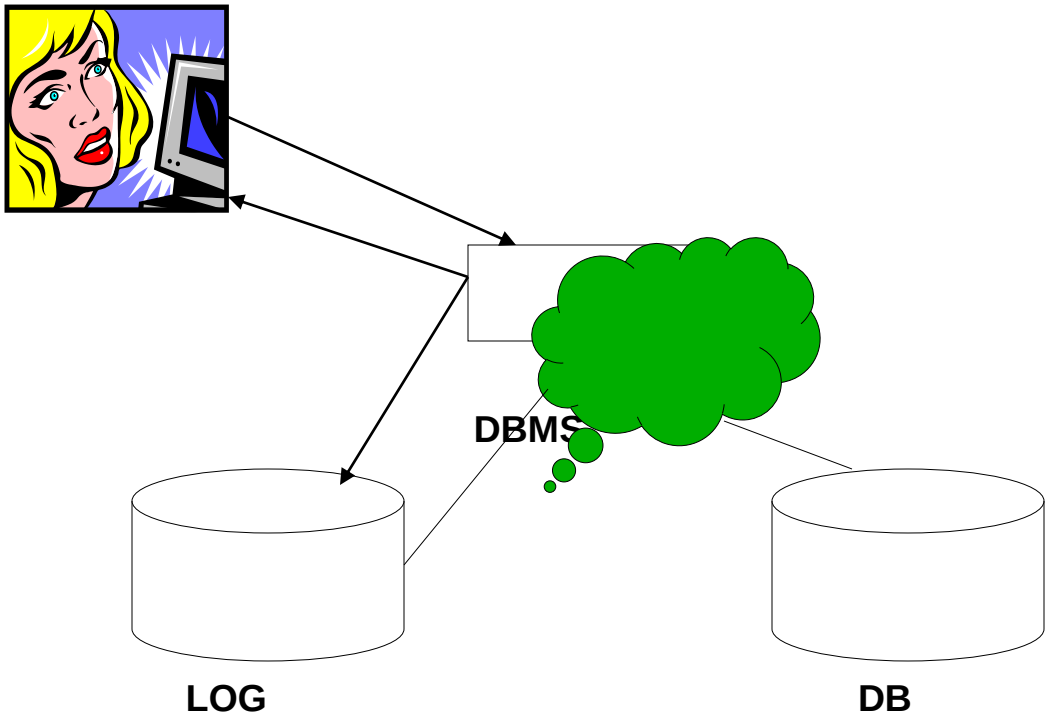
- **La base de datos que cumple con la prueba ACID, asegura que los cambios realizados por las transacciones finalizadas y confirmadas con éxito, perduran aunque el sistema falle con posterioridad.**

Prueba ACID	Funcionalidad	Subsistema de
A tomicidad	Se ejecuta todo o nada	Respaldo-Recuperación
C onsistencia	La base de datos pasa de un estado consistente a otro estado consistente	Integridad, pueden intervenir los programadores
a*I*slamiento	La transacción no muestra los cambios que produce hasta que finaliza	Control de Concurrencia
D urabilidad	Una vez que la transacción finaliza con éxito y es confirmada, los cambios perduran aunque el sistema falle después	Respaldo-Recuperación

- **El subsistema de control de concurrencia y el subsistema de respaldo-recuperación, están integrados dentro del funcionamiento base de datos en tiempo de ejecución.**

Sistema de LOGS

- *El Sistema de LOGs de la base de datos es el registro de todas las operaciones que se realizan en cada transacción*



Sistema de LOG

- Para poder recuperarse de las fallas que afectan las transacciones, el DBMS mantiene un log para realizar un seguimiento de todas las operaciones de transacción que afectan los datos de la base de datos, así como otra información de la transacción que pueda ser necesaria para permitir recuperación de fallas.

- El log es un archivo secuencial, se mantiene en el disco y se lo respalda periódicamente en otro medio de almacenamiento, como por ejemplo en cinta, para protegerlo y evitar que se pierda.

- Los registros se van agregando al final del último registro grabado. No se ve afectado por ningún tipo de falla a excepción de fallas de disco o alguna catástrofe.

- Generalmente, uno o más buffers de la memoria principal, llamados log buffers, mantienen la última parte del archivo de log, para que las entradas de log se agreguen primero al log buffer de la memoria principal. Cuando el log buffer se completa, o cuando ocurren determinadas condiciones, el log buffer que está en la memoria se agrega al final del archivo de log que está en el disco.

LOGS RECORDS

- **Se denomina *logs records* a los tipos de entradas que se escriben en el archivo de log y la acción correspondiente para cada registro de log.**

- **En estas entradas, T identifica un transacción única generada automáticamente por el sistema para cada transacción y que se utiliza para identificarla:**

1. *Start_transaction, T*: Indica que la transacción T inició la ejecución.

2. *Write_item, T, X, Old_value, New_value*: Indica que la transacción T cambió el valor del ítem X de la base de datos de old_value a new_value

3. *Read_item, T, X:* Indica que la transacción T ha leído el valor del ítem X de la base de datos

4. *Commit, T:* Indica que la transacción T se ha completado satisfactoriamente, y confirma que lo ejecutado puede ser registrado permanentemente en la base de datos, es decir que se puede ejecutar el commit.

5. *[Abort, T]*: Indica que la transacción T fue abortada

LOGS RECORDS

- **Los protocolos de recuperación que evitan los rollback en cascada , incluidos en casi todos los protocolos prácticos, no requieren que las operaciones READ sean escritas en el sistema de logs.**

- **Si el log también se utiliza para otros fines, como por ejemplo la auditoría, la cual se utiliza para hacer un seguimiento de todas las operaciones de la base de datos, las operaciones de READ pueden incluirse en el log.**

- **Algunos protocolos de recuperación requieren entradas de ESCRITURA más simples, que solo incluyen new_value u old_value en lugar de incluir ambos.**

- **Debido a que el log contiene un entrada por cada operación de ESCRITURA que cambia el valor de un Ítem en la base de datos, es posible deshacer (UNDO) el efecto de estas operaciones de ESCRITURA de una transacción T rastreando hacia atrás a través del log y restableciendo todos los datos cambiados por una operación WRITE de la transacción a sus valor anterior (old_value).**

- **También puede ser necesario rehacer (REDO) una operación si la transacción tiene sus actualizaciones registradas en el registro, pero se produce una falla antes de que el sistema pueda asegurar que todos los nuevos_valores (new_value) se han escrito desde los búferes de memoria principal a los archivos físicos que soportan a la base de datos**

UP Universidad de Palermo

Punto de COMMIT de una transacción

- **Una transacción T alcanza su punto de COMMIT cuando todas sus operaciones se han ejecutado con éxito y el efecto de todas las operaciones ejecutadas en la transacción se han registrado en el log.**

- **Se dice que la transacción fue “comiteada” y el efecto debe registrarse permanentemente en los archivos físicos de la base de datos.**

- **La transacción escribe un punto de COMMIT [commit, T] en el LOG.**

- **Si ocurre una falla del sistema, es posible buscar en el LOG todas las transacciones que han escrito un [start_transaction, T] dentro del mismo, pero que no han registrado aún el [commit, T], pueden tener que revertirse para deshacer (UNDO) su efecto en la base de datos durante el proceso de recuperación.**

- **Las transacciones que han registrado su [commit, T] en el LOG, también registraron todas sus operaciones de ESCRITURA en el LOG, por lo que su efecto en la base de datos se puede rehacer (REDO) desde los registros de LOG.**

- **El archivo de LOG debe mantenerse en el disco.**

UP Universidad de Palermo

COMMIT de una transacción

- **Actualizar un archivo de disco implica copiar el bloque apropiado del archivo del disco a un búfer en memoria principal, actualizando el búfer en la memoria principal y copiando el búfer a disco.**

- **Es común mantener uno o más bloques del archivo de log en los bufferes de la memoria principal, llamado log buffer, hasta que se llenen con entradas en el log y luego volver a escribirlos en el disco una sola vez, en lugar de escribirlos en el disco cada vez que se agrega una entrada en el log.**

- **Esto ahorra la sobrecarga de múltiples escrituras del mismo buffer de log en disco.**

- **Ante un crash del sistema, sólo las entradas de log que hayan sido escritas nuevamente en el disco se consideran en el proceso de recuperación si el contenido de la memoria principal se pierde. Por lo tanto, antes que una transacción alcance su punto de commit, deberá escribirse en el disco, cualquier parte del log que aún no haya sido escrito en el mismo.**

- **El proceso mencionado se denomina force-writing (escritura forzada)**

UP Universidad de Palermo

PARA que los LOG sean UTILES DEBEN RESIDIR EN UN ALMACENAMIENTO ESTABLE

- **La escritura se realiza de dos formas:**
  - **SINCRONA: por cada registro en el LOG, la página de registro se guarda en un almacenamiento estable**
  - **ASINCRONA: Las páginas de registro se mueven en forma periódica o cuando el LOG se completa**
- **Cada vez que una transacción va a realizar una operación en la BD, se agrega un registro en el LOG.**
- **Típicamente esta actualización se realiza en un buffer de memoria, por lo tanto cuando una transacción se confirma:**
  - **Primero se baja a disco todo el LOG de la transacción que aún no haya sido guardado**
  - **Luego se actualiza la BD con los efectos de la transacción**
- **NUNCA SE DEBEN PERDER A LA VEZ LA BD Y EL LOG, por lo tanto:**
  - **Deben estar en discos distintos**
  - **Deben realizarse respaldos independientes de uno y otro**

UP Universidad de Palermo

DBMS- Políticas específicas de reemplazo de buffers

- **El DBMS cache mantiene las páginas del disco que contienen información que este siendo procesada, en los buffers de memoria principal.**

- **Si todos los buffers en el DBMS Cache se encuentran ocupados y nuevas páginas del disco deben cargarse en la memoria principal desde el disco, se aplicará una política de reemplazo de páginas para seleccionar los buffers que serán reemplazados.**

- **Las políticas o métodos de reemplazo son:**

  - **Separación de Dominios**
  - **Hot Set (Configuración en caliente)**
  - **DBMIN**

Protocolo de recuperación y algoritmos de recuperación

- **El log del sistema generalmente mantiene información sobre los cambios que se aplicaron a los ítems de datos por parte de varias transacciones y la recuperación de errores de transacción generalmente significa que la base de datos se restaura a el estado consistente más reciente antes del momento de la falla.**

- **Un esquema de recuperación y categorización de algoritmos de recuperación típica se puede resumir brevemente de la siguiente manera:**

  - **Si una gran parte de la base de datos se encuentra dañada debido a una falla catastrófica, como un crash del disco, el método de recuperación restaura una copia de la base de datos de la que se realizó una copia de seguridad en otro almacenamiento (ejemplo: en cinta) y reconstruye una versión al estado más actual volviendo a aplicar, o rehaciendo las operaciones de transacciones que alcanzaron el commit, desde el log resguardado, hasta el momento de la falla.**

  - **Si la base de datos en el disco no está físicamente dañada y no ocurrió una falla catastrófica, la estrategia de recuperación es identificar cualquier cambio que pueda causar una inconsistencia en la base de datos. Por ejemplo, una transacción que ha actualizado algunos ítems de la base de datos en el disco pero no ha sido confirmado necesita que sus cambios se reviertan deshaciendo sus operaciones de escritura. También puede ser necesario rehacer algunas operaciones, para restaurar la base de datos a un estado consistente; por ejemplo, si la transacción fue confirmada, pero algunas de sus operaciones de escritura aún no se han sido escrito en el disco.**

  - **Para fallas no catastróficas, el protocolo de recuperación no necesita una copia de la base de datos de la que se realizó una copia de seguridad en otro almacenamiento de archivo completa de la base de datos. Más bien, las entradas mantenidas en el log del sistema en línea en el disco, se analizan para determinar la acciones apropiada para la recuperación.**

LOGS: Algoritmos de recuperación

➢ ***Si hubo un desastre (se pierde el disco, etc.) entonces:***

- **Se debe recuperar el último respaldo conocido de la BD (último resguardo válido)**
- **Se debe recuperar el LOG hasta donde se pueda**
- **Se deben rehacer (redo) todas las operaciones indicadas en el LOG desde el momento en se hizo el respaldo de la DB**

➢ ***Si la falla fue menor (corte de energía, falla de la red, etc.)***

- **Puede ser que haya que deshacer (undo) cambios ya realizados.**
- **Puede ser que haya que rehacer cambios que no se hayan confirmado**

UP Universidad de Palermo

Algoritmos de recuperación:

- **Conceptualmente, podemos distinguir dos políticas principales para la recuperación de situaciones no catastróficas ante fallas en las transacciones**

  - **Actualización Diferida: Cada transacción trabaja en un área local de disco o memoria y recién se baja al disco después que la transacción alcanza el COMMIT. Si hay un Abort o una Falla, no es necesario deshacer ninguna operación (No UNDO/REDO)**

  - **Actualización Inmediata: La base de datos se actualiza antes que la transacción alcance el commit. Si hay un Abort o una Falla, se deben deshacer las operaciones de la transacción. Siempre se graba primero el log para garantizar la recuperación**

UP Universidad de Palermo

LOGS: BeFore Image (old_value) y AFter Image (new_value)

- **Para que el mecanismo de recuperación sea viable, es necesario considerar al menos dos valores para cada ítem:**

  - ✓ **Before Image (BFIM): El valor del ítem antes de ser actualizado por una transacción.**

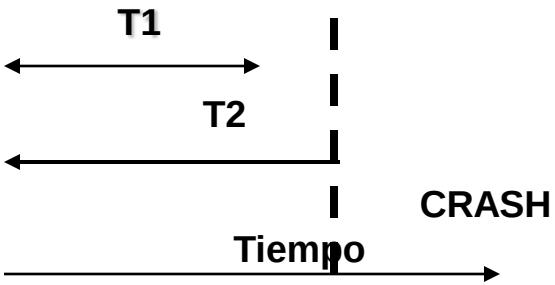
  - ✓ **After Image (AFIM): El valor del ítem después de ser actualizado por una transacción.**

- **De esta forma, los registros del Log pueden estar clasificados en:**

  De Undo: Contienen la operación y el BFIM
  De Redo: Contienen la operación y el AFIM
  Combinados: Contienen la operación y los dos. Se usan en estrategias UNDO/REDO.

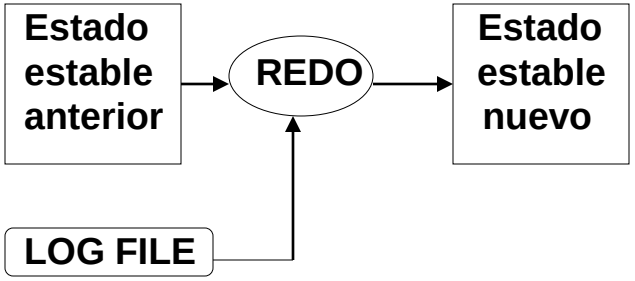
UP Universidad de Palermo

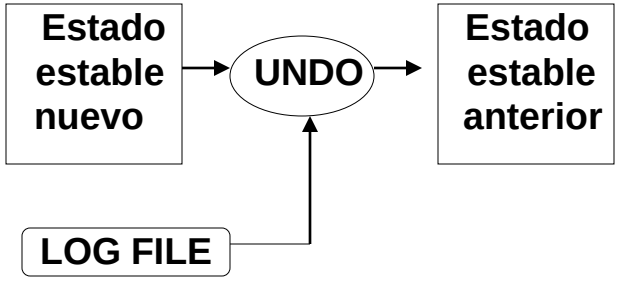
LOGS



- T1 terminó con éxito (COMMIT).
- T2 no pudo concluir.
- Es probable que los cambios de T1 no hayan sido escritos a la BD estable y por lo tanto se deben realizar todas las operaciones de T1 nuevamente (REDO)

- Por otra parte es posible que otras operaciones de T2 hayan pasado al almacenamiento estable, por lo que es necesario deshacer las mismas (UNDO).





RECUPERACION

- **CHECKPOINTS:**
  - **Es el punto en el cual las operaciones en la BD, sincronizan las páginas en el disco con las páginas en la memoria compartida y conjunto de buffer.**
  - **Cuando un CHECKPOINT se completa, se completan las operaciones físicas y la base de datos está físicamente consistente.**
  - **Registro del Log que indica que todos los buffers modificados de la base fueron actualizados al disco.**

ESTABLECER PUNTOS DE REVISION IMPLICA:

- **Grabar físicamente el contenido de los buffers de datos a la base de datos física.**
- **Grabar físicamente un registro de punto de revisión en el archivo de log.**
- **Los puntos marcados como CHECKPOINT, permiten la recuperación de la base de datos en línea, es decir, después de la caída del sistema se obtiene la dirección del registro más reciente y se recorre el archivo de log desde el punto marcado como checkpoint.**
- **Ninguna transacción que aparece en el Log antes que el checkpoint necesita rehacerse (Redo).**

UP Universidad de Palermo

RECUPERACION

- **CHECKPOINTS: El checkpoint consiste de los siguientes pasos:**

  - ✓ **Suspender la ejecución de todas las transacciones.**
  - ✓ **Grabar todos los buffers modificados en el disco**
  - ✓ **Registrar el checkpoint en el log y grabar el Log en disco.**
  - ✓ **Permitir la continuación de las transacciones**

  Write-Ahead Loggin

  - ✓ **Es una estrategia en la cual el mecanismo de recuperación garantiza que el BFIM está en el Log y el Log está en el disco ANTES que el AFIM actualice el BFIM en la base en el disco.**

  - ✓ **Un protocolo WAL(*Write-Ahead Loggin)* podría ser el siguiente:**
    - ✓ **Un AFIM de un ítem no puede actualizar el BFIM de ese ítem hasta que todos los registros del Log de tipo Undo para esa transacción haya sido escritos en el disco.**
    - ✓ **Nunca se puede completar el commit de una transacción hasta que todos los registros de Log (Undo o Redo) hayan sido escritos en el disco.**

  - ✓ **Para Implementar esto, se deben llevar listas de transacciones activas, confirmadas y abortadas.**

RECUPERACION

- **ROLLBACK:**
  - ✓ **Se debe realizar cuando una transacción aborta o no termina**
  - ✓ **Es la recuperación de todos los BFIM's de los ítems que modificó esa transacción y de todas las transacciones que leyeron de la que abortó. (Abortos en Cascada)**
  - ✓ **Lo Abortos (o Rollbacks) en Cascada pueden consumir mucho tiempo por lo que deben evitarse garantizando historias estrictas.**

RECUPERACION

- **ALGORITMOS DE RECUPERACION**

✓ **Cualquier algoritmo de recuperación debería implementar las siguientes operaciones o procedimientos:**
  - **Recuperación: Indica cual es la estrategia general de recuperación**
  - **Undo: Indica cómo se debe hacer el undo de una operación.**
  - **Redo: Indica como se debe hacer el redo de una operación.**

✓ **Es fundamental que la operación de Redo de los mismos resultados si ejecuta varias veces.**

✓ **En general, se asume que se están generando historias estrictas y serializables.**

RECUPERACION

- **RECUPERACION BASADA EN ACTUALIZACION DIFERIDA**

✓ **Idea Básica: Demorar la escritura de la base en el disco hasta que la transacción alcance el commit.**

✓ **Para esto, durante la ejecución la actualización se realiza en el Log y en los buffers o en un área local a la transacción.**

✓ **El protocolo tiene dos reglas básicas:**

- **Los cambios realizados por una transacción T nunca son grabados en el disco hasta que la T alcanza el commit.**
- **Una transacción T nunca puede alcanzar el commit hasta que grabó todas sus operaciones de actualización en el Log y el log fue grabado en el disco**

UP Universidad de Palermo

RECUPERACION

- **RECUPERACION BASADA EN ACTUALIZACION INMEDIATA**

✓ **Idea Básica: grabar en el disco sin esperar al commit.**

✓ **Siempre se trabaja con la estrategia WAL (grabar el log antes que los datos).**

✓ **Dos familias de algoritmos::**

- **Undo/No-Redo: hay que garantizar que todas las modificaciones efectivamente fueron grabadas en la base.**
- **Undo/Redo: No hay que garantizar nada en particular.**

✓ **Undo/Redo es más complejo.**

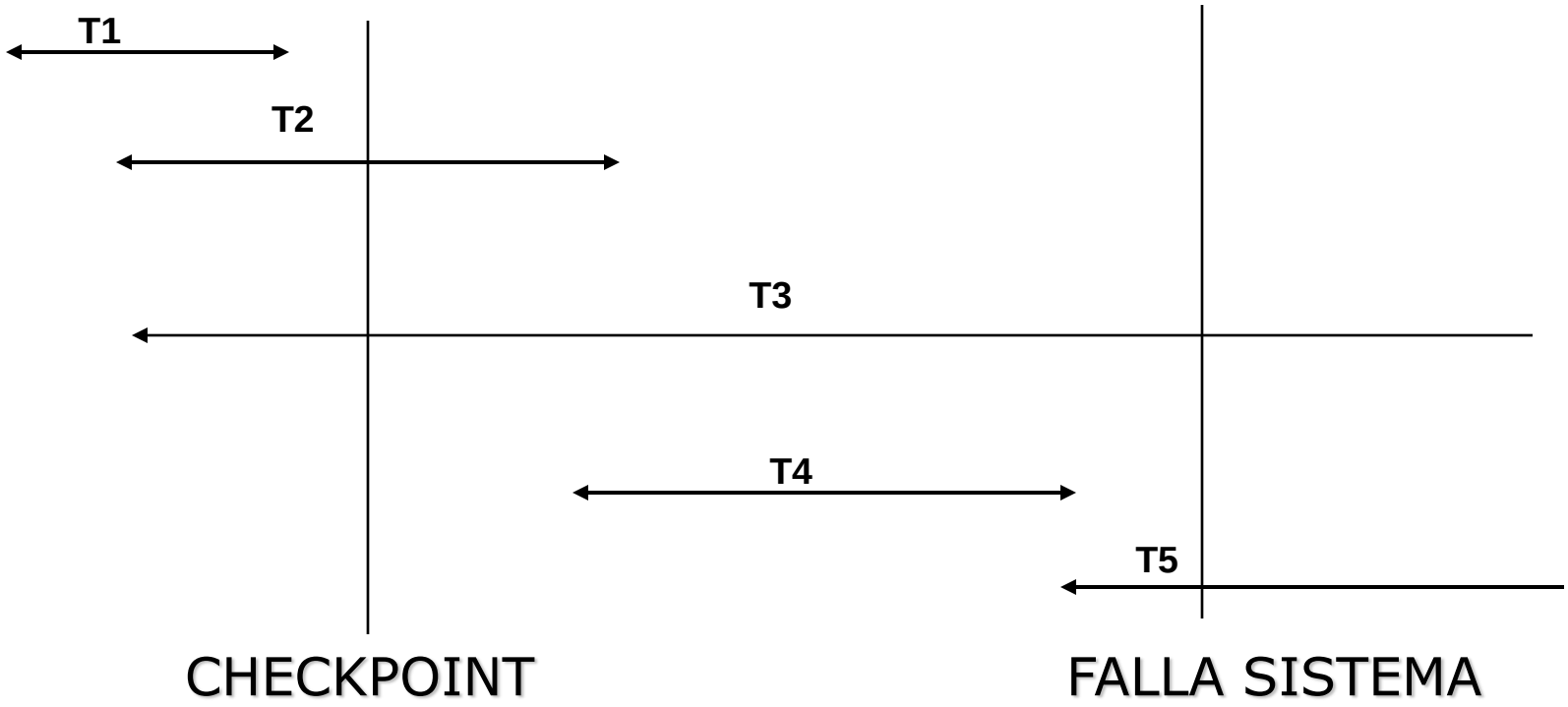
✓ **Se asume que se generan historias estrictas y serializables.**

RECUPERACION

La transacción T1 no se ve afectada por la falla del sistema, ni por el proceso de recuperación, por haberse completado antes del último punto de recuperación.

Las transacciones t2 y t4, a pesar de haber terminado no han sido grabadas en la base de datos, ya que esto ocurriría en un checkpoint, corresponde REDO

Las transacciones t3 y t5 deberán deshacerse (UNDO) ya que no han concluído.



RECUPERACION 2FC

Se utiliza para el control sobre transacciones distribuidas, el protocolo posee dos faces:

1. **El coordinador hace que todos los participantes estén listos para escribir los resultados en la BD**

2. **Todos escriben el resultado en la BD**

La transacción se completa a través de un compromiso global:

1. **El coordinador aborta una transacción si y solamente si al menos un participante indica que se aborte.**

2. **El coordinador hace un commit de la transacción si y solo si todos los participantes indican que se haga un commit.**