



Diseño y Gestión de Bases de Datos

Tema 3

Recuperación de Bases de Datos

Profesoras: Laura Mota y Pedro Valderas

DOCENCIA VIRTUAL

Finalidad:

Prestación del servicio Público de educación superior (art. 1 LOU)

Responsable:

Universitat Politècnica de València.

Derechos de acceso, rectificación, supresión, portabilidad, limitación u oposición al tratamiento conforme a políticas de privacidad:

<http://www.upv.es/contenidos/DPD/>

Propiedad intelectual:

Uso exclusivo en el entorno de aula virtual.

Queda prohibida la difusión, distribución o divulgación de la grabación de las clases y particularmente su comparación en redes sociales o servicios dedicados a compartir apuntes.

La infracción de esta prohibición puede generar responsabilidad disciplinaria, administrativa o civil



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Recuperación en bases de datos.

Objetivos:

- ✓ Estudiar las herramientas y estrategias utilizadas en los SGBD para la recuperación de transacciones en un entorno **monousuario**.

2

En este tema, el problema de la recuperación en bases de datos se estudia en un entorno monousuario. De momento se ignora la ejecución concurrente de transacciones, este caso se estudiará en el Tema 4.

Recuperación en bases de datos.


1 Recuperación de transacciones: introducción.

2 Estrategias de actualización en BD.

3 Estrategias de recuperación en BD.

1 Recuperación de transacciones: introducción.

Las operaciones de acceso a una BD se organizan en transacciones

TRANSACCIÓN  Secuencia de operaciones de acceso a la base de datos (consulta o actualización) que constituyen una unidad de ejecución.

Procesar correctamente una transacción significa:

(a) todas las operaciones de la transacción se ejecutan con éxito y sus cambios (actualizaciones) quedan grabados permanentemente en la base de datos,

o bien

(b) la transacción no tiene ningún efecto en la base de datos.

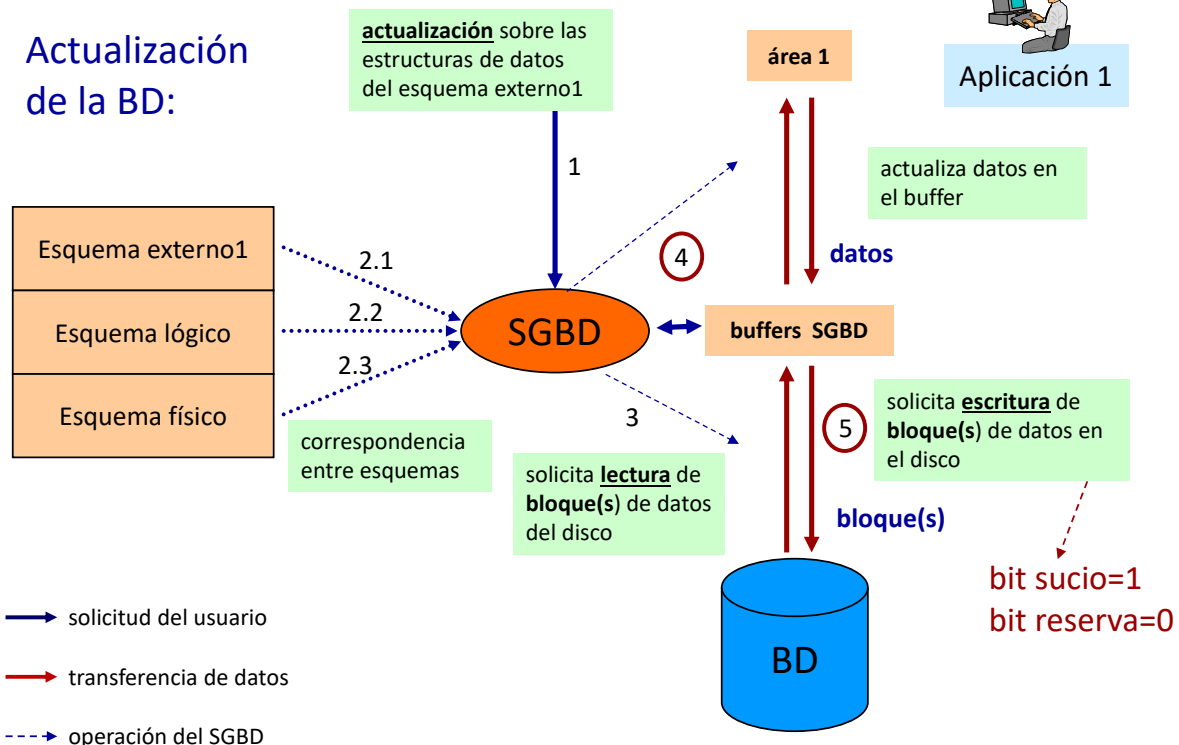
4

En el Tema 2, se ha estudiado el concepto de transacción y lo que significa su correcta ejecución en un SGBD: procesar (ejecutar) correctamente una transacción significa cumplir con el **objetivo (a)** o con el **objetivo (b)**, de acuerdo con el concepto de transacción como **unidad de ejecución**.

De estos dos objetivos, es relevante fijarse en que en el objetivo (a) la transacción no se considera ejecutada correctamente hasta que sus cambios (actualizaciones) han sido grabados en la base de datos en **disco**. A este respecto, es importante recordar el diagrama de pasos para la ejecución de una operación de actualización en un SGBD, que aparece en la transparencia siguiente (Tema 1): recuérdese el desajuste temporal entre la ejecución de la operación de actualización (paso 4) y la actualización de la base de datos (paso 5).

1 Recuperación de transacciones: introducción.

Actualización de la BD:



5

El SGBD dispone de un directorio de buffers para saber qué bloques de disco (de la BD) están almacenados en los buffers de memoria principal asignados. Este directorio contiene entradas de la forma: <dirección de bloque de disco, ubicación de buffer>.

Cada bloque de datos almacenado en un buffer dispone de un **bit de sucio** que indica si el bloque ha sido actualizado por alguna transacción (valor: 1) o no (valor: 0). Cuando los bloques son reemplazados en el buffer por otros, sólo deben ser transferidos a disco los bloques que tienen el bit de sucio a 1.

Cada bloque de datos almacenado en un buffer dispone también de un **bit de reserva** que indica si el bloque puede ser transferido al disco (valor: 0) o debe permanecer todavía en el buffer (valor: 1). El bit de reserva es útil en algunas estrategias de actualización de BD que se estudiarán en el punto 2 de este tema (actualización diferida).

Sólo pueden ser transferidos a disco aquellos bloques cuyo bit de ensuciado está a 1 y su bit de reserva está a 0.

1 Recuperación de transacciones: introducción.

Procesamiento de transacciones: objetivos (a) y (b)

- ✓ Transacciones confirmadas : **COMMIT** (usuario) + confirmación SGBD

Las actualizaciones de una transacción confirmada definitivamente por el SGBD deben quedar grabadas permanentemente en la base de datos → **objetivo (a)**

- ✓ Transacciones anuladas (usuario o SGBD):

- usuario: **ROLLBACK**
- SGBD: comprobación de RI, control de la concurrencia, errores en la ejecución, ...

Las actualizaciones de una transacción anulada no deben quedar grabadas en la base de datos → **objetivo (b)**

- ✓ Transacciones interrumpidas (fallo):

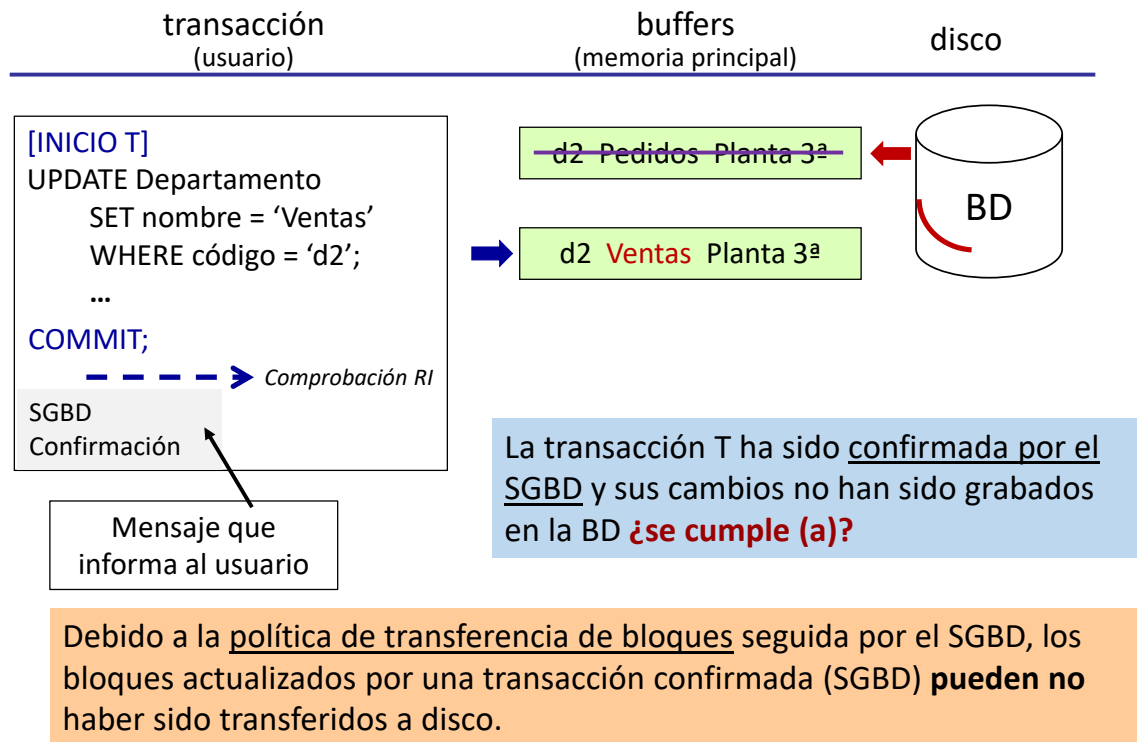
Las actualizaciones de una transacción interrumpida no deben quedar grabadas en la base de datos → **objetivo (b)**

6

Como se vio en el Tema 2, en el procesamiento de transacciones, en un SGBD, las transacciones pueden acabar con **confirmación** o con **anulación**, pero también pueden ser **interrumpidas** durante su ejecución por la ocurrencia de un fallo.

Cada forma de acabar una transacción tiene implicaciones distintas en la base de datos: las transacciones confirmadas por el SGBD deben quedar grabadas en la base de datos (objetivo (a)), mientras que las transacciones anuladas o interrumpidas no deben dejar ningún efecto sobre la base de datos (objetivo (b)).

1 Recuperación de transacciones: introducción.



7

En el ejemplo, se muestra una transacción que finaliza con confirmación del usuario (COMMIT). La comprobación posterior de las restricciones de integridad (en modo diferido) y su cumplimiento conducen al SGBD a confirmar definitivamente la transacción.

Según la idea de ejecución correcta de transacciones, esta transacción se debe ejecutar cumpliendo el objetivo (a). Como se puede observar, la actualización de la transacción (UPDATE) se ha ejecutado correctamente, pero el bloque actualizado no se ha transferido a disco, y permanece en el buffer de memoria principal **después de la confirmación del SGBD**, por lo que el objetivo (a) no se ha cumplido. La transacción no se ha ejecutado correctamente.

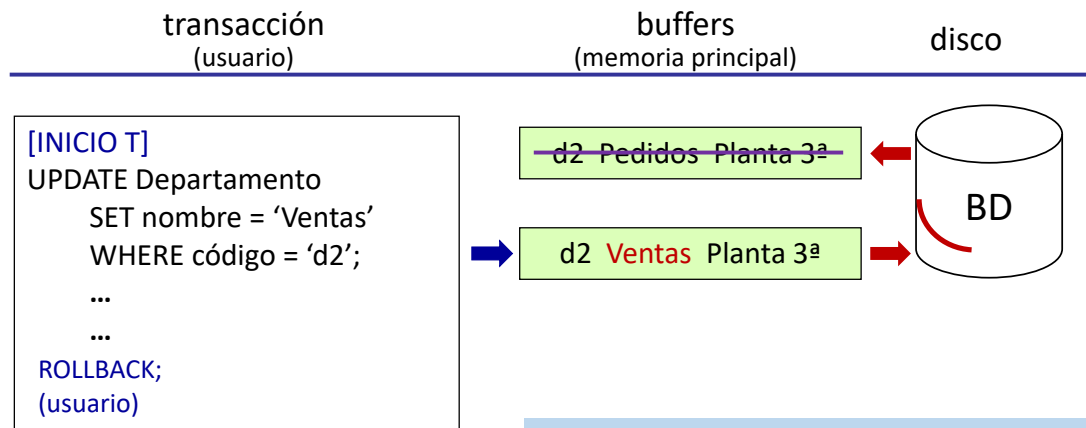
¿Por qué el SGBD no transfiere el bloque actualizado a disco, antes de confirmar definitivamente la transacción del usuario y, de esta forma, asegura el objetivo (a)?

Como se verá en este tema, mantener, en memoria principal, los bloques actualizados por las transacciones confirmadas (SGBD), durante un periodo de tiempo posterior a la confirmación de la transacción, puede ser útil: reduce el número de accesos a disco, y favorece el acceso al mismo bloque de otras transacciones (entorno concurrente). Si esta estrategia de transferencia de bloques es útil, habrá que buscar una solución para asegurar el objetivo (a) cuando la transacción ha sido confirmada por el SGBD y algunas de sus actualizaciones permanecen (sin grabar) en el buffer de memoria principal.

Es importante observar que una transacción confirmada (SGBD), cuyas actualizaciones permanecen (sin grabar) en memoria principal, corre el riesgo de perderse, ya que la memoria principal es volátil. **IMPORTANTE:** en las imágenes de esta transparencia, y en muchas siguientes, hay que tener en cuenta:

1. Aunque parece que la fila modificada ocupa todo el bloque (y todo el buffer cuando está en MP), realmente el bloque/buffer es mucho más grande y puede contener muchas otras filas.
2. La modificación realizada por la operación cambia el contenido del buffer (que luego bajará al bloque) que contenía la fila con sus valores originales.

1 Recuperación de transacciones: introducción.



La transacción T ha sido anulada por el usuario y sus cambios ya han sido grabados en la BD **¿se cumple (b)?**

Debido a la política de transferencia de bloques seguida por el SGBD, los bloques actualizados por una transacción **pueden** haber sido transferidos a disco antes de que la transacción finalice.

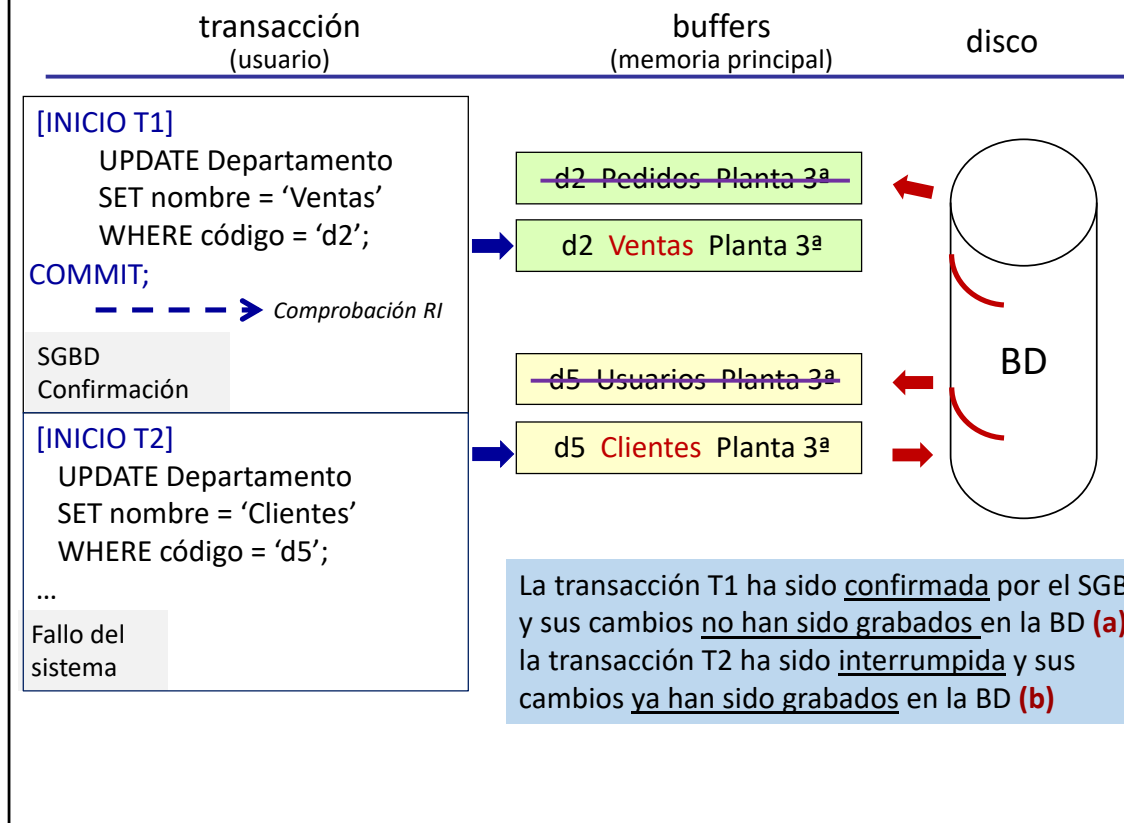
8

En el ejemplo, se muestra una transacción que finaliza con anulación del usuario (ROLLBACK). Según la idea de ejecución correcta de transacciones, esta transacción se debe ejecutar cumpliendo el objetivo (b). Como se puede observar, la actualización de la transacción (UPDATE) se ha ejecutado correctamente, y el bloque actualizado ha sido transferido a disco antes de finalizar la transacción. El objetivo (b) no se ha cumplido. La transacción no se ha ejecutado correctamente.

¿Por qué el SGBD transfiere el bloque actualizado a disco, antes de finalizar la transacción del usuario?

Como se verá en este tema, transferir a disco bloques actualizados por las transacciones, antes de que éstas finalicen, puede ser útil: libera espacio en memoria principal y permite que bloques requeridos por otras transacciones sean transferidos del disco al buffer (entorno concurrente). Si esta estrategia de transferencia de bloques es útil, habrá que buscar una solución para asegurar el objetivo (b) cuando la transacción ha sido anulada y algunas de sus actualizaciones ya han sido grabadas en la base de datos en disco.

1 Recuperación de transacciones: introducción.



En el ejemplo, se muestran dos transacciones:

- la transacción T1 confirmada (SGBD) que no se ha ejecutado correctamente (objetivo (a)).
- la transacción T2 interrumpida, durante su ejecución, que no se ha ejecutado correctamente (objetivo (b)).

En el ejemplo, se observa que el SGBD **mantiene** en memoria principal (sin grabar), bloques actualizados por transacciones confirmadas (SGBD) **después** de su confirmación, y **transfiere** a disco bloques actualizados por transacciones antes de su finalización.

1 Recuperación de transacciones: introducción.

- ✓ Una posible solución a esta situación que garantizaría los objetivos (a) y (b) del procesamiento de transacciones, podría consistir en: “transferir los bloques actualizados por una transacción a disco sólo en el momento en que la transacción es confirmada por el SGBD”.
- ✓ Este comportamiento significa una **política de transferencia de bloques** a disco muy **rígida**: el SGBD debe economizar las operaciones de acceso a disco que son las más costosas y debe tener libertad para liberar espacio (bloques) de memoria principal cuando lo necesite.
- ✓ En los SGBD se contemplan distintas políticas de transferencia de bloques de memoria principal a disco: **estrategias de actualización de la BD** (punto 2).

Nota: En los ejemplos anteriores la política de transferencia de bloques utilizada es muy flexible: un bloque puede ser transferido a disco en cualquier instante (**antes** de finalizar o **después** de confirmar la transacción).

10

Los tres ejemplos anteriores muestran situaciones en las que las transacciones no se ejecutan correctamente, debido a la política de transferencia de bloques seguida por el SGBD.

La solución más fácil sería: “**transferir los bloques actualizados por una transacción a disco sólo en el momento en que la transacción es confirmada por el SGBD**”, es decir, no permitir que un bloque actualizado por una transacción sea transferido a disco antes de su finalización, y obligar a que los bloques actualizados por una transacción confirmada (SGBD) sean transferidos a disco en el momento de la confirmación.

Sin embargo, esta solución es muy **rígida**, como ya se ha explicado anteriormente. Por ello, para flexibilizar la política de transferencia de bloques de memoria principal a disco y asegurar la correcta ejecución de transacciones, habrá que buscar otras **soluciones**.

1 Recuperación de transacciones: introducción.

Política flexible de transferencia de bloques:

- ✓ Cuando una transacción es confirmada por el SGBD, algunas de sus actualizaciones **pueden no haber sido grabadas** aún en la BD.
- ✓ Cuando una transacción es anulada (usuario o SGBD) algunas de sus actualizaciones **pueden haber sido ya grabadas** en la BD.
- ✓ Cuando una transacción es interrumpida por un fallo del sistema algunas de sus actualizaciones **pueden haber sido ya grabadas** en la BD (además puede haber transacciones confirmadas previamente cuyas actualizaciones **pueden no haber sido grabadas** aún en la BD).

Para procesar correctamente transacciones, el SGBD debe incorporar **técnicas de recuperación** que aseguren los objetivos **(a)** y **(b)**.

11

En la transparencia, se resumen las implicaciones que una política de transferencia de bloques flexible puede tener en la ejecución de transacciones.

Estas implicaciones justifican la existencia, en los SGBD, de técnicas de recuperación de las transacciones que no se han ejecutado correctamente.

1 Recuperación de transacciones: introducción.

Objetivo de las técnicas de recuperación:

Asegurar el correcto procesamiento de las transacciones.

Herramientas para la recuperación:

- ✓ Diario
- ✓ Copias de la BD

12

En este punto de introducción del tema, se va a mostrar, de forma simplificada, la **solución** que se ha anunciado en los ejemplos anteriores. Una solución que ofrezca flexibilidad en la transferencia de bloques a disco y que asegure los objetivos (a) y (b) del correcto procesamiento de transacciones.

Se habla de **recuperación de transacciones** porque estas técnicas **recuperan las transacciones que no se han ejecutado correctamente**: transacciones confirmadas (SGBD) cuyas actualizaciones no se han grabado todavía en disco, y transacciones anuladas o interrumpidas cuyas actualizaciones ya se han grabado en disco.

1 Recuperación de transacciones: introducción.

DIARIO



Fichero en el que se registran todas las operaciones de las transacciones ejecutadas en el SGBD.

- ✓ El diario se almacena en disco.
- ✓ Para la actualización del diario se sigue la estrategia de transferencia de bloques entre los buffers de memoria principal y disco: existen buffers específicos destinados a contener bloques del diario.
- ✓ Para prevenir pérdidas del diario por fallos en el disco, periódicamente se hacen copias de seguridad del diario.

13

La herramienta principal para asegurar la correcta ejecución de transacciones, en un SGBD con una política de transferencia de bloques flexible, es el **diario**.

En el diario, se guarda información sobre las operaciones de todas las transacciones que se ejecutan en el SGBD.

1 Recuperación de transacciones: introducción.

Diario: entradas (registros) del diario

- ✓ [inicio, T]
- ✓ [escribir, T, X, valor_antes, valor_después]
- ✓ [leer, T, X]
- ✓ [confirmar, T]: anotación que indica que la transacción T ha sido confirmada por el SGBD al haber finalizado el usuario la transacción y tras haber comprobado todas las restricciones de integridad.
- ✓ [anular, T]: anotación que indica que la transacción T ha sido anulada por el SGBD al haber anulado el usuario la transacción o porque el propio sistema la anula.

14

Los registros (entradas) de un diario pueden ser de cinco tipos distintos, correspondientes a los cinco tipos de operaciones que pueden aparecer en una transacción: **inicio**, **leer**, **escribir**, **confirmar**, **anular**.

En las entradas, se guarda el identificador de la transacción a la que pertenece la operación, el identificador de la operación, y toda la información sobre ella:

- **[leer, T, X]**: anotación que indica que la transacción T ha leído el elemento de datos X
- **[escribir, T, X, valor_antes, valor_después]** : anotación que indica que la transacción T ha escrito en elemento de datos X, su valor antes de la actualización (valor_antes) y su valor después de la actualización (valor_después)
- **[confirmar, T]**: anotación que indica que la transacción T ha sido confirmada por el SGBD al haber finalizado el usuario la transacción y tras haber comprobado todas las restricciones de integridad.
- **[anular, T]**: anotación que indica que la transacción T ha sido anulada por el SGBD al haber anulado el usuario la transacción o porque el propio sistema la anula (por violación de una restricción de integridad p.e.).

En las transparencias siguientes, se va a ilustrar cómo la información guardada en el diario, relativa a las operaciones de lectura y escritura de las transacciones, va a permitir mantener una política de transferencia de bloques flexible y asegurar la correcta ejecución de las transacciones: cumplir con los objetivos (a) y (b).

Hay que recordar que, en esta política de transferencia de bloques flexible, un bloque puede ser transferido a disco en cualquier instante: **antes** de que la transacción finalice o **después** de que la transacción sea confirmada (SGBD). Esta política puede conducir, como se ha mostrado en los ejemplos anteriores, a situaciones en las que las transacciones no se ejecutan correctamente.

1 Recuperación de transacciones: introducción.

Estrategia de recuperación de **transacciones confirmadas** frente a fallos del sistema con **pérdida de memoria principal***

Las actualizaciones de una transacción confirmada (SGBD) pueden no haber sido grabadas en la base de datos



Rehacer** transacciones a partir del diario

* Esta estrategia de recuperación asume una política de transferencia de bloques flexible que será definida más adelante (punto 2): los bloques pueden ser transferidos a disco en cualquier instante (antes de finalizar la transacción y después de confirmar la transacción)

** **Rehacer** no significa volver a ejecutar sino asegurar que los cambios realizados por la transacción no se pierdan.

15

Supóngase una situación de fallo del sistema con **pérdida de memoria principal** (caída del SGBD, caída del SO, interrupción del suministro eléctrico, ...).

En esta situación de fallo, si se usa una política de transferencia de bloques flexible, puede suceder que bloques actualizados, por transacciones **confirmadas (SGBD)** antes del fallo, no hayan sido todavía transferidos a disco, y que con el fallo se pierdan. Puede haber transacciones (ya confirmadas) que no se han ejecutado correctamente.

El mantenimiento de un diario en el SGBD, donde se registran las operaciones de todas las transacciones ejecutadas en el sistema, va a permitir **recuperar** esas **transacciones**, después del fallo, y **rehacerlas**, es decir conseguir que los cambios realizados por esas transacciones no se pierdan.

1 Recuperación de transacciones: introducción.

Estrategia de recuperación de transacciones confirmadas frente a fallos del sistema con pérdida de memoria principal

Herramienta de recuperación: **diario**

[escribir, T, X, valor_antes, valor_después]

Técnica de recuperación:

Transacciones T **confirmadas** en el diario:

([inicio, T] [confirmar, T])



se deben **rehacer** las actualizaciones de T (valor_después)

en el orden en que aparecen en el diario

16

La forma de **recuperar** las **transacciones confirmadas (SGBD)** antes del fallo, de las que no se tiene la seguridad de haber sido grabadas en disco, es la siguiente: de todas las transacciones, que aparecen **confirmadas** en el diario antes del fallo, se vuelven a ejecutar (**rehacer**) todas las operaciones **escribir** en el orden que aparecen registradas en el diario, aplicando el **valor_después** (valor actualizado).

1 Recuperación de transacciones: introducción.

Estrategia de recuperación de transacciones falladas (transacciones anuladas o interrumpidas)

Las actualizaciones de una transacción anulada o interrumpida pueden haber sido grabadas en la base de datos



Deshacer transacciones
a partir del diario

17

Como ya se ha dicho, las transacciones anuladas o interrumpidas deben ejecutarse cumpliendo el objetivo (b): no deben dejar ningún efecto en la base de datos.

Con una política de transferencia de bloques flexible, como la que se está asumiendo, es posible que bloques actualizados por las transacciones sean transferidos a disco antes de finalizar la transacción.

En esta situación, bloques actualizados por las transacciones anuladas (por el usuario o el SGBD) o interrumpidas (por un fallo) pueden haber sido ya transferidos a disco. La correcta ejecución de estas transacciones obliga al SGBD a cumplir el objetivo (b), es decir, no dejar ningún efecto en la base de datos. Por este motivo, el sistema debe **recuperar** estas transacciones y **deshacer**, después de la anulación o del fallo, sus actualizaciones que puedan haber sido ya grabadas en disco. Esto es posible con la información registrada en el diario.

1 Recuperación de transacciones: introducción.

Estrategia de recuperación de transacciones falladas (transacciones anuladas o interrumpidas)

Herramienta de recuperación: **diario**

[escribir, T, X, valor_antes, valor_después]

Técnica de recuperación:

Transacciones T **anuladas** en el diario:

([inicio, T] [anular, T])



se deben **deshacer** las actualizaciones de T
(valor_antes)

Transacciones T **interrumpidas** en el diario:

([inicio, T])



se deben **deshacer** las actualizaciones de T
(valor_antes)

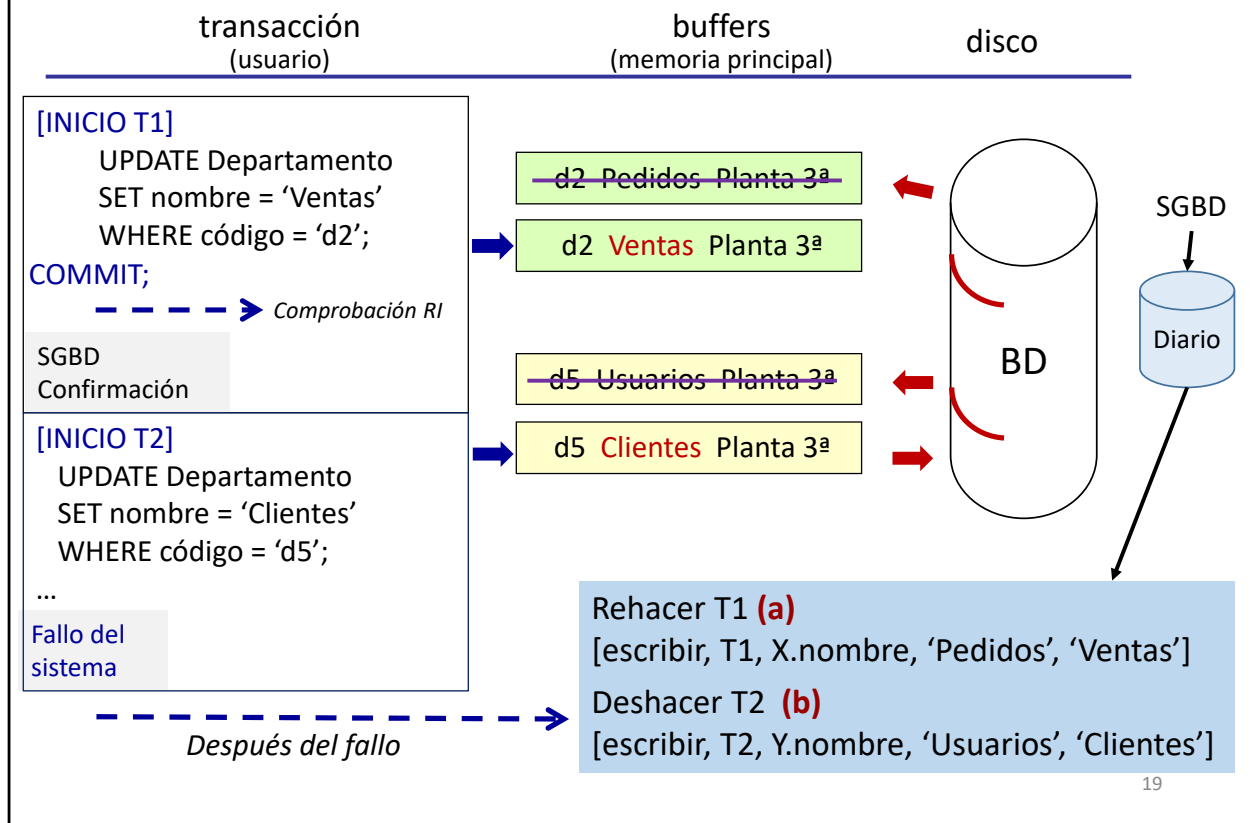
en orden inverso al que aparecen en
el diario

18

La forma de **recuperar** las **transacciones anuladas o interrumpidas**, cuyas actualizaciones pueden haber sido ya grabadas en disco, es la siguiente:

- Transacción **anulada**: en el momento de la anulación del usuario (o del SGBD), se deben **deshacer** las actualizaciones de la transacción, para ello se vuelven a ejecutar, en orden inverso al que aparecen registradas en el diario, las operaciones **escribir** de la transacción, aplicando el **valor_antes** (valor anterior a la actualización).
- Transacción **interrumpida**: después del fallo, se deben **deshacer** las actualizaciones de la transacción, para ello se vuelven a ejecutar, en orden inverso al que aparecen registradas en el diario, las operaciones **escribir** de la transacción, aplicando el **valor_antes** (valor anterior a la actualización).

1 Recuperación de transacciones: introducción.



Si suponemos que X es la fila que ha modificado la transacción T1 e Y la modificada por la transacción T2, después del fallo, se debe **rehacer** T1 y se debe **deshacer** T2, a partir de la información registrada en el diario del SGBD.

1 Recuperación de transacciones: introducción.

Estrategia de recuperación de transacciones confirmadas frente a fallos del sistema de almacenamiento secundario

Herramienta de recuperación:

- Diario
- Copias de seguridad

Técnica de recuperación:



Cargar la base de datos a partir de la última copia de seguridad y a continuación rehacer todas las transacciones que aparecen confirmadas en el diario desde la fecha de la copia.

20

En el caso de que se produzca un fallo con pérdida de memoria secundaria, es decir, se pierda o se deteriore la base de datos, la técnica de recuperación es distinta.

El objetivo será recuperar la base de datos al estado más próximo en el tiempo al momento del fallo.

La forma de hacer la recuperación es la siguiente: se carga la base de datos a partir de su última copia de seguridad, y sobre ese estado se **rehacen** todas las transacciones que aparecen en el fichero de diario **confirmadas** después de la fecha de la copia.

Para este tipo de recuperación, será muy importante tener prevista una política de copias de seguridad de la base de datos y del fichero de diario.

1 Recuperación de transacciones: introducción.

Gestión del diario

Principios para la gestión de un diario:

- ✓ Forzar la escritura del diario.
- ✓ Escritura anticipada del diario.
- ✓ Uso de puntos de control (*checkpoint*) en el diario.

21

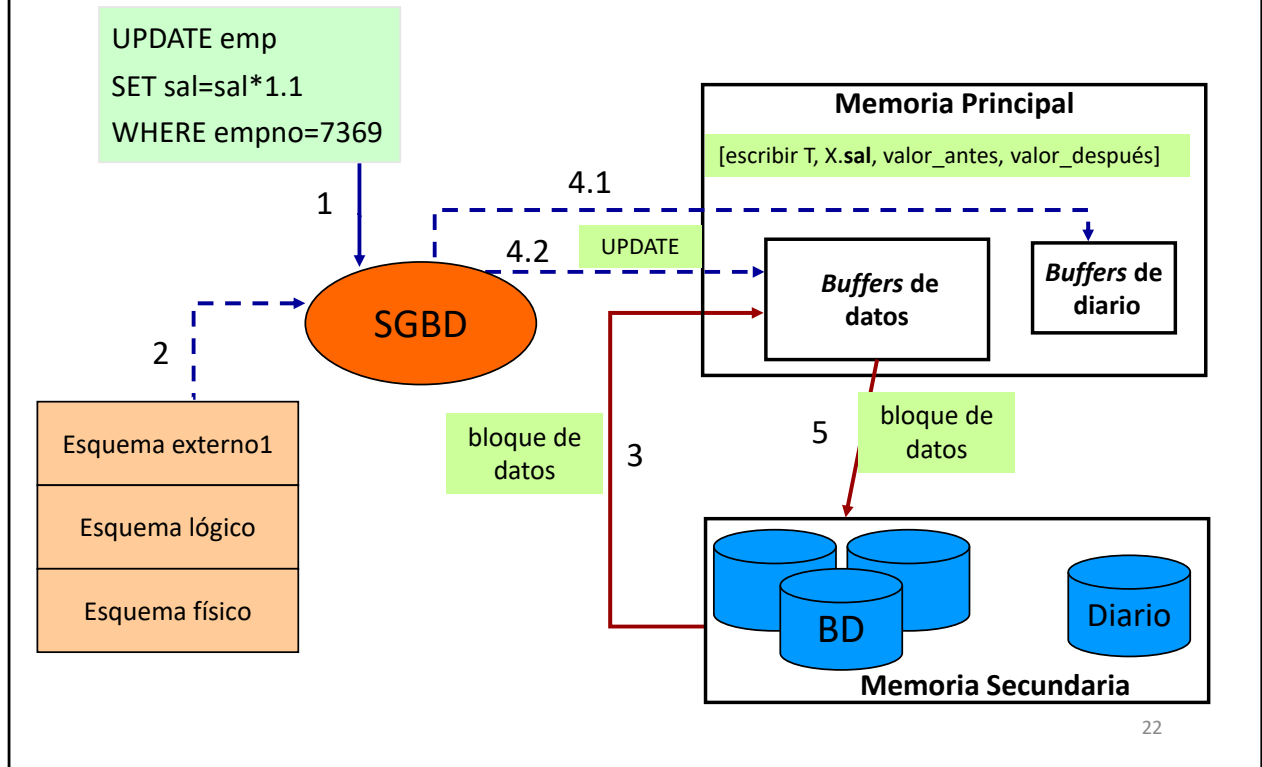
Se ha visto como, en un SGBD con una política de transferencia de bloques flexible, el uso del diario permite ejecutar las transacciones correctamente.

Es importante tener en cuenta que las anotaciones en el diario durante la ejecución de las transacciones se realizan en bloques de memoria principal específicos del diario, y que estos bloques del fichero de diario deben ser transferidos, posteriormente a disco. Existen buffers específicos para contener bloques del fichero de diario.

El diario se debe gestionar de forma que, en caso de tener que recuperar transacciones, toda la información necesaria para la recuperación esté disponible en el **fichero de diario en disco**, y no sólo en un buffer de memoria principal (volátil). Por ello, la gestión de un diario se realiza siguiendo ciertos **principios de seguridad** que se presentan a continuación.

1 Recuperación de transacciones: introducción.

Procesamiento de una operación de actualización (uso del diario)



En la transparencia, se ilustran los pasos en la ejecución de una operación de actualización: ejecución del UPDATE (paso 4.2), actualización de la BD (paso 5) y uso del diario (**paso 4.1**).

Respecto a diagramas anteriores, se ha introducido el paso correspondiente al uso del diario (suponemos que X es la fila afectada por la modificación).

1 Recuperación de transacciones: introducción.

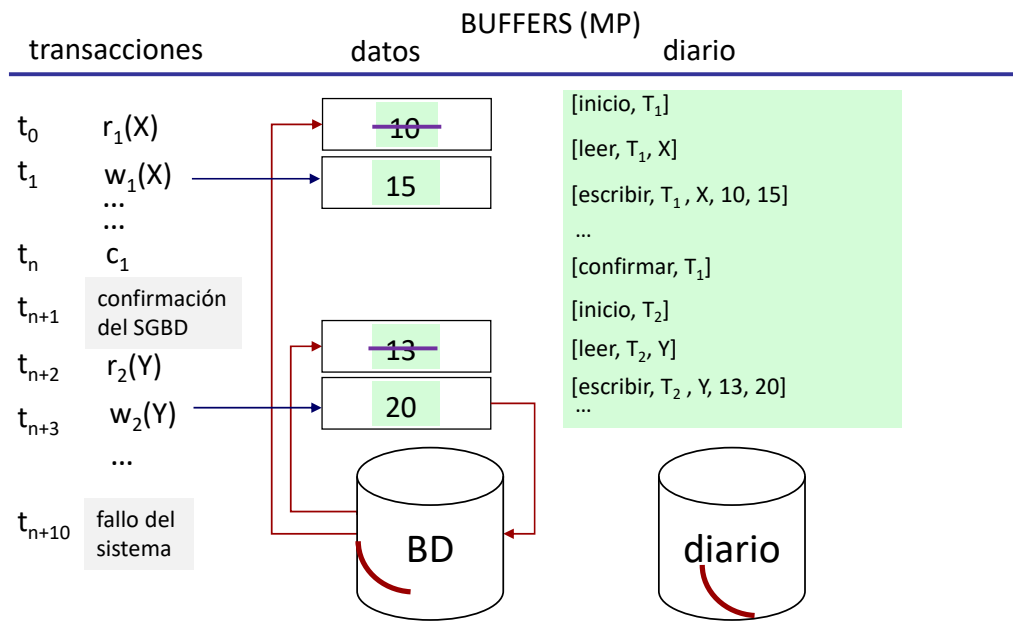
Notación simplificada para representar las operaciones de una transacción T_i :

- ✓ leer(x): $r_i(x)$
- ✓ escribir(x): $w_i(x)$
- ✓ fin (confirmación de usuario): c_i
- ✓ anulación (de usuario): a_i

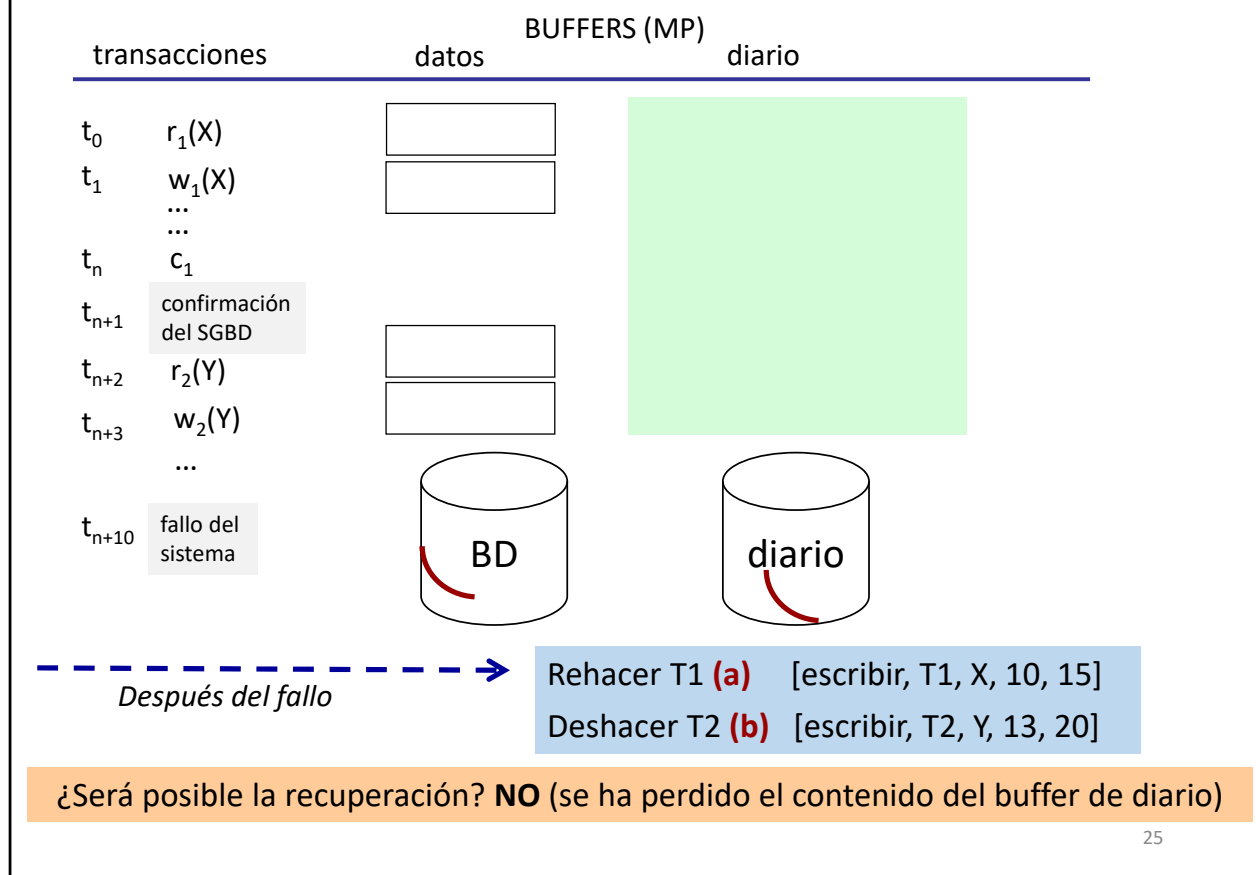
23

A partir de ahora y para simplificar la escritura de transacciones, se propone una notación simplificada para las operaciones de la transacción, como se muestra en la transparencia. En algunos ejemplos, se seguirá usando sintaxis SQL.

1 Recuperación de transacciones: introducción.



1 Recuperación de transacciones: introducción.



25

En el ejemplo, se muestra la ejecución de dos transacciones, en un SGBD con una política de transferencia de bloques flexible, indicando las entradas que se van grabando en el buffer de diario durante la ejecución de la transacción. Como ya se ha explicado, en este ejemplo, después del fallo del sistema, se debería **rehacer** la transacción T1 y se debería **deshacer** la transacción T2, a partir de la información registrada en el diario.

Pero, ¿dónde está, después del fallo, la información (las entradas) registrada en el diario sobre estas dos transacciones?

Obviamente, tanto el buffer de datos como el buffer de diario se han perdido como consecuencia del fallo, por lo que la información para la recuperación de las transacciones ya no existe.

El ejemplo justifica la necesidad de una **política de gestión del diario segura** que garantice que la información para la recuperación de las transacciones se encuentra en el fichero de diario en **disco**, en el momento en que se necesita, es decir, que ha sido transferida del buffer de diario al fichero de diario en disco en el momento adecuado.

Se trata de evitar que información necesaria, para la recuperación de transacciones, se pierda en el buffer de memoria principal (volátil).

En el instante t_{n+1} , “confirmación del SGBD”, no se ejecuta realmente una operación de la transacción (al igual que la que aparece en t_{n+10}); esta entrada indica que el SGBD informa al usuario de que la confirmación es definitiva.

1 Recuperación de transacciones: introducción.

Diario: principios de gestión

Forzar la escritura del diario: todas las entradas de diario correspondientes a una transacción deben haber sido grabadas en el fichero de diario **en disco antes** de que la transacción sea confirmada (SGBD).



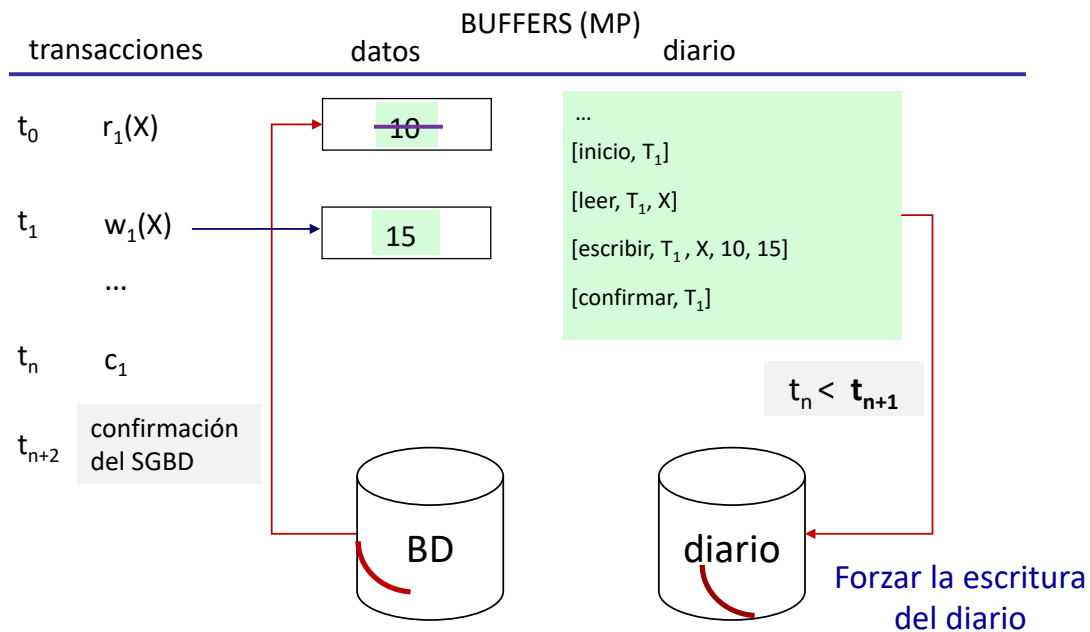
Asegura la recuperación de las transacciones **confirmadas**.

26

El principio de **forzar la escritura del diario** obliga a que, **antes** de que una transacción sea confirmada definitivamente por el SGBD, el contenido del buffer de diario, que contiene las entradas correspondientes a la transacción, sea grabado en el fichero de diario en **disco**.

Este principio asegura que, en caso de producirse un fallo después de la confirmación (SGBD), la transacción podrá **rehacerse**, ya que las entradas correspondientes a sus actualizaciones están registradas en el fichero de diario en disco.

1 Recuperación de transacciones: introducción.



Las entradas del diario correspondientes a una transacción confirmada por el usuario (COMMIT) se graban en disco antes de que se confirme definitivamente (SGBD) la transacción, independientemente de que se hayan transferido o no los bloques de datos actualizados.

En el ejemplo, se ilustra el **principio de forzar la escritura del diario**, en un SGBD con una política de transferencia de bloques flexible.

Obsérvese cómo, antes de que el SGBD confirme definitivamente la transacción, el contenido del buffer de diario se graba en el fichero de diario en disco. Si ocurre un fallo, después de la confirmación (SGBD) de la transacción, toda la información necesaria para su recuperación (**rehacer** las actualizaciones) está disponible en el fichero de diario en el disco.

1 Recuperación de transacciones: introducción.

Diario: principios de gestión

Escritura anticipada en el diario: las entradas de diario correspondientes a actualizaciones: ([escribir, T, X, valor_antes, valor_después]) deben haber sido grabadas en el fichero de diario **en disco antes** de que los bloques de datos con dichas actualizaciones sean transferidos a disco.



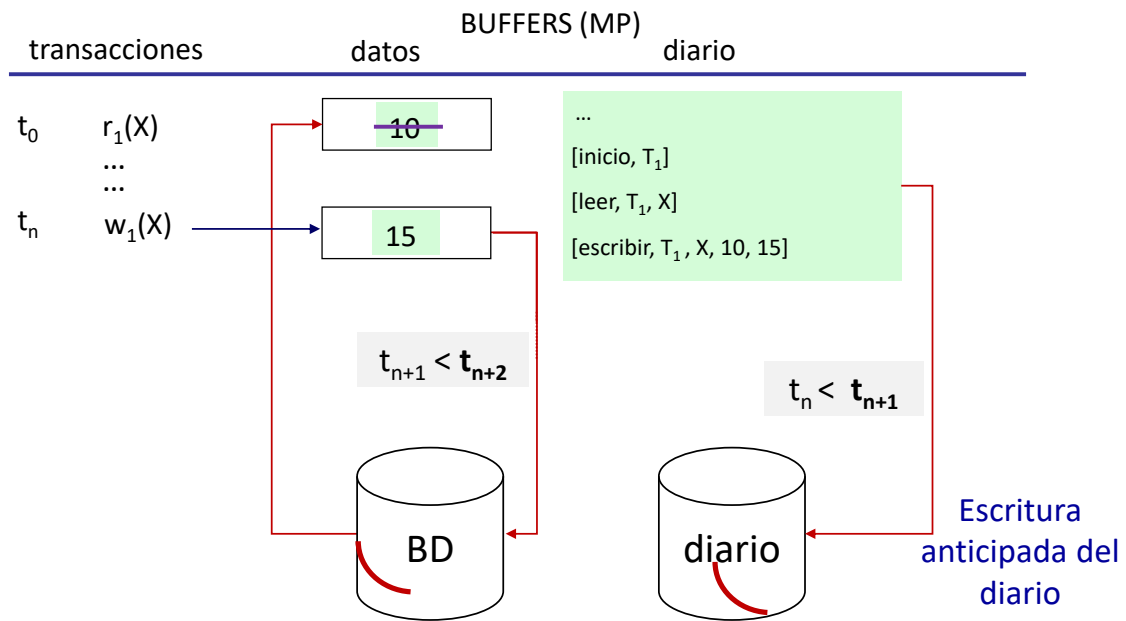
Asegura la recuperación de las transacciones **anuladas e interrumpidas.**

28

El principio de **escritura anticipada en el diario** obliga a que, **antes** de que un bloque actualizado por una transacción sea transferido a disco, el contenido del buffer de diario, que contiene las entradas correspondientes a actualizaciones del bloque, sea grabado en el fichero de diario en disco.

Este principio asegura que, en caso de anulación o interrupción de la transacción (fallo), sus actualizaciones podrán **deshacerse**, ya que las entradas correspondientes a esas actualizaciones están registradas en el fichero de diario en disco.

1 Recuperación de transacciones: introducción.



Las entradas del diario se graban en disco antes de que se transfieran a disco bloques de datos actualizados por una transacción.

29

En el ejemplo, se ilustra el **principio de escritura anticipada en el diario**, en un SGBD con una política de transferencia de bloques flexible.

Obsérvese cómo, antes de que el bloque actualizado por la transacción sea transferido a disco, el contenido del buffer de diario se graba en el fichero de diario en disco. Si finalmente la transacción es anulada, o interrumpida por un fallo, toda la información necesaria para su recuperación (**deshacer** las actualizaciones) está disponible en el fichero de diario en el disco.

1 Recuperación de transacciones: introducción.

Diario: principios de gestión

Para que sea posible la recuperación de transacciones, en el fichero de diario en disco deben estar grabadas todas las entradas necesarias para la recuperación.

Protocolo de escritura en el diario

(WAL: *Write-Ahead Logging*):

1. Antes de que se confirme definitivamente una transacción (SGBD), las entradas de diario correspondientes a actualizaciones de la transacción deben haber sido grabadas en el fichero de diario en disco. **(Forzar la escritura del diario).**
2. Antes de que un bloque de datos actualizado por una transacción sea transferido a disco, las entradas de diario correspondientes a actualizaciones del bloque deben haber sido grabadas en el fichero de diario en disco. **(Escritura anticipada del diario).**

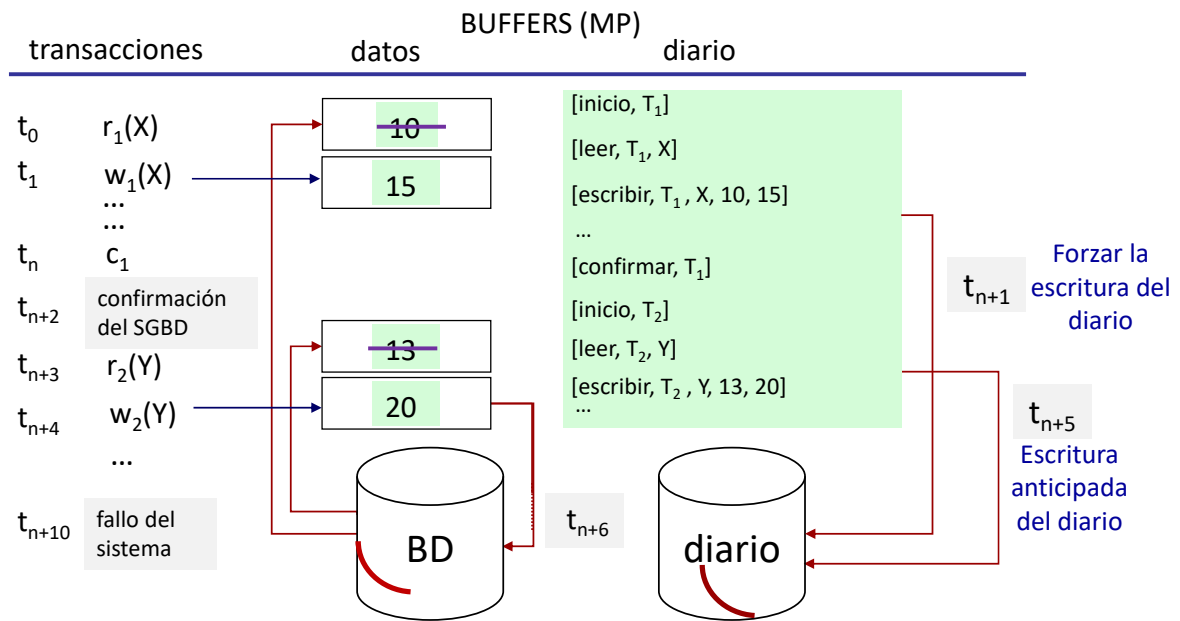
30

El protocolo WAL reúne los dos principios de gestión de un diario.

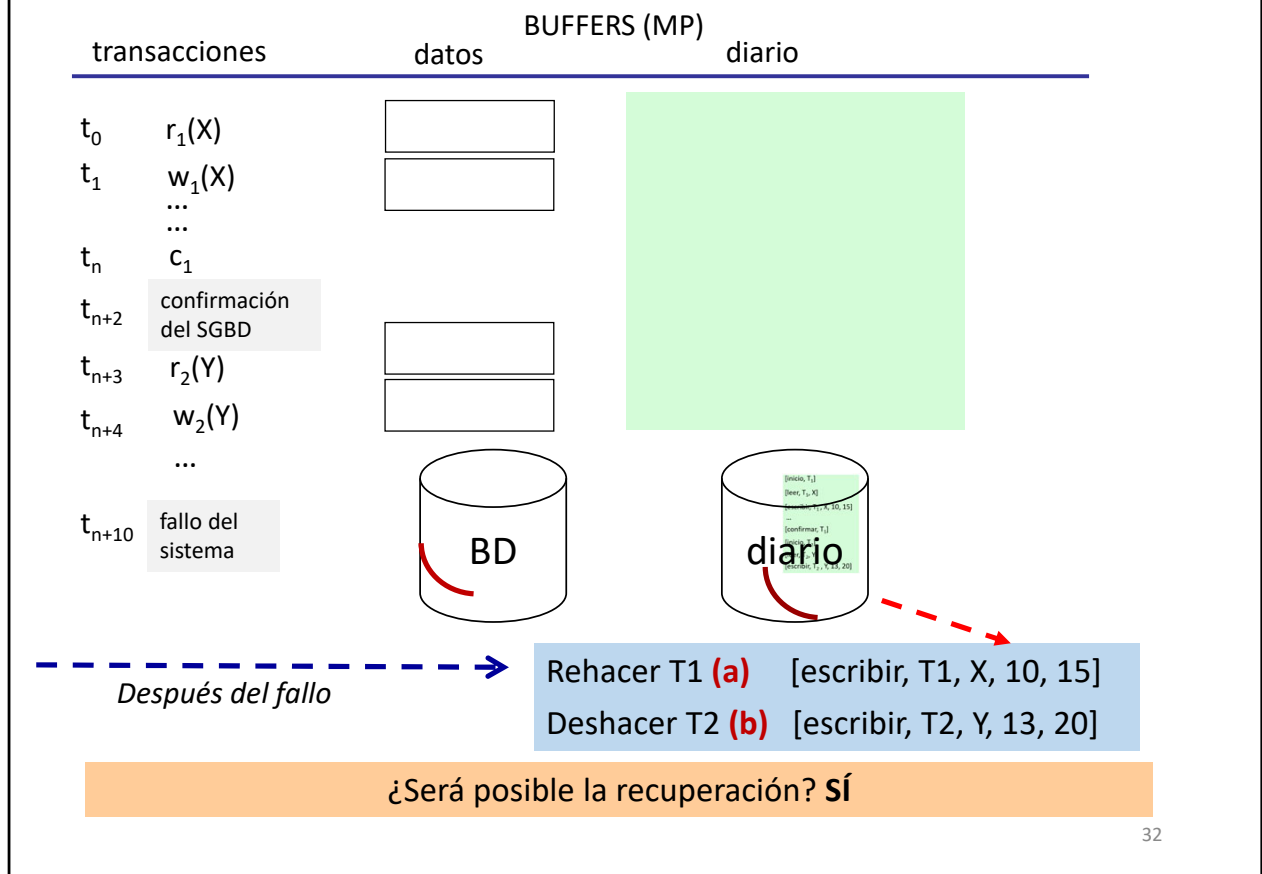
En los ejemplos anteriores, se ha visto cómo, en un SGBD con una política de transferencia de bloques flexible, estos dos principios aseguran la recuperación de transacciones, en caso de anulación o de fallo (con pérdida de memoria principal).

En sistemas de gestión de bases de datos, con otras políticas de transferencia de bloques a disco, el protocolo de gestión del diario puede simplificarse (punto 2).

1 Recuperación de transacciones: introducción.



1 Recuperación de transacciones: introducción.



32

Esta transparencia y la anterior muestran el mismo ejemplo que se ha presentado anteriormente, pero ahora usando una política de gestión del diario segura.

En el ejemplo, se puede observar que si se cumplen los dos principios de gestión del diario, las dos transacciones se pueden recuperar después del fallo (con pérdida de memoria principal) ya que toda la información necesaria para la recuperación está disponible en el fichero de diario en disco.

1 Recuperación de transacciones: introducción.

Diario: principios de gestión

PUNTO DE CONTROL



Marca que el SGBD registra en el diario indicando que en ese momento todas las actualizaciones de transacciones confirmadas, es decir con una entrada en el diario [confirmar, T], han sido grabadas en disco.

Los puntos de control en el diario:

- ✓ Simplifican el proceso de recuperación: las transacciones con una entrada de confirmación anterior al último punto de control no se deben rehacer durante la recuperación.
- ✓ El SGBD decide la frecuencia con la que se registran los puntos de control.

33

Como se ha visto, después de un fallo del sistema (con pérdida de memoria principal), el SGBD debe recuperar las transacciones confirmadas antes del fallo, ya que es posible que algunas de sus actualizaciones no hayan sido grabadas en la base de datos.

Para reducir el número de transacciones que se deben **rehacer** durante la recuperación, el SGBD registra, periódicamente, en el diario, **puntos de control**. Cuando se registra un punto de control, el SGBD transfiere a disco los bloques actualizados por todas las transacciones que han sido confirmadas desde el último punto de control.

1 Recuperación de transacciones: introducción.

Anotación de un punto de control en el diario: ¿qué significa?

- ✓ Suspenden temporalmente las transacciones activas.
- ✓ Transferir a disco todos los bloques actualizados por transacciones confirmadas después del último punto de control (implica escritura anticipada en el diario)
- ✓ Anotar el punto de control en el buffer de diario y forzar la escritura del diario.
- ✓ Reactivar las transacciones suspendidas.

34

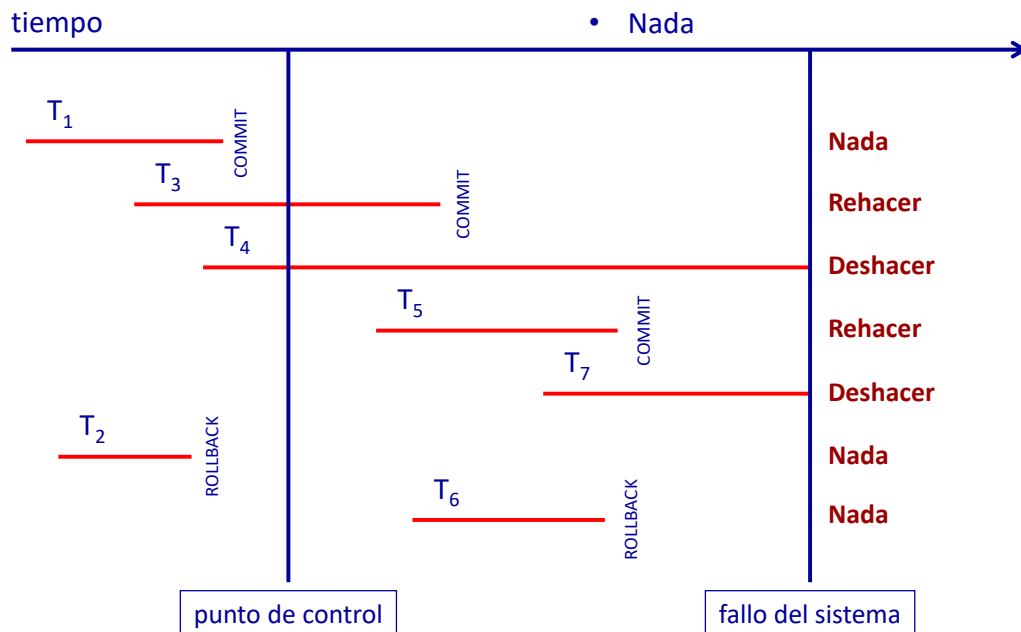
En la transparencia, se muestran las tareas que el SGBD realiza cuando registra un punto de control en el diario.

1 Recuperación de transacciones: introducción.

¿Qué hacer para recuperarse después del fallo?

- Rehacer
- Deshacer
- Nada

Ejemplo:



El ejemplo muestra la ejecución de varias transacciones en un entorno concurrente (Tema 4)

35

En el ejemplo, se ilustra el uso de los puntos de control durante el proceso de recuperación.

- La transacción T1 había sido confirmada antes del último punto de control, por lo tanto se tiene la seguridad de que sus cambios han sido grabados permanentemente sobre la base de datos.
- La transacción T2 fue anulada antes del fallo del sistema por lo que sus cambios, de haber llegado a disco, fueron anulados en ese momento.
- La transacción T3 se inició antes del último punto de control y se confirmó después de él, antes de producirse el fallo del sistema, por lo tanto no se puede asegurar que todos sus cambios hayan sido grabados en disco. Hay que rehacerla.
- La transacción T4 se inició también antes del último punto de control y no había finalizado cuando se produjo el fallo del sistema, por lo tanto deberá ser anulada automáticamente cuando se reinicie el sistema y deberán deshacerse todos sus cambios.
- La transacción T5 se inició después del último punto de control y se confirmó antes de producirse el fallo, por lo tanto no se puede asegurar que sus cambios hayan sido grabados sobre la base de datos. Hay que rehacerla.
- La transacción T6 fue anulada antes del fallo del sistema por lo que sus cambios, de haber llegado a disco, fueron anulados en ese momento.
- Por último la transacción T7 se inició después del último punto de control y no había finalizado al producirse el fallo, por lo tanto deberá ser anulada automáticamente cuando se reinicie el sistema y deberán deshacerse todos sus cambios.

Recuperación en bases de datos.

- 1 Recuperación de transacciones: introducción.
- 2 Estrategias de actualización en BD.
- 3 Estrategias de recuperación en BD.

2 Estrategias de actualización en BD.

Las tareas de recuperación de transacciones están condicionadas por la estrategia de actualización de la BD (en disco) seguida por el SGBD:



Política de transferencia de bloques de datos entre los buffers asignados en memoria principal y el disco.

37

En el punto 1 del tema, se ha visto cómo una política de transferencia de bloques flexible obliga a mantener, en el SGBD, un diario y a aplicar técnicas de recuperación de transacciones, en distintas situaciones: fallos del sistema y anulación de transacciones.

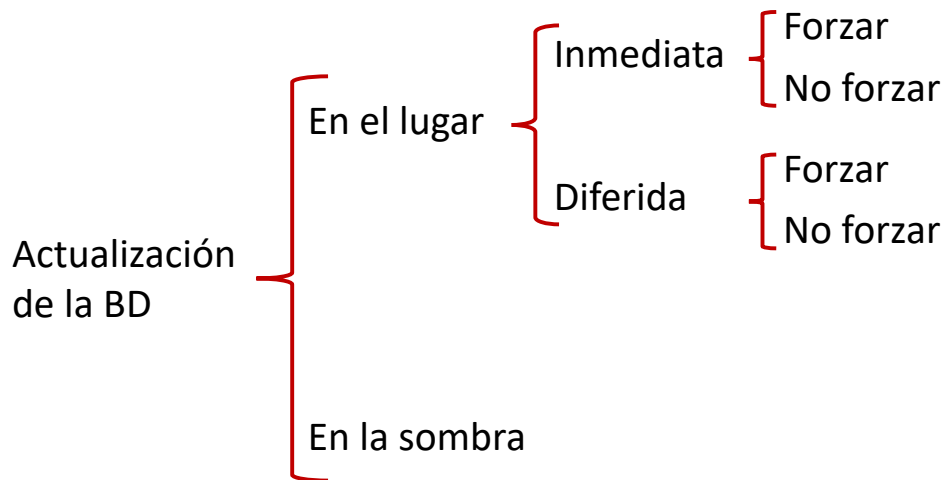
Pero, **¿es esa la única política de transferencia de bloques posible en un SGBD?**

En este punto, se van a estudiar distintas alternativas a la política de transferencia de bloques en un SGBD. Estas políticas van a determinar las técnicas de recuperación de transacciones que se deban utilizar.

A estas políticas se les conoce como **estrategias de actualización** de la base de datos.

2 Estrategias de actualización en BD.

Clasificación de las estrategias de actualización en BD:



38

Las estrategias de actualización de la BD, es decir, las políticas de transferencia de bloques a disco, se pueden clasificar según el presente cuadro.

2 Estrategias de actualización en BD.

Clasificación de las estrategias de actualización en BD:

Actualización en el lugar: cuando un bloque de datos se trasfiere del buffer al disco se graba en la ubicación original del bloque, sobrescribiendo el valor antiguo de cualquier elemento de datos que haya sido actualizado en el bloque.

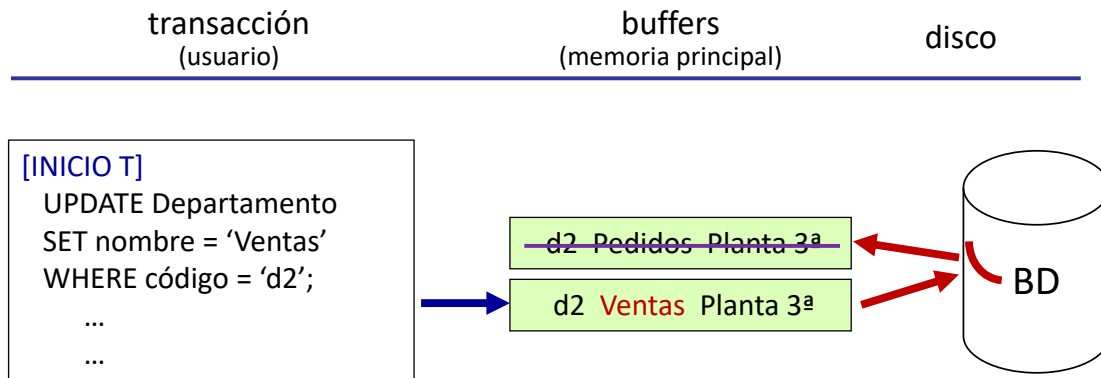
Actualización en la sombra: cuando un bloque de datos se trasfiere del buffer al disco se graba en una nueva ubicación, lo que permite mantener (temporalmente) varias versiones de los bloques de datos actualizados.

39

En un primer nivel, las estrategias de actualización de la BD se pueden clasificar en: actualización **en el lugar** y actualización **en la sombra**, según en **qué dirección** en el disco (ubicación) se graban los bloques actualizados por una transacción.

Estas estrategias se ilustran en las transparencias siguientes.

2 Estrategias de actualización en BD.



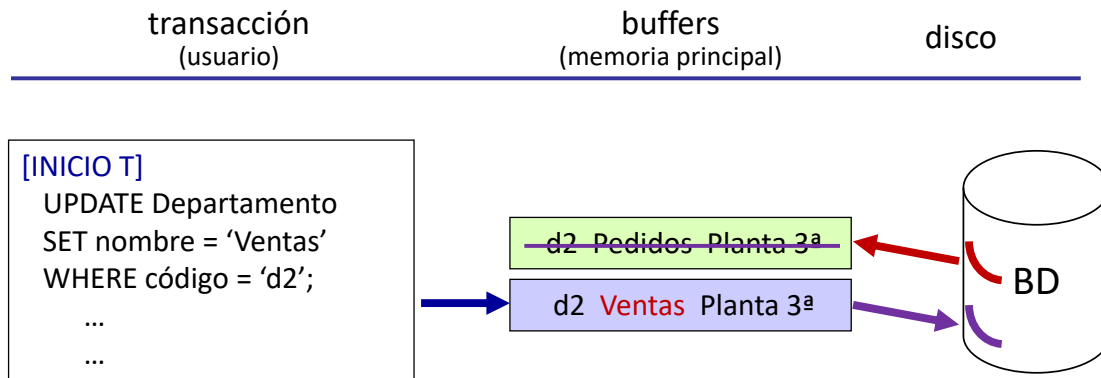
El bloque de datos con las actualizaciones de T es grabado en la misma dirección en el disco.

Actualización en
el lugar

40

En la estrategia de actualización **en el lugar**, cuando un bloque se transfiere, de memoria principal a disco, se escribe en la misma dirección que ocupaba antes de ser leído. Es importante observar que, en esta estrategia, cuando el bloque se transfiere a disco, se **reescribe** su contenido y se pierde el valor anterior de todos sus elementos de datos que ha sido actualizados.

2 Estrategias de actualización en BD.



El bloque de datos con las actualizaciones de T es grabado en una dirección distinta en el disco.

Actualización en
la sombra

41

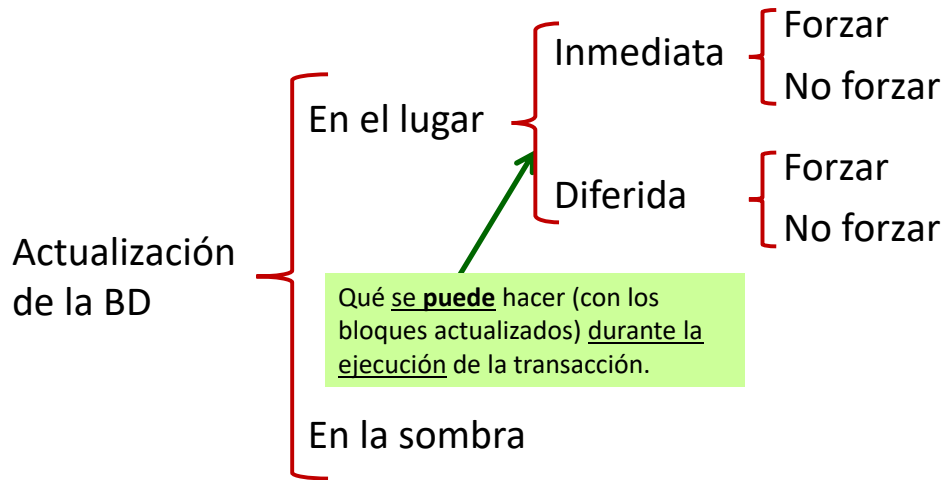
En la estrategia de actualización **en la sombra**, cuando un bloque se transfiere, de memoria principal a disco, se escribe en una dirección distinta a la que ocupaba antes de ser leído. Es importante observar que, en esta estrategia, cuando el bloque se transfiere a disco, se conserva (temporalmente) su valor anterior.

Esta estrategia debe llevar asociada una política por la cual el SGBD libere periódicamente las versiones anteriores de un bloque de datos y se quede con la última versión.

Esta estrategia de actualización de la BD no es muy frecuente en los SGBD relacionales comerciales, por lo que no será considerada en este tema.

2 Estrategias de actualización en BD.

Clasificación de las estrategias de actualización en el lugar:



42

En un segundo nivel, las estrategias de actualización en el lugar se clasifican en **inmediatas** o **diferidas**, según que los bloques actualizados por las transacciones **se puedan** o **no** (se puedan) **transferir** a disco **antes de finalizar la transacción**, es decir, durante la ejecución de la transacción.

2 Estrategias de actualización en BD.

Clasificación de las estrategias de actualización en el lugar:

Actualización inmediata: en esta estrategia, los bloques de datos actualizados por las transacciones (en el buffer) se pueden transferir a disco antes de que finalice la correspondiente transacción.

Actualización diferida: en esta estrategia, los bloques de datos actualizados por las transacciones (en el buffer) no se pueden transferir a disco antes de que finalice la correspondiente transacción.

43

En la estrategia de actualización **inmediata**, los bloques de datos actualizados por las transacciones (en el buffer) se pueden transferir a disco antes de que finalice la correspondiente transacción.

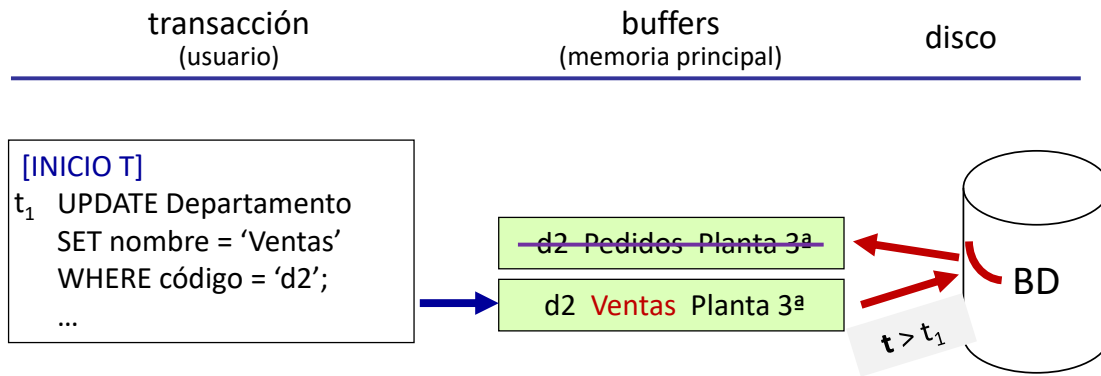
En la estrategia de actualización **diferida**, los bloques de datos actualizados por las transacciones (en el buffer) no se pueden transferir a disco antes de que finalice la correspondiente transacción.

Es importante fijarse en que, en la estrategia **inmediata**, se permite la transferencia de bloques antes de finalizar la transacción, pero esto no significa que haya que hacerlo necesariamente.

En la estrategia **diferida**, se prohíbe la transferencia de bloques antes de finalizar la transacción, es decir, sólo se pueden transferir bloques a disco a partir de la finalización de la transacción. En este caso, como es obvio, sólo si la transacción finaliza con confirmación (SGBD) los bloques actualizados serán transferidos a disco en algún momento posterior.

En las transparencias siguientes, se ilustran estas dos estrategias.

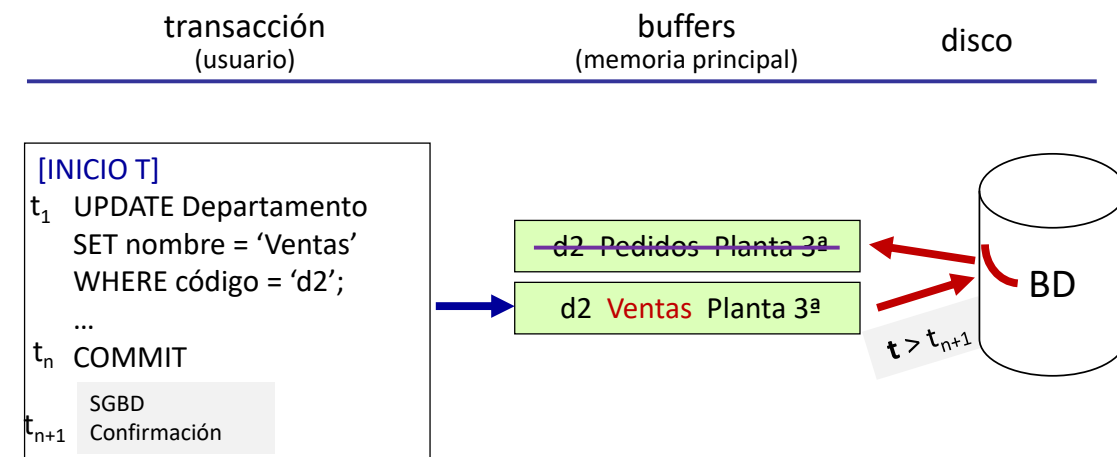
2 Estrategias de actualización en BD.



La transacción T no ha finalizado pero a partir de t_1 el bloque con su actualización puede ser transferido a disco.

Actualización
inmediata

2 Estrategias de actualización en BD.



La transacción T ya ha finalizado (confirmada) y a partir de ese momento los bloques con sus actualizaciones pueden ser transferidos a disco.

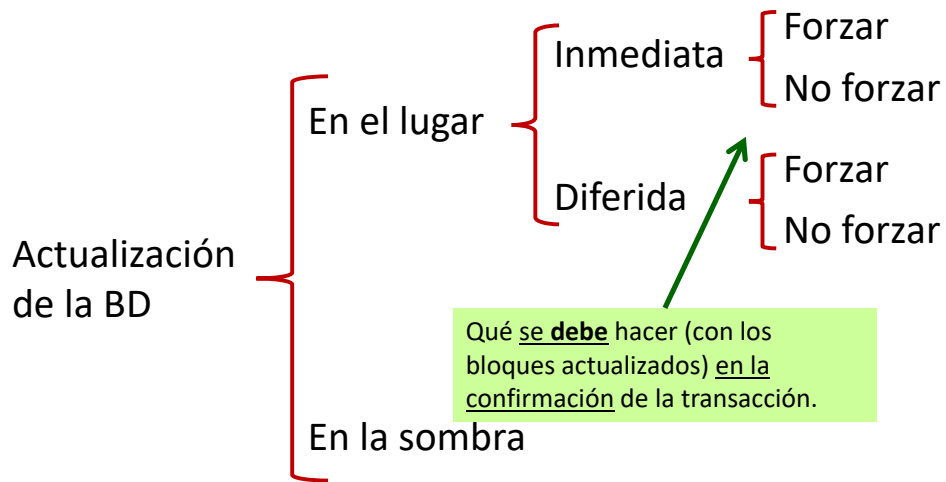
Actualización
diferida

45

Obsérvese en el ejemplo que, una vez se ha confirmado la transacción (SGBD), en el instante de tiempo t_{n+1} , los bloques actualizados se podrán transferir a disco en cualquier instante de tiempo posterior.

2 Estrategias de actualización en BD.

Clasificación de las estrategias de actualización en BD:



46

En un tercer nivel, las políticas de actualización en el lugar, inmediatas o diferidas, se clasifican en estrategias **forzar o no forzar**, según que, en el momento de la **confirmación** de la transacción (SGBD), **se obligue o no** (se obligue) a **transferir** a disco los bloques actualizados por la transacción.

2 Estrategias de actualización en BD.

Clasificación de las estrategias de actualización en el lugar:

Estrategia no forzar: los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, no serán obligatoriamente transferidos en la confirmación de la transacción (**no se realiza ninguna acción en el momento de la confirmación de la transacción**).

Estrategia forzar: los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, serán transferidos antes de que el SGBD confirme definitivamente la transacción (**forzar la escritura en disco de las actualizaciones de la transacción en el momento de la confirmación**).

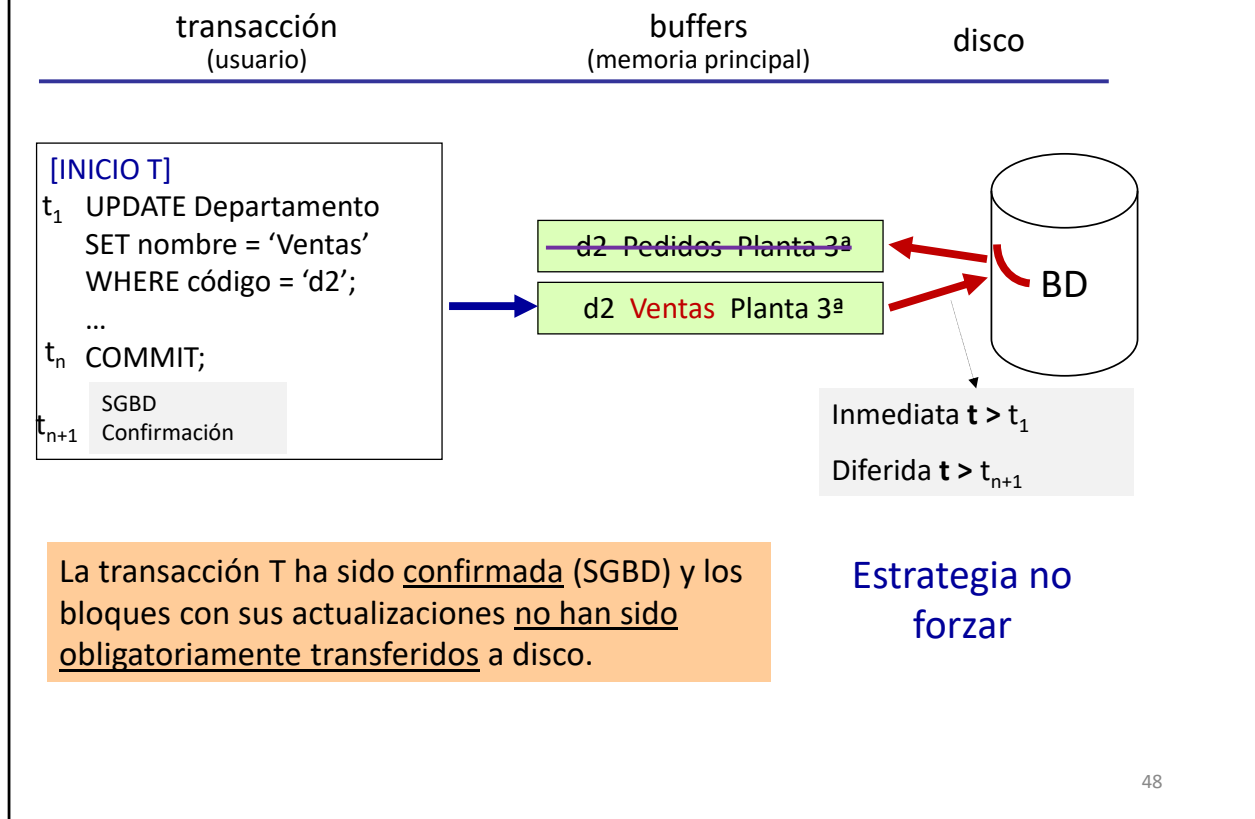
47

En la estrategia **no forzar**, los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, no serán transferidos obligatoriamente en la confirmación de la transacción.

En la estrategia **forzar**, los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, serán transferidos antes de confirmar definitivamente (SGBD) la transacción.

En las transparencias siguientes, se ilustran estas dos estrategias.

2 Estrategias de actualización en BD.

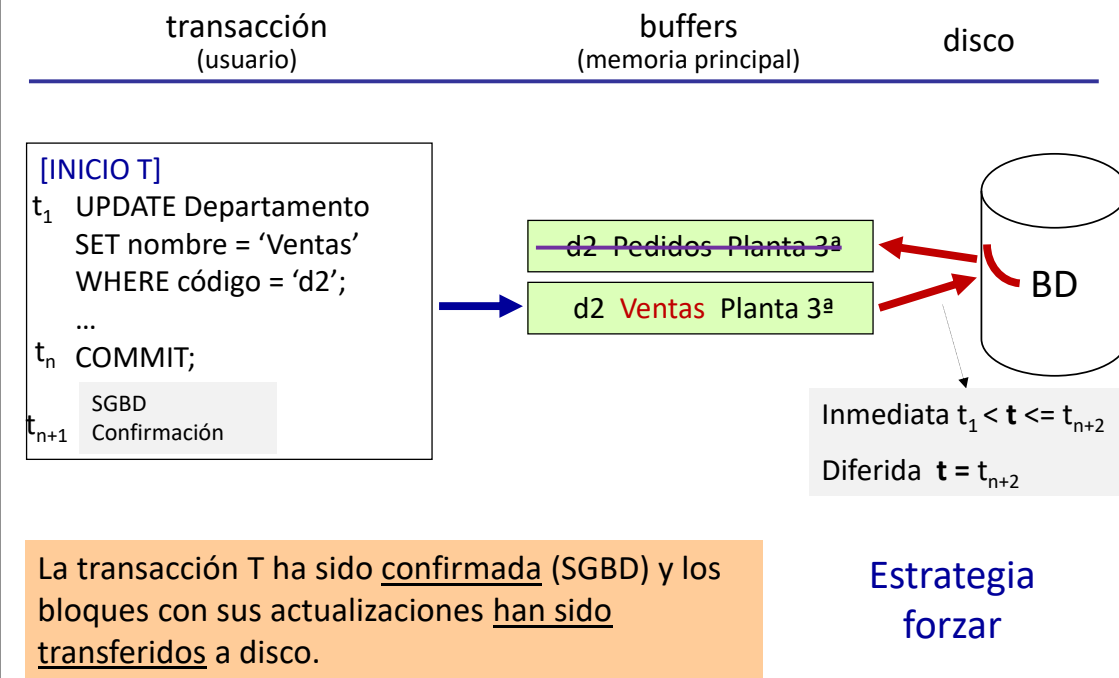


En el ejemplo, la transacción ha sido confirmada por el SGBD y sus actualizaciones no se sabe cuándo llegarán a disco:

- En el caso de actualización inmediata pueden llegar en cualquier momento después de t_1 .
- En el caso de actualización diferida pueden llegar en cualquier momento después de t_{n+1} .

Con la estrategia no forzar se sabe cuándo pueden empezar a llegar a disco pero no cuando llegan.

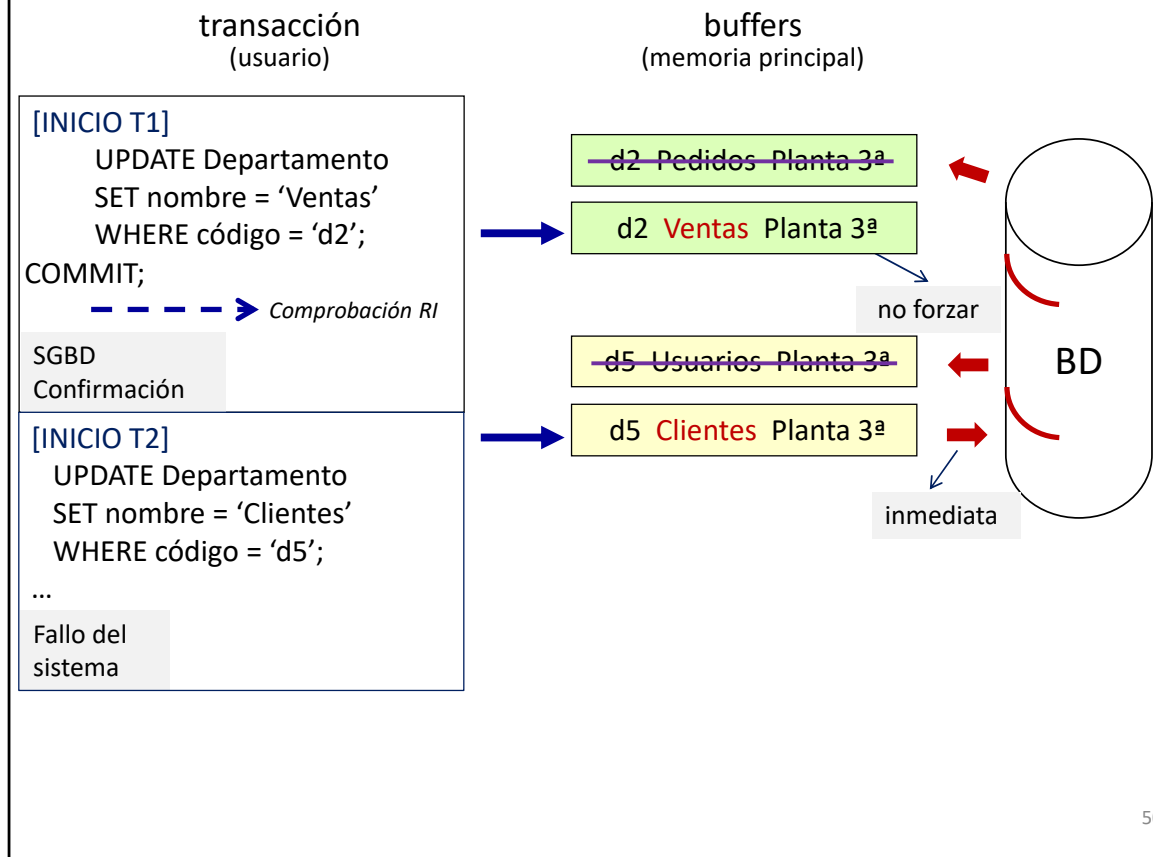
2 Estrategias de actualización en BD.



49

En el ejemplo, la transacción ha sido confirmada por el SGBD y sus actualizaciones deben estar en disco cuando se confirme la transacción definitivamente.

2 Estrategias de actualización en BD.



En el ejemplo, introducido en el punto 1 del tema, se puede observar que la política de transferencia de bloques flexible que se usa coincide con una estrategia de actualización de la BD **en el lugar, inmediata, y no forzar**:

- hay bloques actualizados por transacciones que no han finalizado (T2) que ya han sido transferidos a disco (**inmediata**);
- hay bloques actualizados por transacciones que han sido confirmadas (SGBD) (T1) que no han sido transferidos todavía a disco (**no forzar**).

Recuperación en bases de datos.

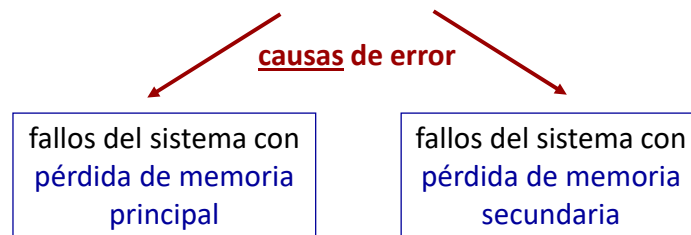
- 1 Recuperación de transacciones: introducción.
- 2 Estrategias de actualización en BD.
- 3 Estrategias de recuperación en BD.

3 Estrategias de recuperación en BD.

Objetivo de las técnicas de recuperación: ejecutar correctamente transacciones

¿Por qué una transacción puede no haber sido ejecutada correctamente?

Los cambios producidos por una **transacción confirmada** deben ser grabados en la BD en disco y una vez grabados no se deben perder: **objetivo (a)**



52

En primer lugar, se analiza **por qué motivos** una transacción puede no ser ejecutada correctamente en un SGBD. El análisis se hace tanto para las transacciones confirmadas (SGBD) como para las transacciones anuladas o interrumpidas por un fallo.

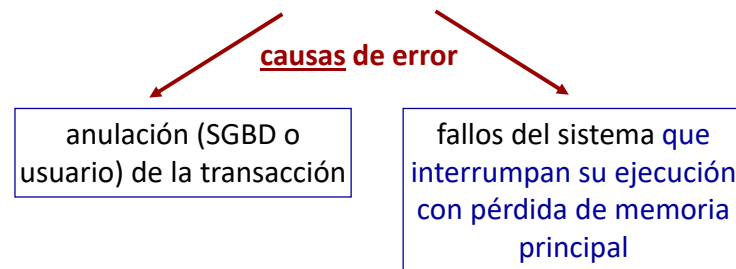
En las transparencias, se muestran las posibles causas por las que se incumplen los objetivos (a) y (b) del correcto procesamiento de transacciones.

3 Estrategias de recuperación en BD.

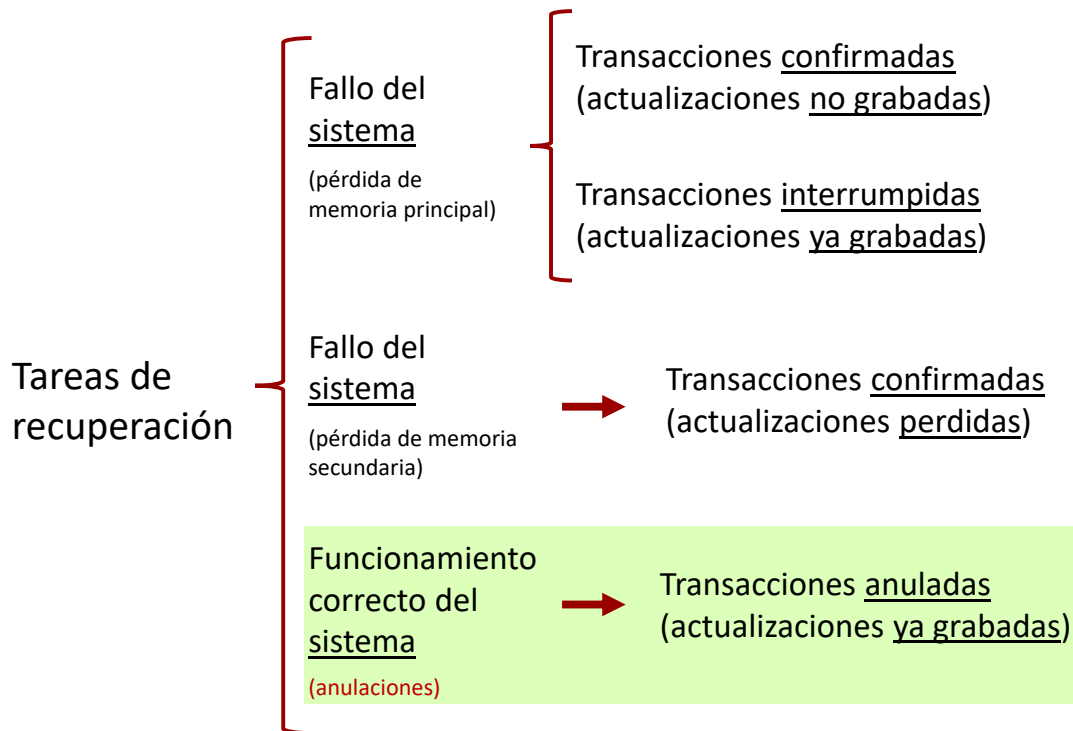
Objetivo de la técnicas de recuperación: ejecutar correctamente transacciones

¿Por qué una transacción puede no haber sido ejecutada correctamente?

Los cambios producidos por una transacción fallada (anulada o interrumpida) que ya han sido grabados en la BD en disco, deben ser deshechos: objetivo (b)



3 Estrategias de recuperación en BD.



54

Como se deduce del análisis anterior, existen **tres situaciones** en las que puede haber transacciones que no se han ejecutado correctamente: en la **ocurrencia de un fallo** (con pérdida de memoria principal o secundaria) y en la **anulación de un transacción** (usuario o SGBD) durante el correcto funcionamiento del SGBD. Es decir, se sabe **cuándo** el SGBD debe realizar tareas de recuperación.

En estas situaciones, el SGBD deberá hacer tareas de recuperación, pero las transacciones que serán objeto de recuperación serán distintas en cada situación. Es decir, se sabe de **qué** transacciones se debe preocupar el SGBD en caso de recuperación.

* Recordad que se está considerando un entorno monousuario en el que en cada momento sólo se está ejecutando una transacción.

3 Estrategias de recuperación en BD.

Recuperación en caso de **anulación**:

- ✓ Cuando una transacción T es anulada por algún motivo, el SGBD debe **deshacer** sus efectos (actualizaciones) como si la transacción no hubiese existido.
- ✓ El procedimiento para realizar la anulación dependerá de la estrategia de actualización de la BD seguida por el SGBD (inmediata, diferida).

55

La primera situación que se va a estudiar es la **anulación de transacciones**, durante el correcto funcionamiento del SGBD.

Esta situación se produce continuamente durante el uso del SGBD, ya que tanto el usuario como el SGBD pueden anular transacciones.

Una transacción anulada debe ejecutarse cumpliendo el objetivo (b), es decir, no debe dejar ningún efecto sobre la base de datos.

Las tareas de recuperación que debe hacer el SGBD en el momento de la anulación, para asegurar el objetivo (b), dependen de la estrategia de actualización de la BD utilizada en el SGBD.

3 Estrategias de recuperación en BD.

Recuperación en caso de **anulación**:

- ✓ El proceso de recuperación debe realizarse cuando se ejecuta la operación **anular(T)**.
- ✓ Deben deshacerse las actualizaciones de la transacción tanto en los buffers de memoria principal como en la BD en disco, a partir del diario.
- ✓ Cuando se ha eliminado el efecto de la transacción anulada, el SGBD graba una entrada [anular, T] en el buffer de diario.
- ✓ Durante la recuperación de la BD en caso de fallo del sistema, estas transacciones no son consideradas porque ya han sido recuperadas al anularse.

56

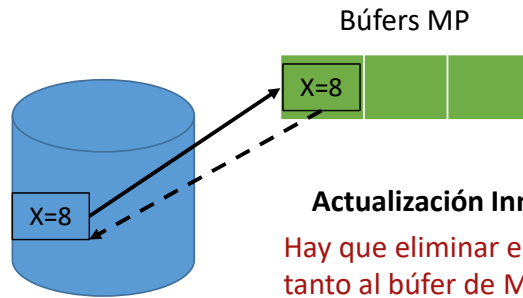
Es importante observar que una transacción anulada no debe dejar huella: ni en la base de datos en disco ni el buffer de memoria principal.

Otras transacciones se deben ejecutar, como si la transacción anulada no hubiese existido.

3 Estrategias de recuperación en BD.

Recuperación en caso de **anulación**:

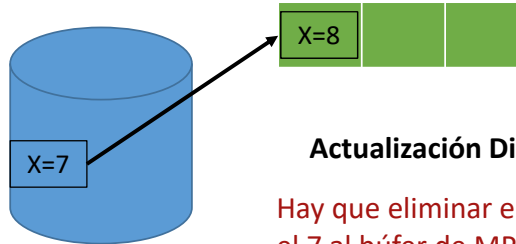
Transacción T
 t_0 Leer (X)
 t_1 $X \leftarrow 8$
 t_2 Escribir (X)
...
 t_n Anular



Actualización Inmediata

Hay que eliminar el 8 y devolver el 7 tanto al búfer de MP como al disco (donde puede haber llegado el 8 o no). El valor 7 está en el diario.

Transacción T
 t_0 Leer (X)
 t_1 $X \leftarrow 8$
 t_2 Escribir (X)
...
 t_n Anular



Actualización Diferida

Hay que eliminar el 8 y devolver el 7 al búfer de MP que nunca puede haber llegado a disco. El valor 7 está en el diario.

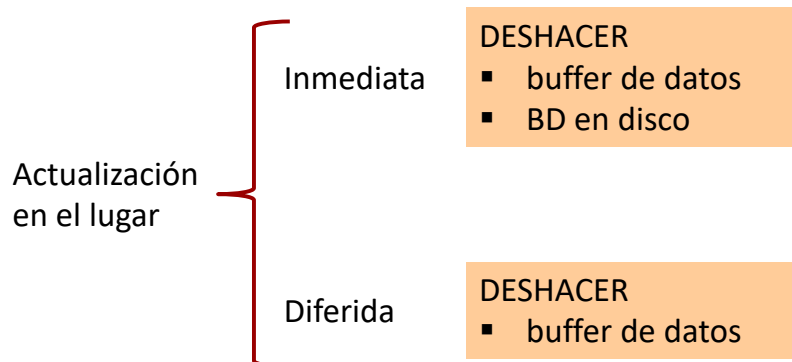
57

En el primer caso, con la actualización inmediata, el valor 8 “**puede**” haber llegado a disco por lo que la restauración del *valor_antes* (el 7) se realizará también en el disco (haya o no haya llegado el 8 a disco).

En el segundo caso, con la actualización diferida, se tiene la certeza de que el 8 no ha llegado a disco.

3 Estrategias de recuperación en BD.

Recuperación transacciones anuladas (funcionamiento correcto del sistema)



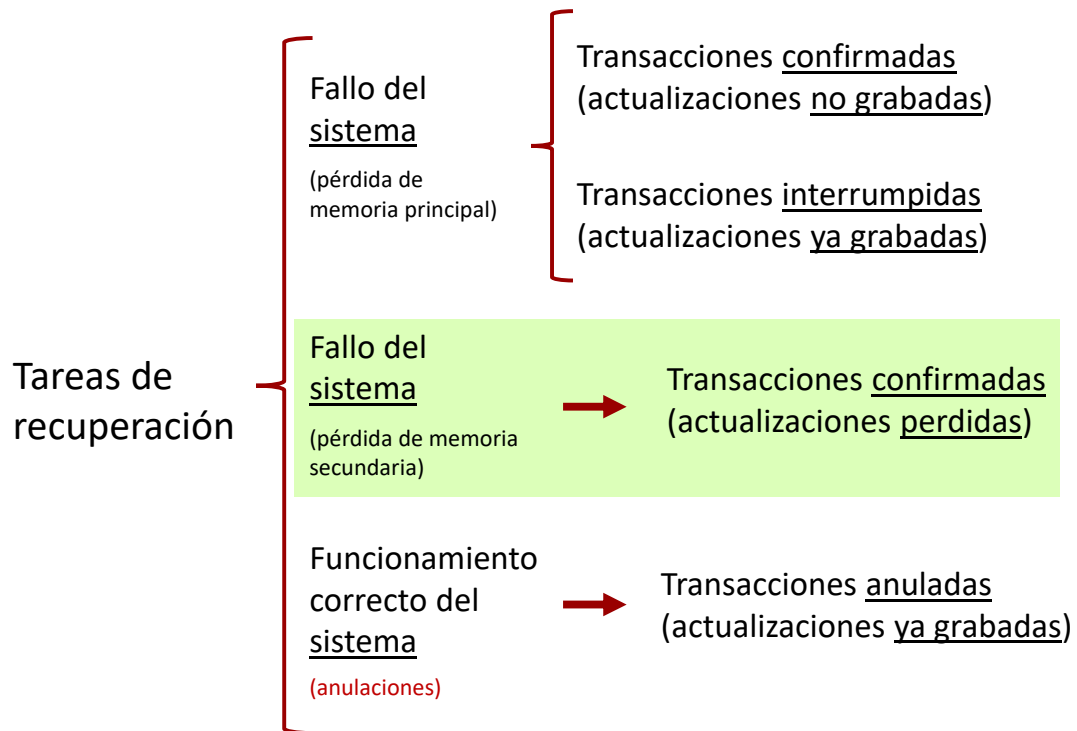
58

Dependiendo de la estrategia de actualización de la BD utilizada en el SGBD, anular una transacción implica tareas de recuperación distintas.

- Si la estrategia es **diferida**, se tiene la seguridad de que los bloques actualizados por la transacción anulada no han sido transferidos todavía a disco, por lo tanto sólo habrá que anular las actualizaciones, en los buffers de memoria principal.
- Si la estrategia es **inmediata**, los bloques actualizados por la transacción anulada pueden haber sido transferidos ya a disco, por lo tanto habrá que anular las actualizaciones, tanto en disco como en los buffers de memoria principal.

El que la estrategia sea **forzar/no forzar** es irrelevante en este caso ya que la transacción no ha sido confirmada.

3 Estrategias de recuperación en BD.



3 Estrategias de recuperación en BD.

Estrategia de recuperación de **transacciones confirmadas** frente a **fallos** del sistema de **almacenamiento secundario**

Herramienta de recuperación:

- **Diario**
- **Copias de seguridad**

Técnica de recuperación:



Sea cual sea la estrategia de actualización de la bd, cargar la base de datos a partir de la última **copia de seguridad** y a continuación **rehacer** todas las transacciones que aparecen confirmadas en el diario desde la fecha de la copia.

60

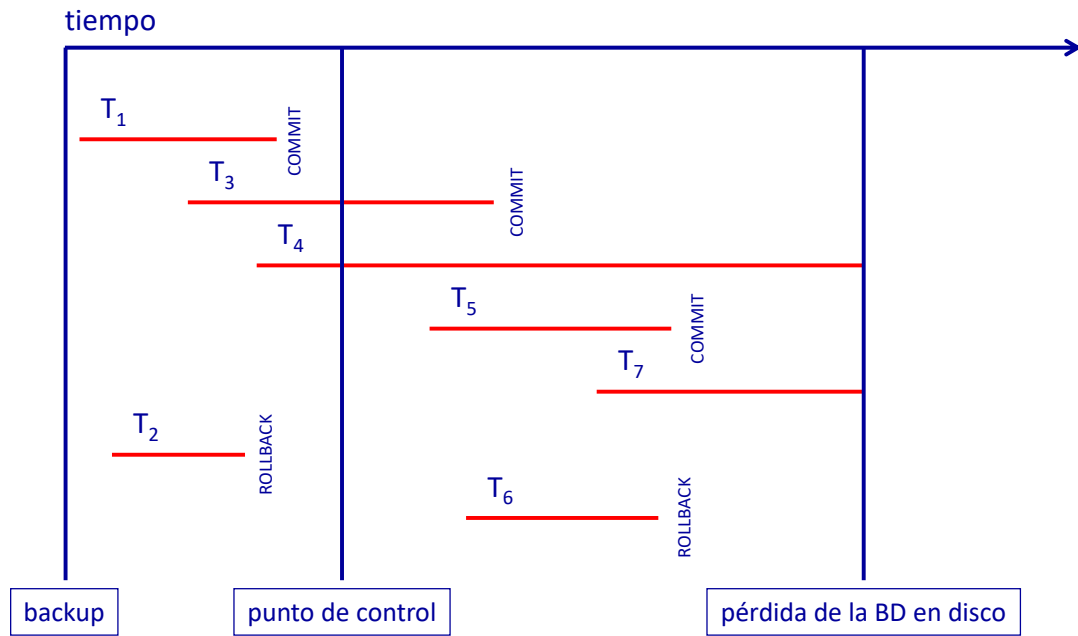
En el caso de que se produzca un fallo con pérdida de memoria secundaria, es decir, se pierda o se deteriore la base de datos, la técnica de recuperación es distinta.

El objetivo será recuperar la base de datos al estado más próximo en el tiempo al momento del fallo.

La forma de hacer la recuperación es la siguiente: se carga la base de datos a partir de su última copia de seguridad, y sobre ese estado se **rehacen** todas las transacciones que aparecen en el fichero de diario **confirmadas** después de la fecha de la copia.

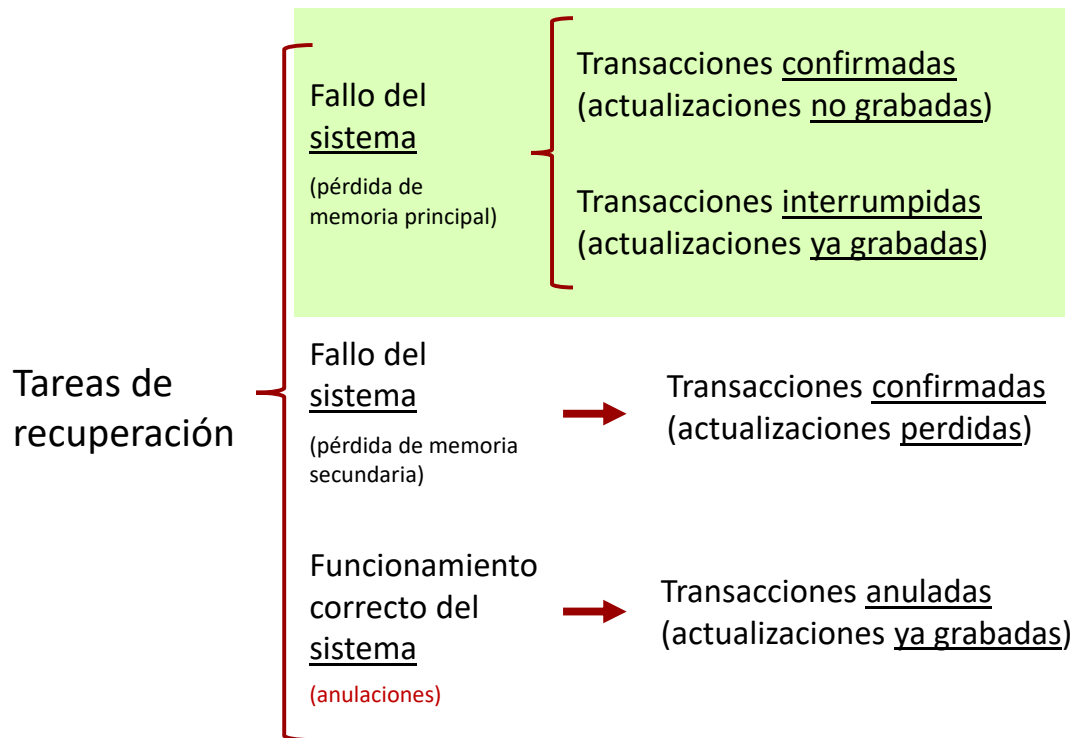
Para este tipo de recuperación, será muy importante tener prevista una política de copias de seguridad de la base de datos y del fichero de diario.

3 Estrategias de recuperación en BD.



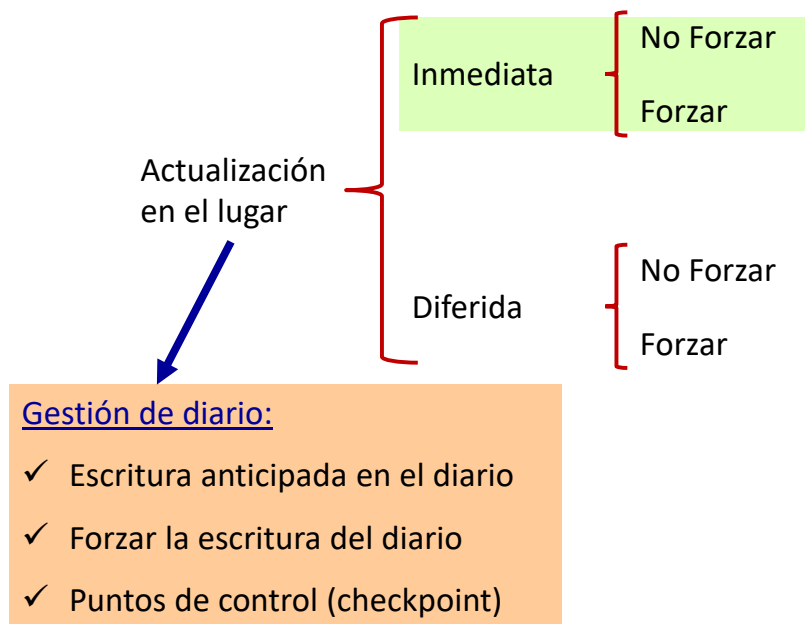
Cargar la base de datos a partir del *backup* y rehacer el efecto de las transacciones T₁, T₃, T₅ usando el diario.

3 Estrategias de recuperación en BD.



3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP



63

Como existen cuatro estrategias de actualización de la BD distintas, se van a analizar las tareas de recuperación, en caso de fallo, en cada una de estas cuatro estrategias.

Se empezará con las estrategias de actualización de la BD **inmediatas**:

- Inmediata–Forzar
- Inmediata–No Forzar

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Inmediata-No Forzar**

1. Es posible que las actualizaciones de las transacciones (bloques de datos actualizados) se graben en disco antes de que finalice la correspondiente transacción.
2. Los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, no serán, obligatoriamente, transferidos en la confirmación de la transacción.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Inmediata-No Forzar**

1. Las actualizaciones de la transacción interrumpida se deben **deshacer** ya que algunas de ellas pueden haber sido grabadas en disco. (Actualización inmediata).
2. Puede suceder que algunas de las actualizaciones de las transacciones confirmadas no hayan sido grabadas en disco antes del fallo, por lo tanto hace falta **rehacer** estas transacciones. (Estrategia no forzar).



Algoritmo DESHACER/REHACER

65

En la transparencia, aparece el análisis de la recuperación de transacciones, en el caso de fallo con pérdida de memoria principal, cuando la actualización es **inmediata-no forzar**.

El algoritmo incluye dos tareas, la tarea de **deshacer** las actualizaciones de la transacción interrumpida (sus actualizaciones pueden haber sido ya grabadas en disco) y la tarea de **rehacer** las actualizaciones de las transacciones confirmadas antes del fallo y después del último punto de control.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Algoritmo DESHACER/REHACER :

Usar dos listas de transacciones mantenidas por el sistema: la de transacciones confirmadas desde el último punto de control y la de la transacción activa*.

(1) **Deshacer** todas las operaciones *escribir(X)* de la transacción activa (interrumpida) a partir del diario en el orden inverso en que se escribieron en él, usando el procedimiento **DESHACER**.

(2) **Rehacer** todas las operaciones *escribir(X)* de las transacciones confirmadas antes del fallo, desde el último punto de control a partir del diario en el orden en que se escribieron en él, usando el procedimiento **REHACER**.

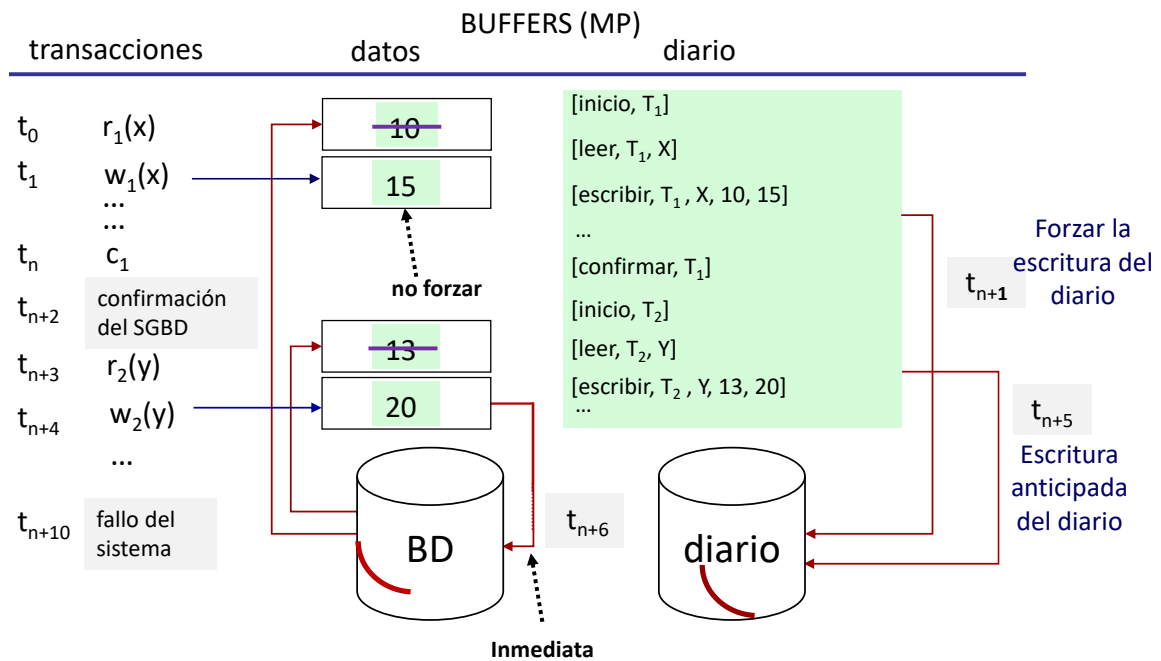
*En un entorno concurrente (Tema 4) podrá haber varias transacciones activas.

66

En el caso de actualización **inmediata-no forzar**, el algoritmo de recuperación es el **algoritmo DESHACER/REHACER** transacciones, que invoca a los procedimientos DESHACER y REHACER operaciones de actualización.

Se puede optimizar el proceso de recuperación durante la ejecución del paso (2) (REHACER) del algoritmo: en vez de aplicar secuencialmente las actualizaciones sobre un elemento de datos en el orden, en que se registraron en el diario, se puede iniciar el proceso desde el final del diario y grabar sólo la última actualización de cada elemento de datos. (Un elemento de datos puede haber sido actualizado por varias transacciones confirmadas).

3 Estrategias de recuperación en BD.



Actualización de la BD: en el lugar + **inmediata** + **no forzar**

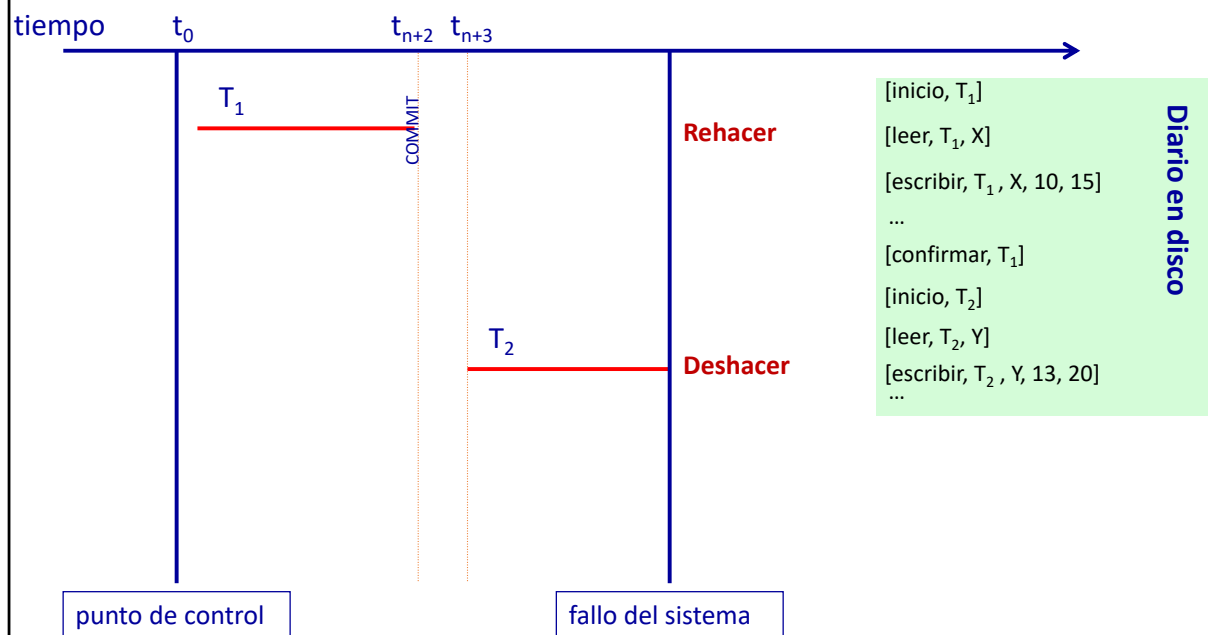
67

En el ejemplo, se muestra la ejecución de dos transacciones, en un SGBD con estrategia de actualización de la BD **inmediata-no forzar**.

Se muestra también la ejecución de los dos principios de gestión segura del diario.

3 Estrategias de recuperación en BD.

Algoritmo Deshacer/Rehacer



68

En el ejemplo anterior, después del fallo, el SGBD deberá realizar las tareas de recuperación que se muestran en la transparencia, por aplicación del **algoritmo DESHACER/REHACER**.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Inmediata-Forzar**

1. Es posible que las actualizaciones de las transacciones (bloques de datos actualizados) se graben en disco **antes** de que finalice la correspondiente transacción.
2. Los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, serán transferidos **antes** de que el SGBD confirme definitivamente la transacción.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Inmediata-Forzar**

1. Las actualizaciones de la transacción interrumpida* se deben **deshacer** ya que algunas de ellas pueden haber sido grabadas en disco. (Actualización inmediata).
2. Se tiene la seguridad de que las actualizaciones de las transacciones confirmadas han sido grabadas en disco, por lo tanto **no hace falta rehacer** estas transacciones. (Estrategia forzar).



Algoritmo DESHACER/NO REHACER

* En un entorno concurrente (Tema 4) podrá haber varias transacciones activas.

70

En la transparencia, aparece el análisis de la recuperación de transacciones, en el caso de fallo con pérdida de memoria principal, cuando la actualización es **inmediata-no forzar**.

El algoritmo incluye sólo la tarea de **deshacer** las actualizaciones de la transacción interrumpida (sus actualizaciones pueden haber sido ya grabadas en disco).

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Algoritmo DESHACER/NO REHACER :

Usar una lista de transacciones mantenida por el sistema: la de la transacción activa*.

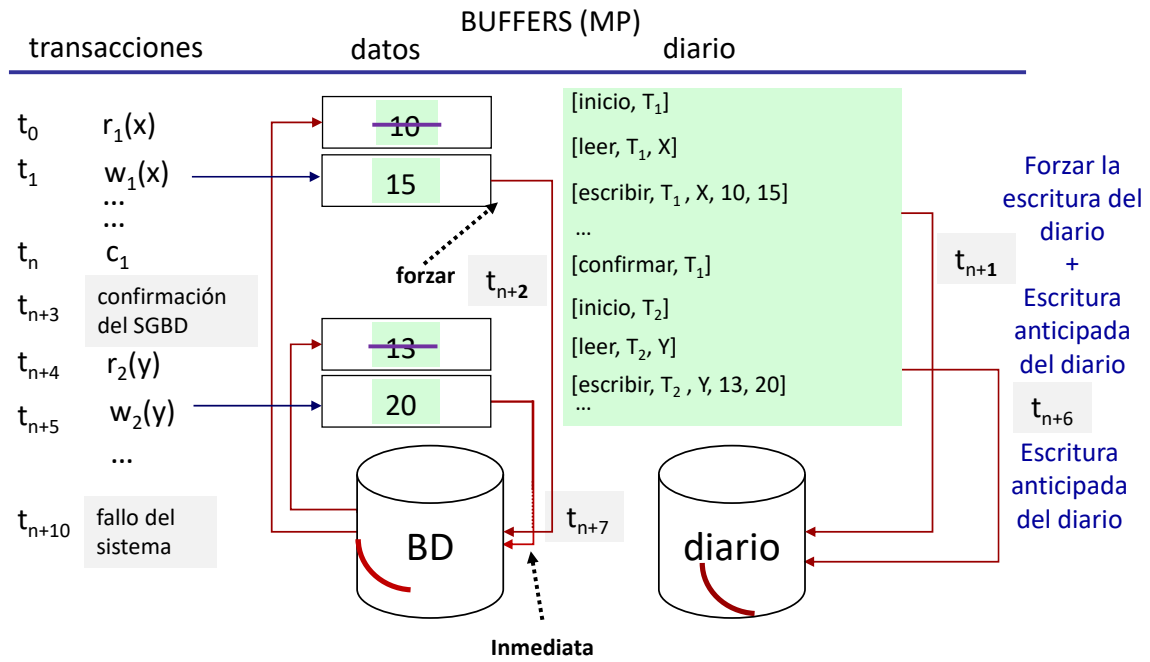
(1) **Deshacer** todas las operaciones escribir(X) de la transacción activa (interrumpida) a partir del diario en el orden inverso en que se escribieron en él, usando el procedimiento **DESHACER**.

*En un entorno concurrente (Tema 4) podrá haber varias transacciones activas.

71

En el caso de actualización **inmediata-forzar**, el algoritmo de recuperación es el **algoritmo DESHACER** transacciones, que invoca al procedimiento DESHACER operaciones de actualización.

3 Estrategias de recuperación en BD.



Actualización de la BD: en el lugar + **inmediata** + **forzar**

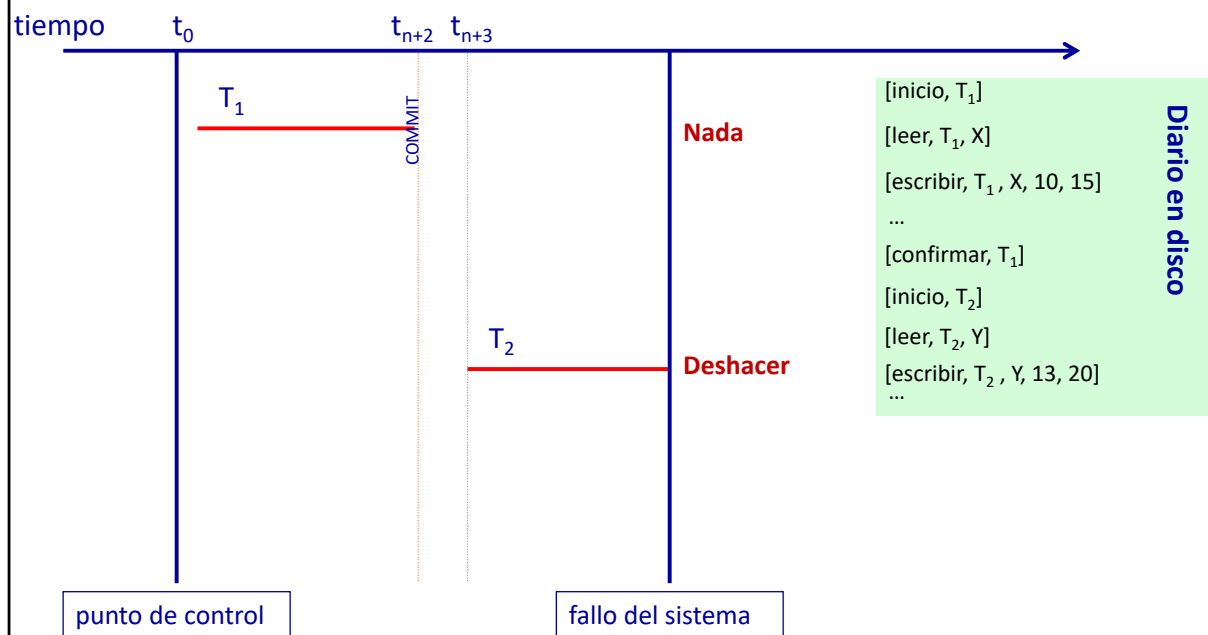
72

En el ejemplo, se muestra la ejecución de dos transacciones, en un SGBD con estrategia de actualización de la BD **inmediata-forzar**.

Se muestra también la ejecución de los dos principios de gestión segura del diario.

3 Estrategias de recuperación en BD.

Algoritmo Deshacer/No Rehacer

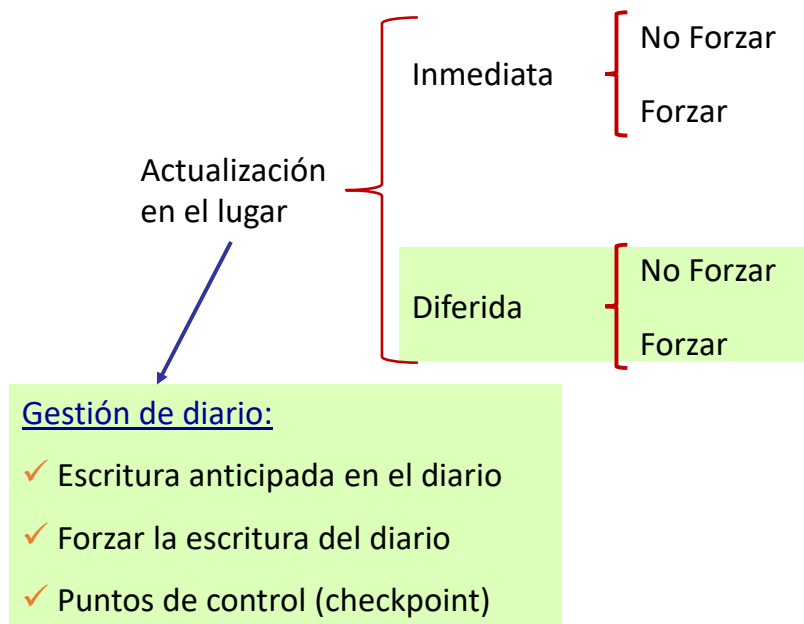


73

En el ejemplo anterior, después del fallo, el SGBD deberá realizar las tareas de recuperación que se muestran en la transparencia, por aplicación del **algoritmo DESHACER/NO REHACER**.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP



74

Se sigue con las estrategias de actualización de la BD **diferidas**:

Diferida–Forzar
Diferida–No Forzar

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Diferida-No Forzar**

1. Las actualizaciones de las transacciones (bloques de datos actualizados) no se pueden grabar en disco antes de que finalice (se confirme) la correspondiente transacción.
2. Los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, no serán transferidos, obligatoriamente, en la confirmación de la transacción.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Diferida-No Forzar**

1. Las actualizaciones de la transacción interrumpida* no han sido grabadas a disco **y no hace falta deshacer** sus actualizaciones. (Actualización diferida).
2. Puede suceder que algunas de las actualizaciones de las transacciones confirmadas no hayan sido grabadas en disco antes del fallo, por lo tanto hace falta **rehacer** estas transacciones. (Estrategia no forzar).



Algoritmo NO DESHACER/REHACER

*En un entorno concurrente (Tema 4) podrá haber varias transacciones activas.

76

En la transparencia, aparece el análisis de la recuperación de transacciones, en el caso de fallo con pérdida de memoria principal, cuando la actualización es **diferida-no forzar**.

El algoritmo sólo incluye la tarea de **rehacer** las actualizaciones de las transacciones confirmadas antes del fallo y después del último punto de control.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Algoritmo NO DESHACER/REHACER :

Usar una lista de transacciones mantenidas por el sistema: la de transacciones confirmadas desde el último punto de control.

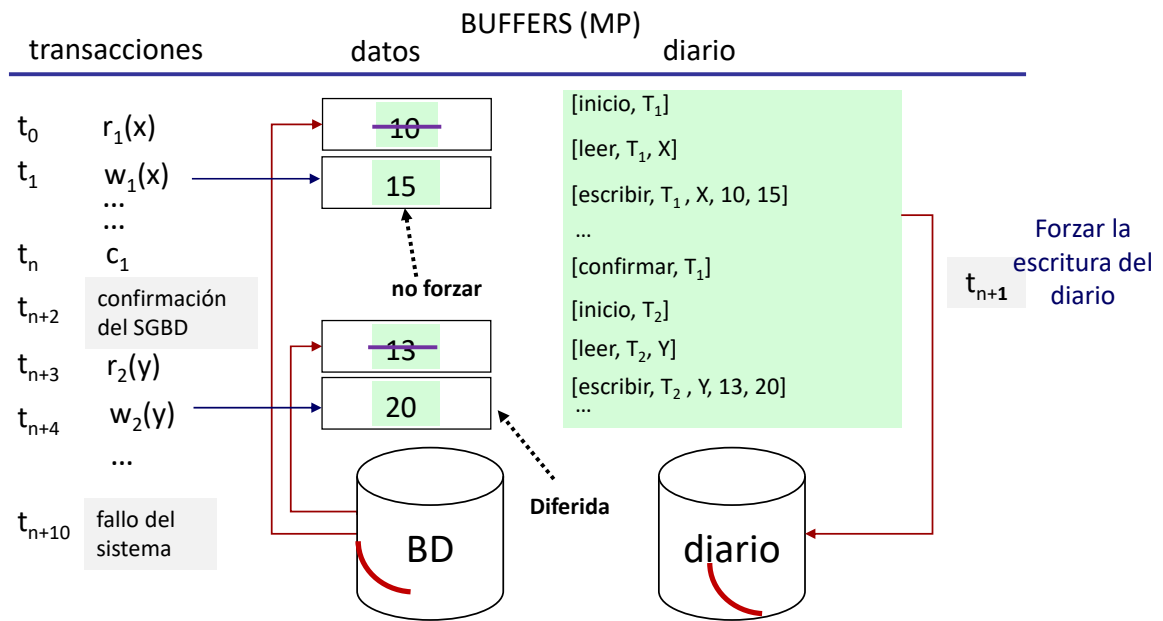
(1) **Rehacer** todas las operaciones escribir(X) de las transacciones confirmadas antes del fallo, desde el último punto de control a partir del diario en el orden en que se escribieron en él, usando el procedimiento **REHACER**.

77

En el caso de actualización **diferida-no forzar**, el algoritmo de recuperación es el **algoritmo REHACER** transacciones, que invoca al procedimiento REHACER operaciones de actualización.

Se puede optimizar el proceso de recuperación durante la ejecución del paso (1) (REHACER) del algoritmo: en vez de aplicar secuencialmente las actualizaciones sobre un elemento de datos, en el orden en que se registraron en el diario, se puede iniciar el proceso desde el final del diario y grabar sólo la última actualización de cada elemento de datos (un elemento de datos puede haber sido actualizado por varias transacciones confirmadas).

3 Estrategias de recuperación en BD.



Actualización de la BD: en el lugar + **diferida** + **no forzar**

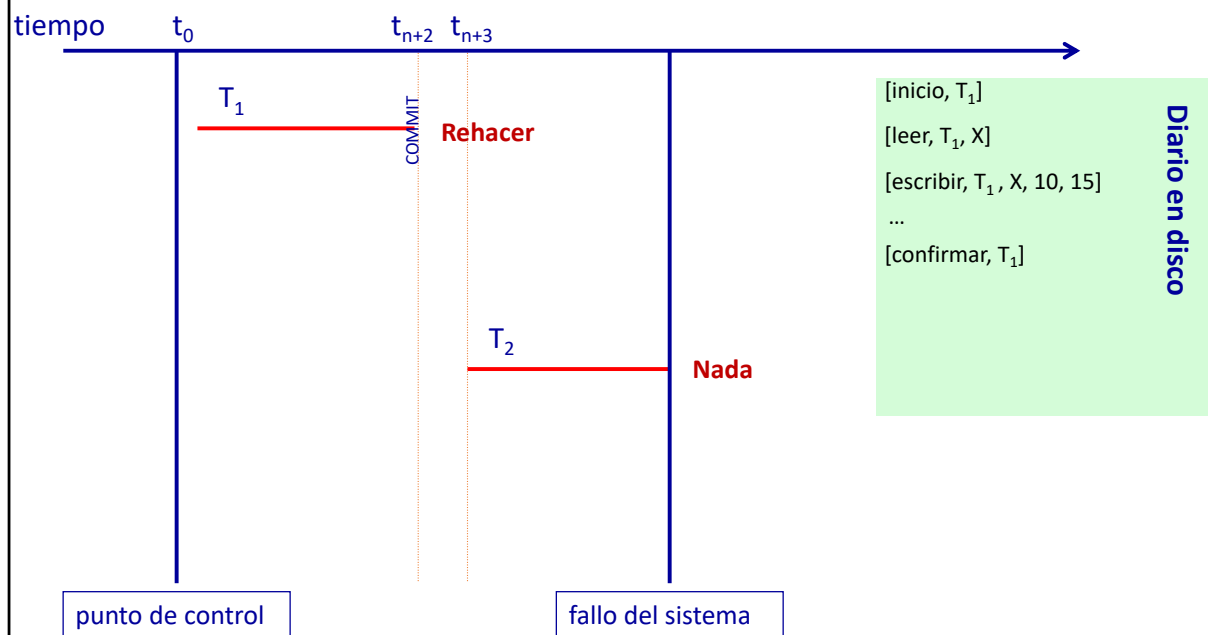
78

En el ejemplo, se muestra la ejecución de dos transacciones, en un SGBD con estrategia de actualización de la BD **diferida-no forzar**.

Se muestra también la ejecución de los principios de gestión segura del diario.

3 Estrategias de recuperación en BD.

Algoritmo No Deshacer/Rehacer



79

En el ejemplo anterior, después del fallo, el SGBD deberá realizar las tareas de recuperación que se muestran en la transparencia, por aplicación del **algoritmo NO DESHACER/REHACER**.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Diferida-Forzar**

1. Las actualizaciones de las transacciones (bloques de datos actualizados) no se pueden grabar en disco antes de que finalice (se confirme) la correspondiente transacción.
2. Los bloques actualizados por una transacción, que no hayan sido transferidos todavía a disco, serán transferidos antes de que el SGBD confirme definitivamente la transacción.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Actualización **Diferida-Forzar**

1. Las actualizaciones de la transacción interrumpida* no han sido grabadas a disco **y no hace falta deshacer** sus actualizaciones. (Actualización diferida).
2. Se tiene la seguridad de que las actualizaciones de las transacciones confirmadas han sido grabadas en disco, por lo tanto **no hace falta rehacer** estas transacciones. (Estrategia forzar).



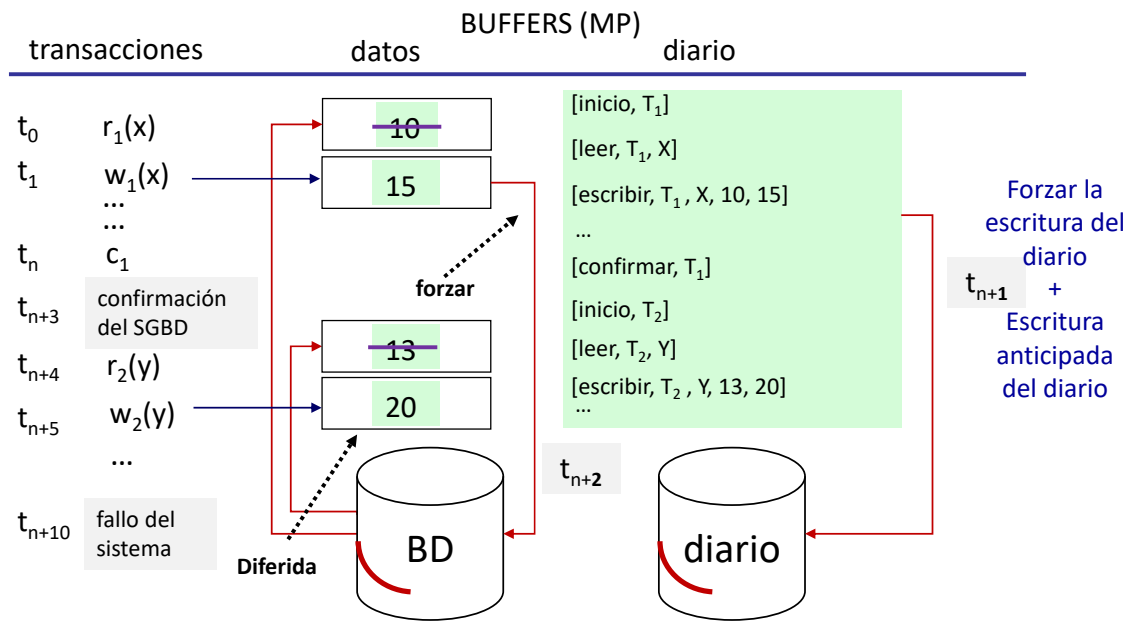
Algoritmo NO DESHACER/NO REHACER

* En un entorno concurrente (Tema 4) podrá haber varias transacciones activas.

81

En la transparencia, aparece el análisis de la recuperación de transacciones, en el caso de fallo con pérdida de memoria principal, cuando la actualización es **diferida-forzar**. Obsérvese que en el algoritmos **no se incluye** la tarea de **DESHACER** las actualizaciones de la transacción interrumpida (sus actualizaciones no pueden haber sido grabadas en disco) ni la tarea de **REHACER** las actualizaciones de las transacciones confirmadas, antes del fallo (desde el último punto de control).

3 Estrategias de recuperación en BD.



Actualización de la BD: en el lugar + **diferida** + **forzar**

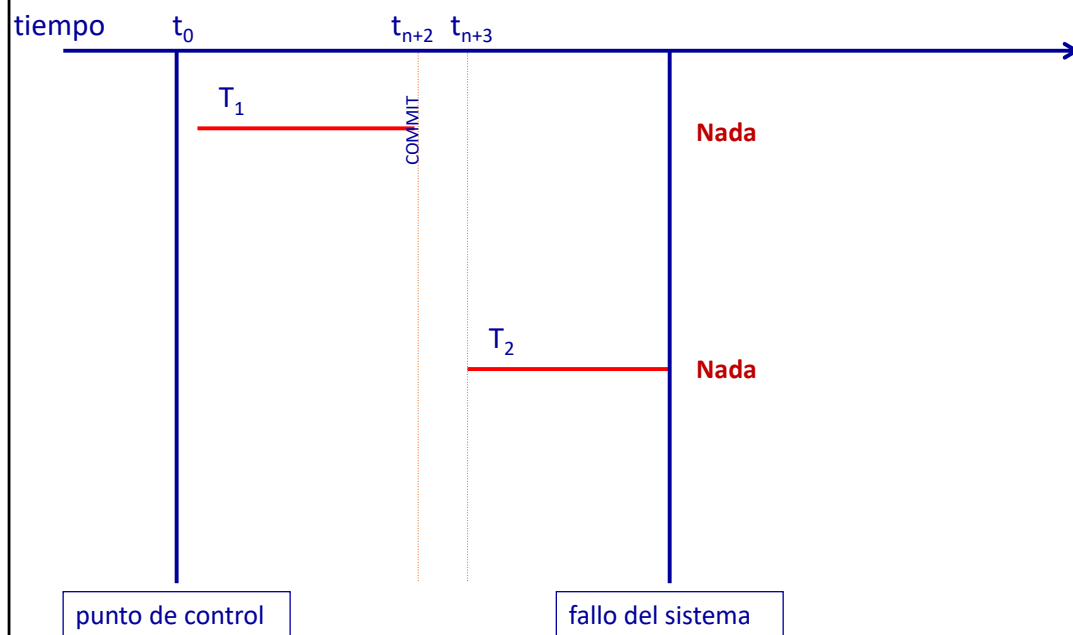
82

En el ejemplo, se muestra la ejecución de dos transacciones, en un SGBD con estrategia de actualización de la BD **diferida-forzar**.

Se muestra también la ejecución de los principios de gestión segura del diario.

3 Estrategias de recuperación en BD.

Algoritmo No Deshacer/No Rehacer



83

En el ejemplo anterior, después del fallo, el SGBD deberá realizar las tareas de recuperación que se muestran en la transparencia, por aplicación del **algoritmo NO DESHACER/NO REHACER**.

3 Estrategias de recuperación en BD.

Recuperación BD ante fallo del sistema con pérdida de MP

Procedimiento DESHACER ([escribir, T, X, valor_antes, valor_despues])

Asignar al elemento de datos X en la base de datos (en disco) el **valor_antes**.

Procedimiento REHACER ([escribir, T, X, valor_antes, valor_despues])

Asignar al elemento de datos X en la base de datos (en disco) el **valor_despues**.

84

En la transparencia, se presentan los dos procedimientos **DESHACER** y **REHACER**, para deshacer y rehacer actualizaciones a partir de la información registrada en el fichero de diario, en disco.

Ambos procedimientos tienen como argumento una entrada del diario correspondiente a una operación de actualización (**escribir**).

3 Estrategias de recuperación en BD.

Resumen

Tareas de recuperación

Fallo del sistema

(pérdida de memoria principal)

Transacciones confirmadas
(actualizaciones no grabadas)

Transacciones interrumpidas
(actualizaciones ya grabadas)

Funcionamiento correcto del sistema

(anulaciones)

Transacciones anuladas
(actualizaciones ya grabadas)

¿Cuándo?

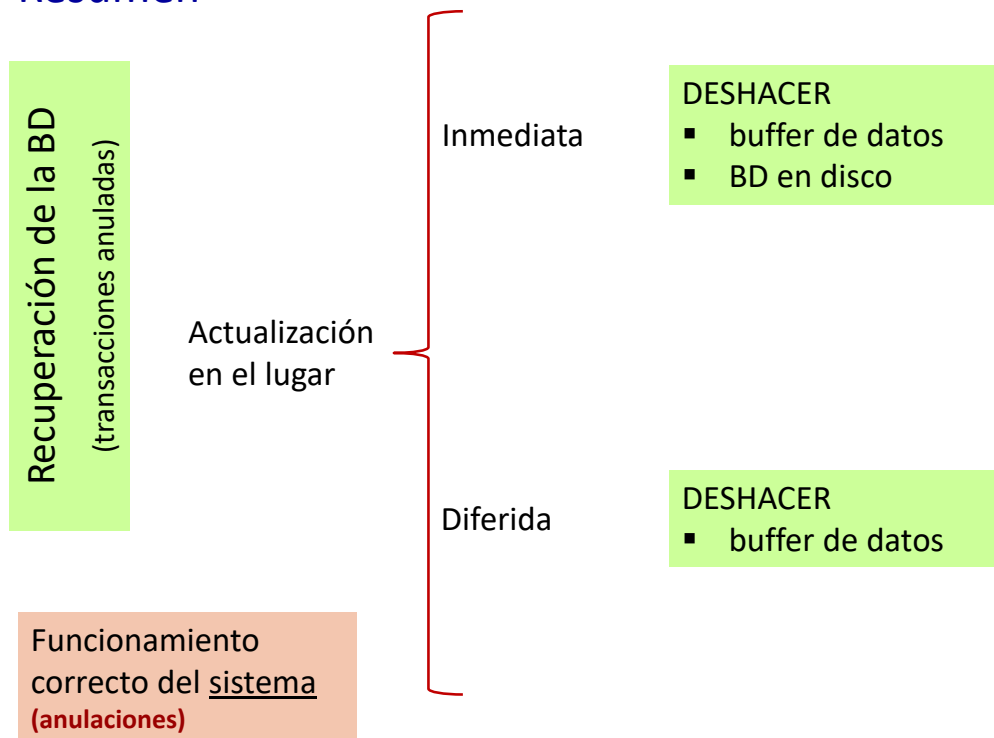
¿Qué?

85

Para finalizar el estudio de la recuperación de transacciones en bases de datos, se presenta un resumen de todo lo visto.

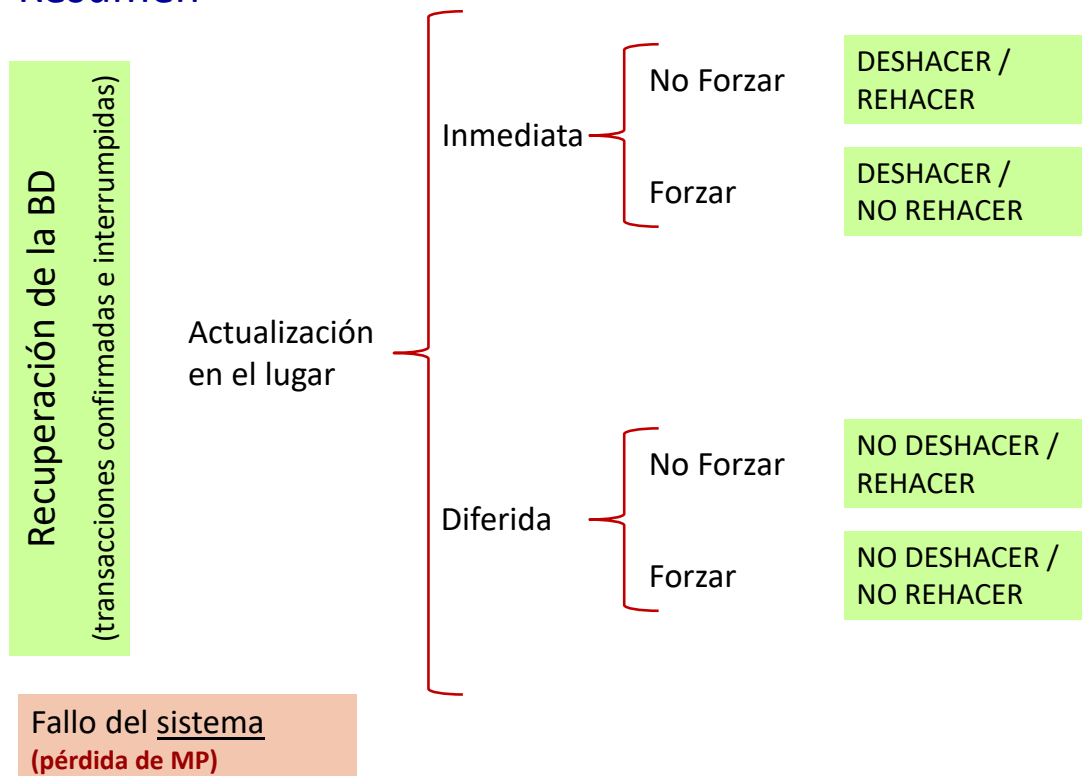
3 Estrategias de recuperación en BD.

Resumen



3 Estrategias de recuperación en BD.

Resumen



3 Estrategias de recuperación en BD.

Recuperación en caso de fallos con actualización en el lugar

En cualquiera de los algoritmos de recuperación se asegura que el proceso de recuperación es idempotente, es decir ejecutarlo una y otra vez debe ser equivalente a ejecutarlo una única vez.



Si durante el proceso de recuperación ocurre un fallo, el siguiente intento de recuperación restaurará la BD: el resultado de la recuperación después de un fallo durante la recuperación es el mismo que el resultado de la recuperación cuando no ha habido fallo durante la recuperación.