Conceptos sobre procesamiento de transacciones

Tema 3: Bases de Datos II

Contenidos del tema 3

1. Introducción
2. Propiedades deseables en las transacciones.
3. Conceptos de transacciones y sistemas.
4. Planificaciones y concurrencia.
   - Planificaciones en serie.
   - Planificaciones serializables.
5. Planificaciones y recuperabilidad.
6. Definición de transacciones en SQL.

Introducción

- La ejecución concurrente de programas de diversos usuarios es esencial para un uso eficiente de una BD multiusuario
- Los (programas de) usuarios realizan diversas operaciones sobre la BD y el SGBD se encarga de controlar qué datos se leen o escriben en la BD
- Una transacción es una abstracción desde el punto de vista del usuario:
  - es una secuencia de operaciones de lectura y escritura de datos originadas por un (programa de) usuario que llevan una base de datos de un estado consistente a otro estado también consistente.

Introducción

- **Modelo simplificado de BD**:
  - Colección de elementos de datos con nombre (gránulos).
  - Tamaño de cada elemento se le llama granularidad.
    - (podría ser campo, registro, bloque o tabla)

- Una **acción** es una operación de procesamiento primitivo e indivisible ejecutado por un único usuario sobre un gránulo.

- Una **transacción** es una secuencia de acciones ejecutadas por usuario determinado que respeta la consistencia de BD.

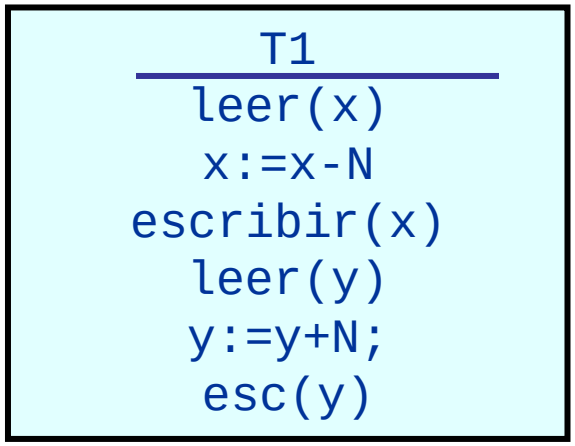
Introducción

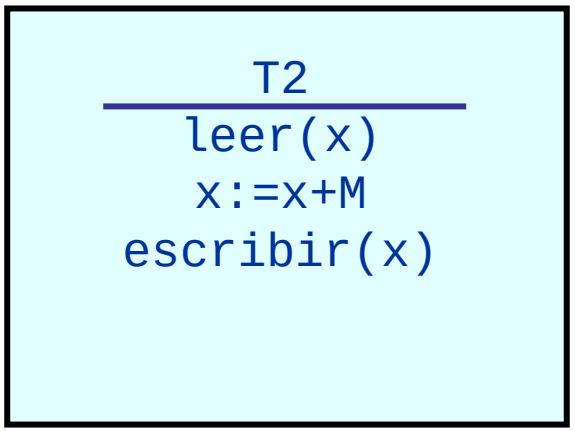
- Definiciones implicadas:

  - Restricciones o reglas de integridad son **auto-consistentes** si no contienen contradicciones.
  - Una BD es **consistente** si sus reglas son auto-consistentes y si los datos no violan dichas reglas.

Introducción

- Acciones básicas que se consideran para el modelo de BD especificado:
  - lee(x): lee un gránulo de la BD y lo guarda en una variable del programa.
  - escribe(x): escribe el valor de la variable de programa en el elemento de la BD llamado x.
- Una transacción estará formada por operaciones básicas de lectura y escritura.

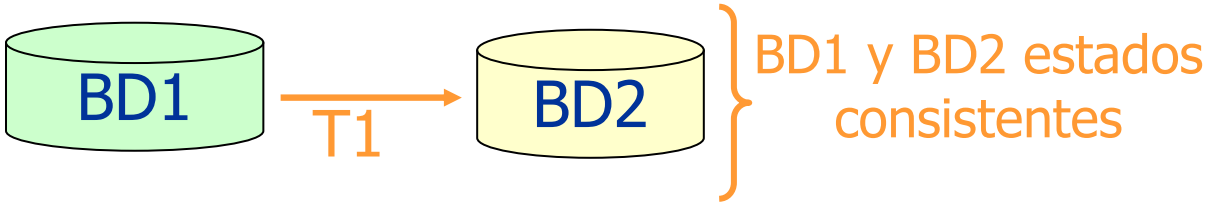




Introducción

- Característica fundamental de una transacción correcta: pasa de un estado consistente de la BD a otro también consistente

(aunque puede ser temporalmente inconsistente durante la misma).



- Nota: es tarea del usuario diseñar correctamente las transacciones y del módulo encargado de conservar la integridad detectar las transgresiones.

- Esto asegura la consistencia de la BD si el procesamiento de las transacciones fuera siempre secuencial, sin concurrencia de transacciones.

Introducción

- Las circunstancias que debe contemplar el SGBD respecto a las transacciones son:
  - efectos de **transacciones concurrentes** (*interleaving*)
    - el mejor aprovechamiento de los recursos computacionales y la reducción de los tiempos de respuesta exige la concurrencia de transacciones en BD multiusuario
    - se profundiza en el control de concurrencia en el Tema 4
  - efectos de **fallos en el sistema**
    - el sistema o las propias transacciones pueden fallar por diversas causas
    - se profundiza en las técnicas de recuperación en el Tema 5

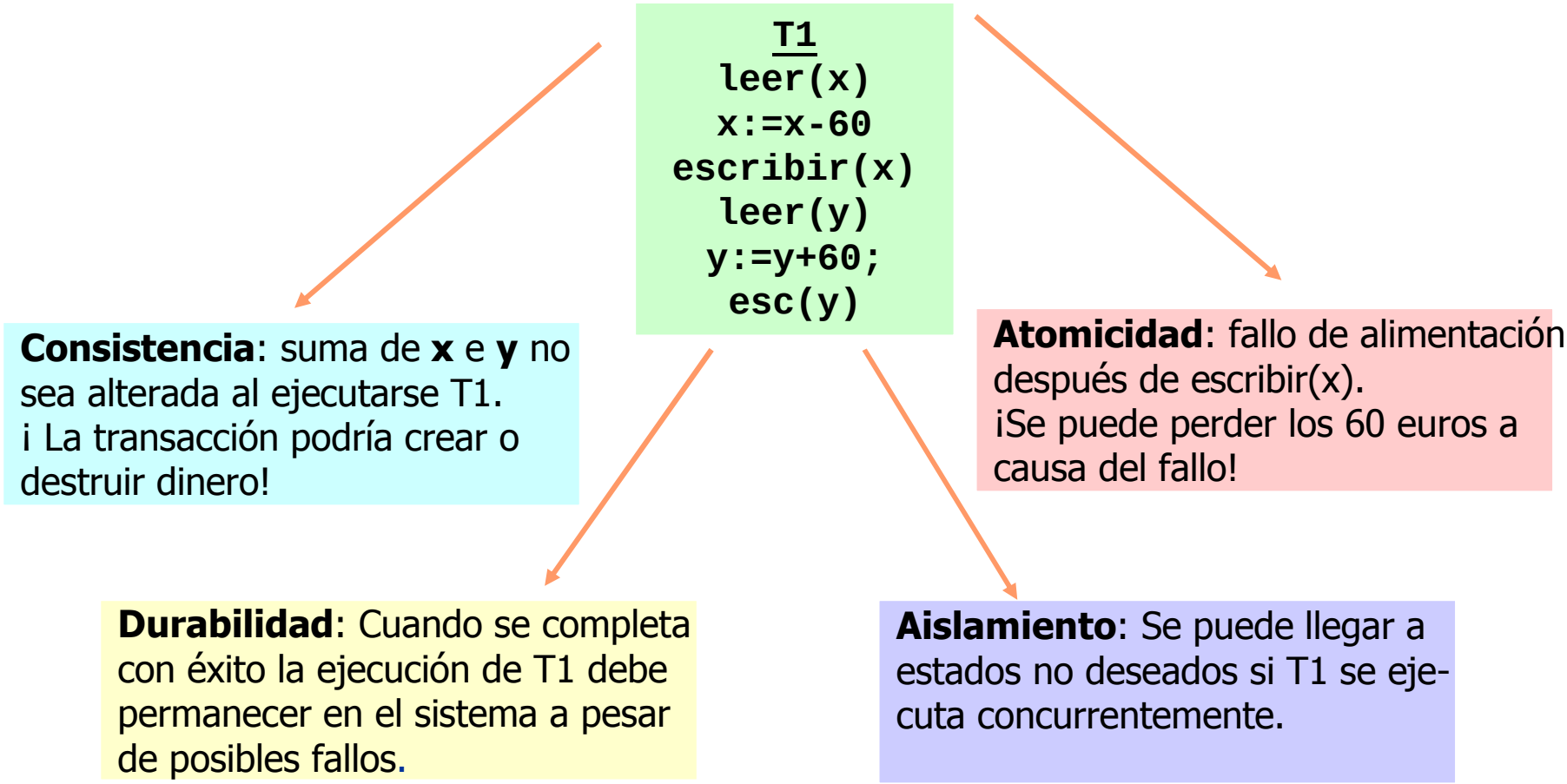
Propiedades deseables de las transacciones

- Las transacciones y su procesamiento deben poseer varias propiedades (**ACID**) y su cumplimiento debe asegurarlo el SGBD:
  - **Atomicidad:** una transacción o se ejecuta completamente o no tiene efecto alguno (responsab. del SGBD - subsistema de recuperación).
  - **Conservación de la consistencia:** toda transacción que parte de un estado consistente deja la BD en un estado consistente (responsabilidad de los programadores y/o del módulo del SGBD que asegura el cumplimiento de las reglas de integridad).

Propiedades deseables de las transacciones

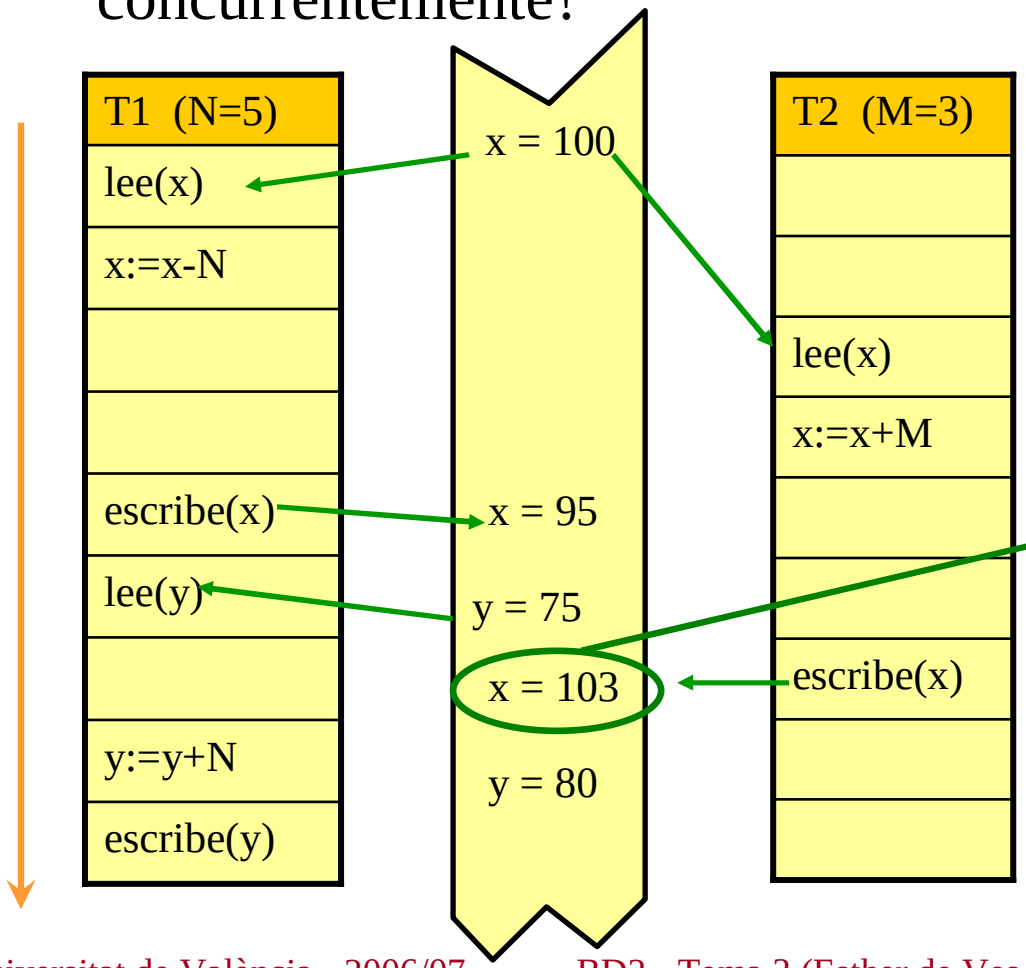
- Las transacciones y su procesamiento deben poseer varias propiedades (**ACID**) y su cumplimiento debe asegurarlo el SGBD:
  - **Aislamiento *(Isolation)*:** los efectos de una transacc. no se ven influenciados por otras transacciones concurrentes (responsabilidad del SGBD - subsistema de control de concurrencia)
  - **Durabilidad o permanencia:** los efectos de una transacción una vez confirmada son permanentes (responsabilidad del SGBD - subsistema de recuperación).

Propiedades deseables de las transacciones



Conceptos de transacciones y sistemas. Concurrencia.

- ¿Qué puede pasar cuando las transacciones se ejecutan concurrentemente?

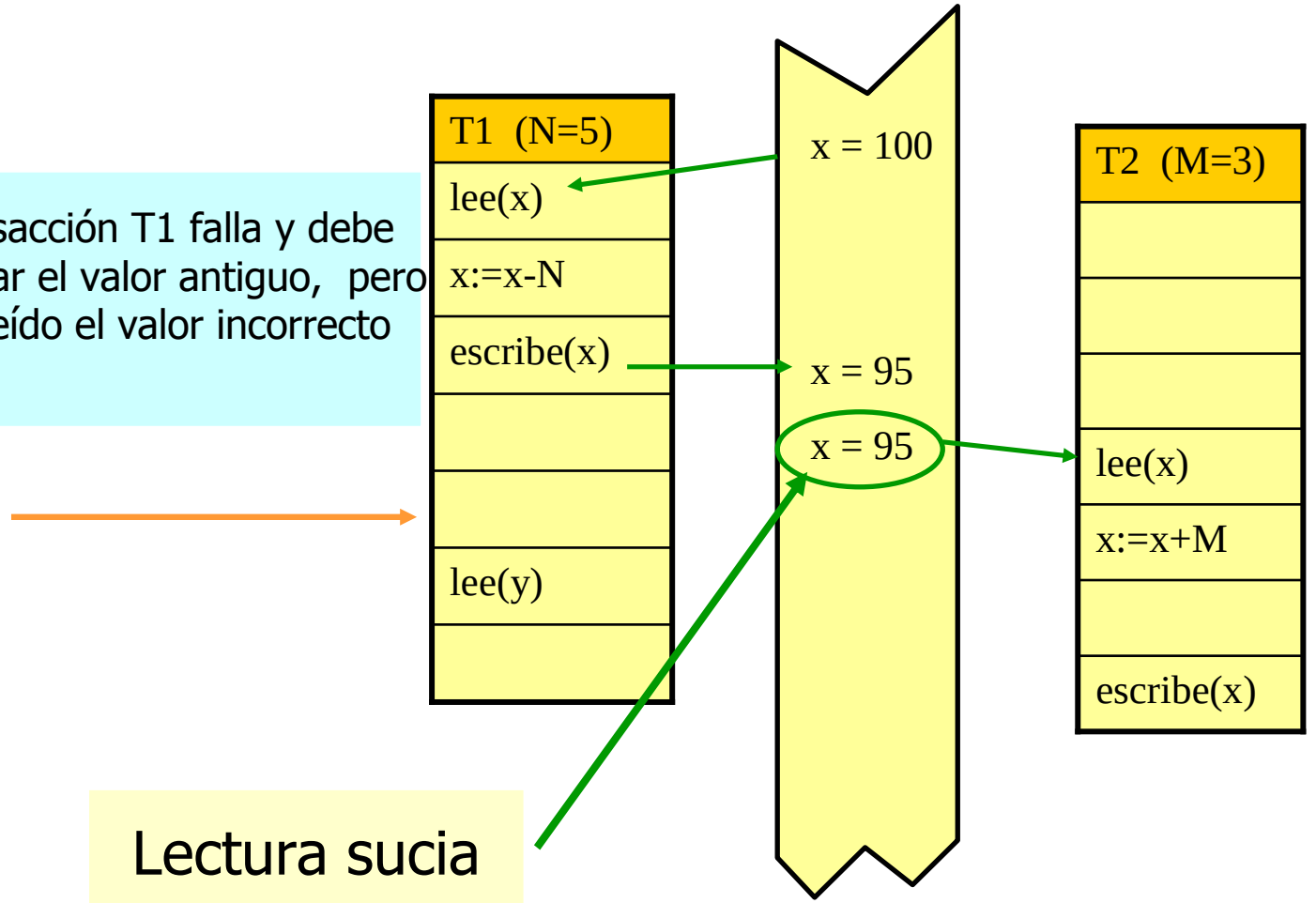


El elemento x tiene un valor incorrecto porque la actualización realizada por T1 se ha perdido

Actualización perdida

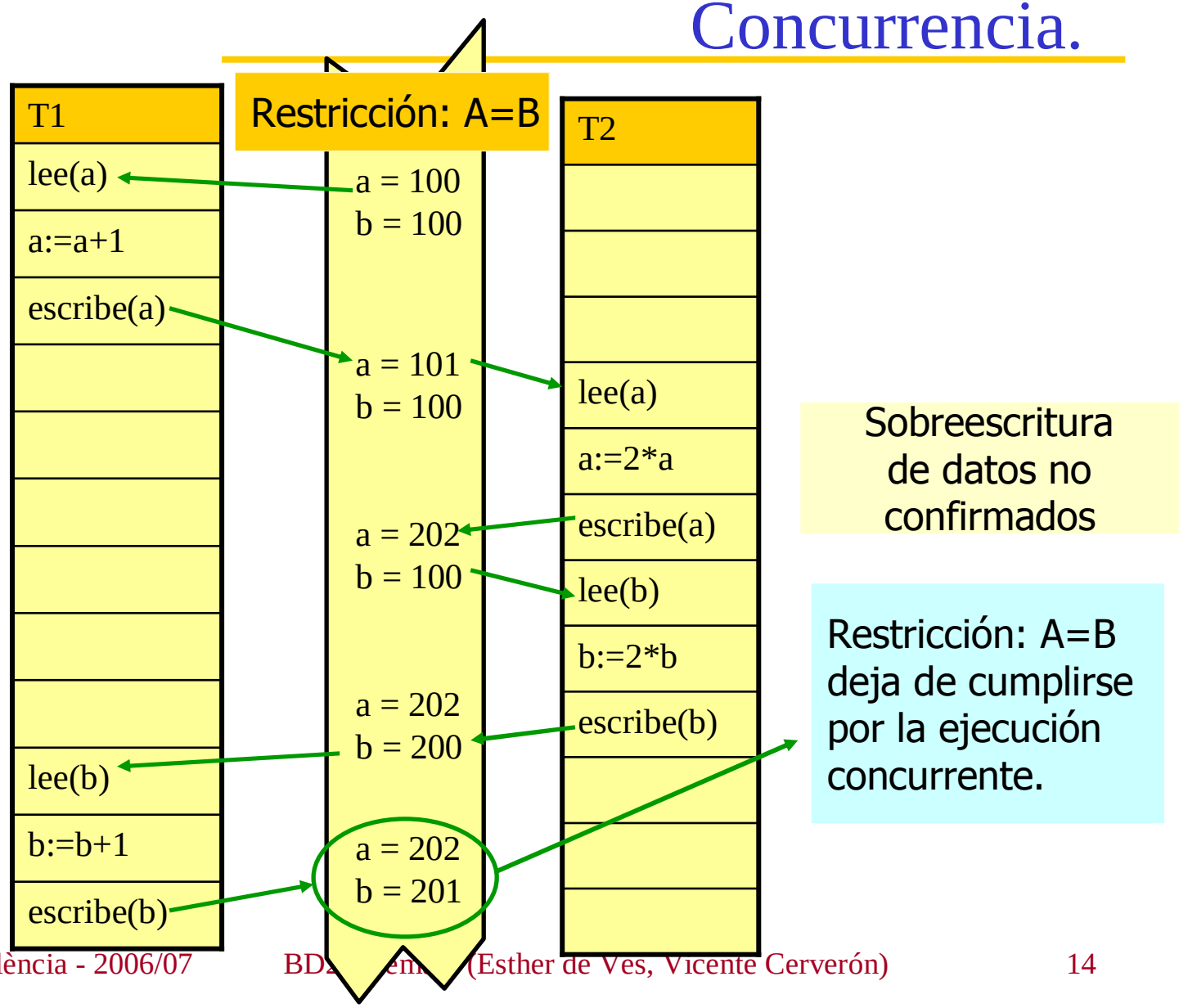
Conceptos de transacciones y sistemas. Concurrencia.

La transacción T1 falla y debe restaurar el valor antiguo, pero T2 ha leído el valor incorrecto de X.

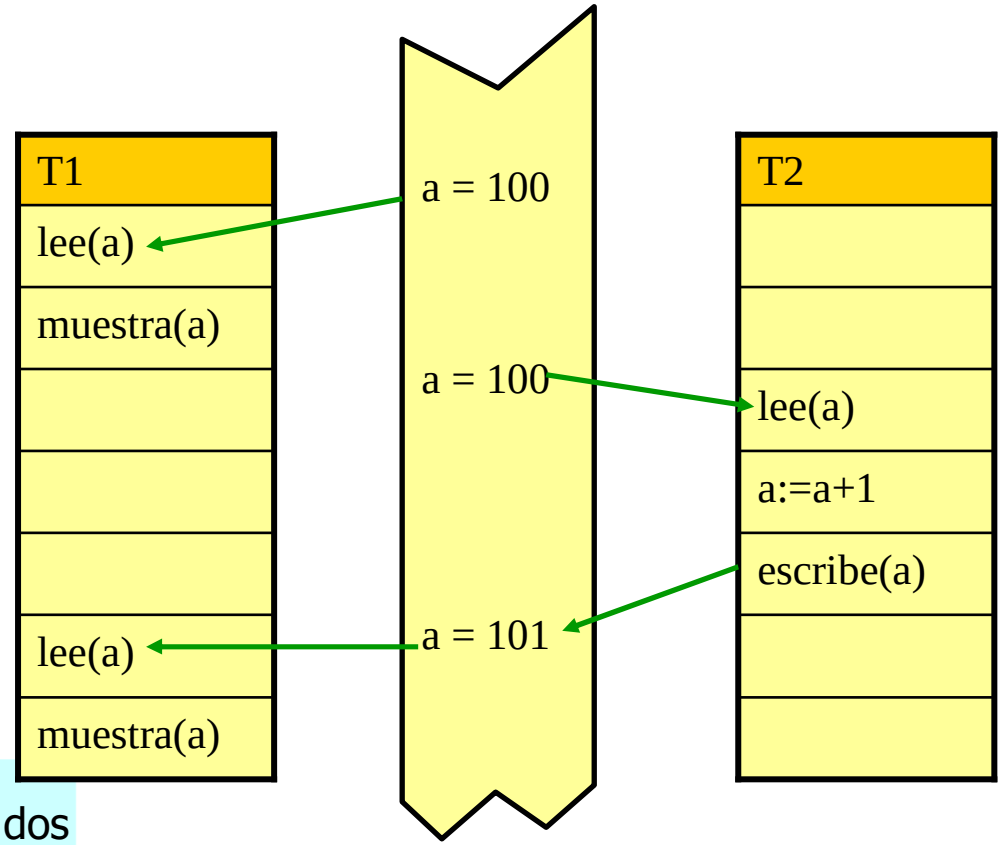


Lectura sucia

Conceptos de transacciones y sistemas. Concurrencia.



Conceptos de transacciones y sistemas. Concurrencia.



La transacción T1 lee dos valores diferentes para el dato **a**, en una misma transacción.
¡Extraño!

Lecturas no reproducibles

Conceptos de transacciones y sistemas. Concurrencia.

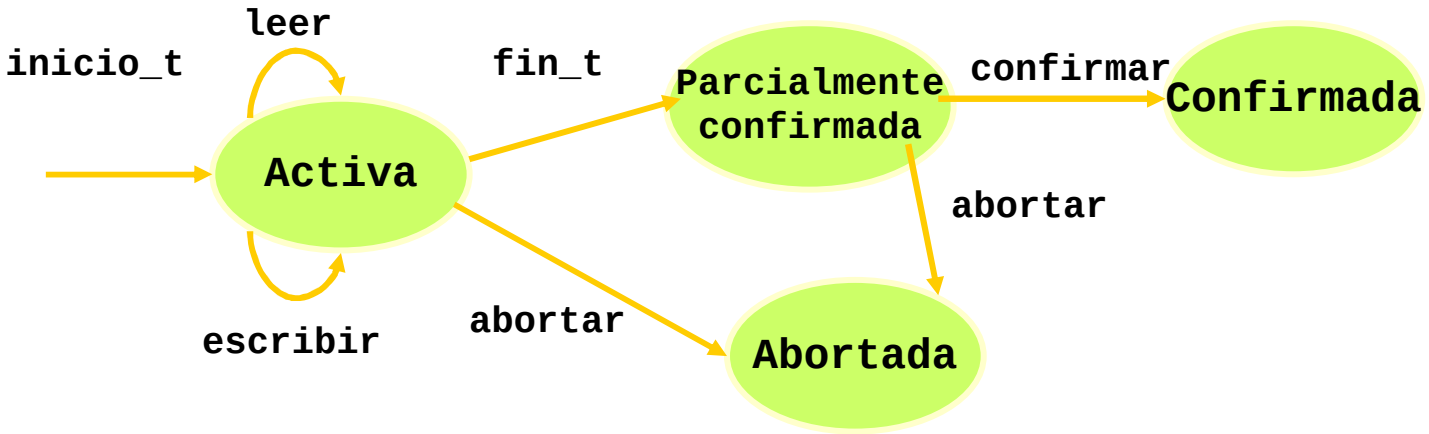
- El SGBD debe asegurarse del aislamiento y la conservación de la consistencia
  - los efectos de una transacción no deben verse influenciados por otras transacciones concurrentes
  - la ejecución completa (y confirmada) de una transacción, aún en concurrencia con otras, debe asegurar el mantenimiento de la consistencia de la BD
  - una transacción debe ejecutarse como si fuera la única transacción que estuviera ejecutándose en la BD en ese preciso momento

Conceptos de transacciones y sistemas. Recuperación.

- El SGBD debe asegurar la atomicidad y durabilidad
  - O: Todas las operaciones se superen con éxito y su efecto queda registrado en la BD.
  - O: La transacción no tenga efecto alguno sobre el sistema y el resto de transacciones.
- Diversos fallos pueden poner en peligro dichas condiciones:
  1. Fallo del ordenador (caída del sistema).
  2. Error en una transacción (errores lógicos de programación).
  3. Imposición del control de concurrencia.
  4. Problemas físicos y catástrofes.
- El sistema debe mantener suficiente información para lograr recuperarse del fallo (con técnica WAL write-ahead-log) .
- Con objeto de garantizar la recuperación, el sistema necesita mantenerse al tanto del estado de la transacción.

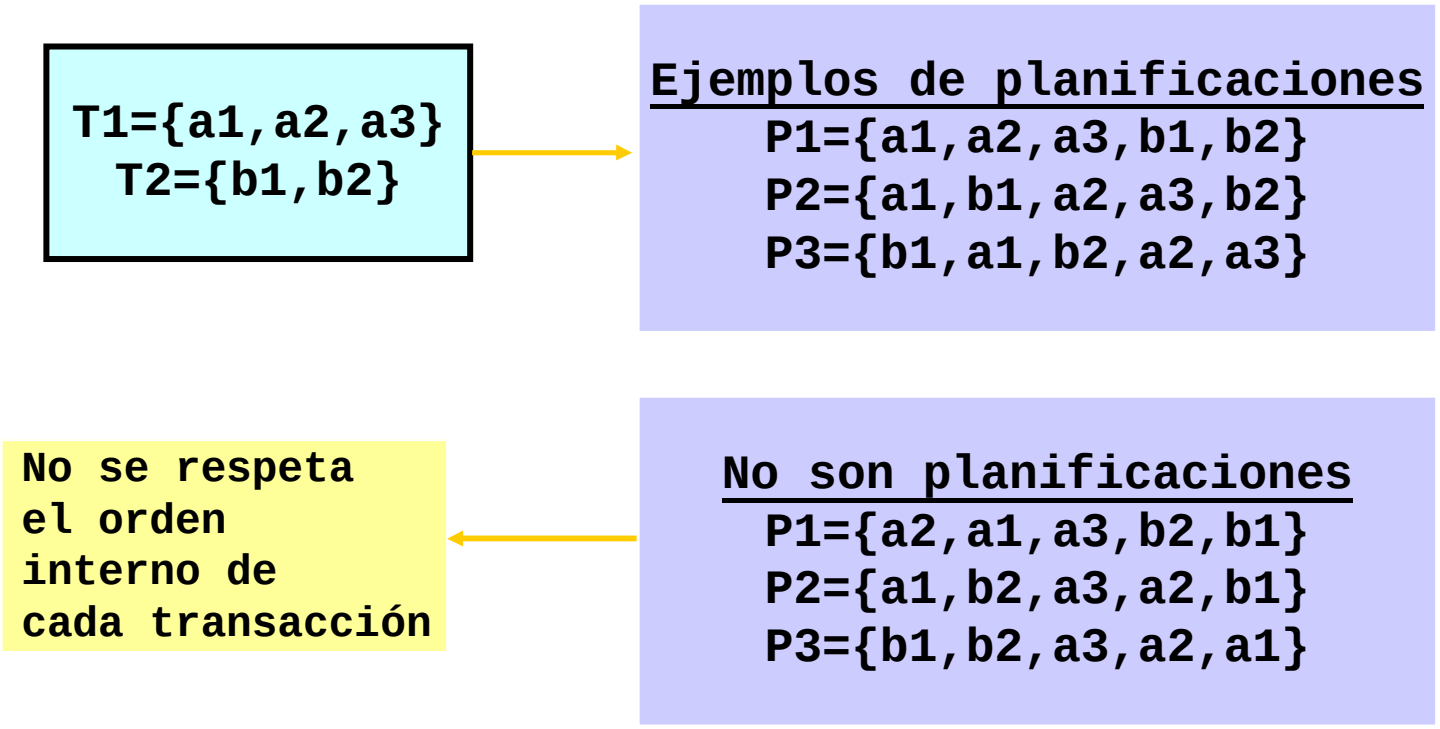
Conceptos de transacciones y sistemas. Estados de las transacciones.

- Operaciones registrables de las transacciones:
  - inicio_t
  - leer (gránulo y valor leído)
  - escribir (gránulo, valor escrito y valor anterior)
  - fin_t
  - confirmar
  - abortar
- Estados por los que puede pasar una transacción:



Planificaciones y concurrencia

- Una **planificación** p de n transacciones T1,..,Tn es un ordenamiento global de las operaciones de cada una de las transacciones que respete el orden interno de las operaciones dentro de cada transacción.



Planificaciones y concurrencia

- Las planificaciones se clasifican en:
  - Planificación en serie (serializada):

    Si para cada transacción T que participa en la planificación, las operaciones de T se ejecutan consecutivamente.
  - Planificación no en serie (no serializada)
    - Planificación serializable si es equivalente a alguna planificación en serie del mismo conjunto de transacciones.
    - Planificación no serializable si no es equivalente a ninguna planificación en serie.

    Pero, ¿qué significa equivalente? Dos posibles definiciones

Equivalencia por conflictos

Equivalencia por vistas

Planificaciones y concurrencia

- Planificaciones equivalentes por conflictos:

  aquellas en que el orden de dos operaciones cualesquiera en conflicto es el mismo en ambos planes.

- ¿Cuándo dos operaciones de un plan están en conflicto?

  Dos acciones A_i y A_j NO están en conflicto si toda ejecución de A_i seguida de A_j conduce al mismo resultado que si ejecutáramos A_j seguido de A_j.

  Dos acciones ESTÁN EN CONFLICTO si:

  - pertenecen a diferentes transacciones
  - tienen acceso al mismo elemento X
  - al menos una de ellas escribe en X

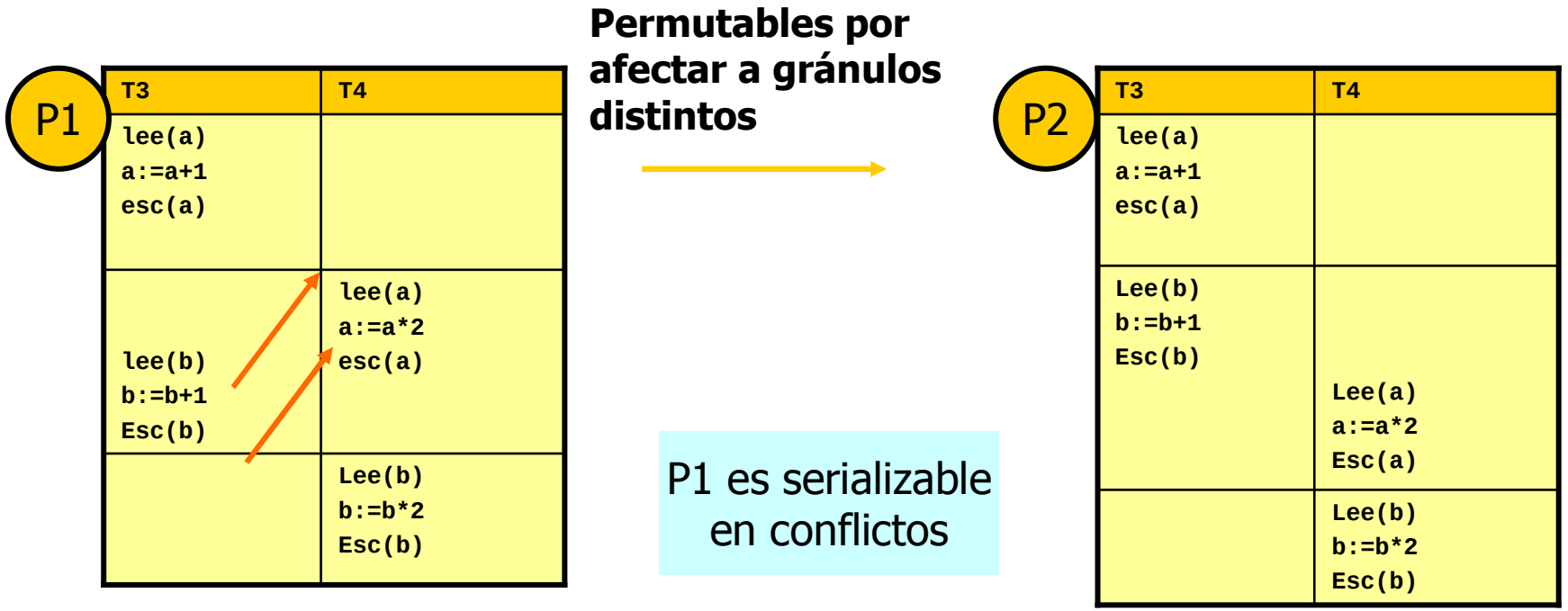
Acciones sobre gránulos distintos no son conflictivas.

Acciones de lectura y escritura (sobre el mismo gránulo) son conflictivas

Acciones de escritura (sobre el mismo gránulo) son conflictivas.

Planificaciones y concurrencia

- Una **planificación es serializable por conflictos** si es equivalente por conflictos a alguna planificación serializada.
- Es decir, si se puede transformar en una planificación en serie mediante la permutación de acciones no conflictivas (permutables) entre sí.



Planificaciones y concurrencia

- Interés: determinar un método que permita detectar planes serializables por conflictos.

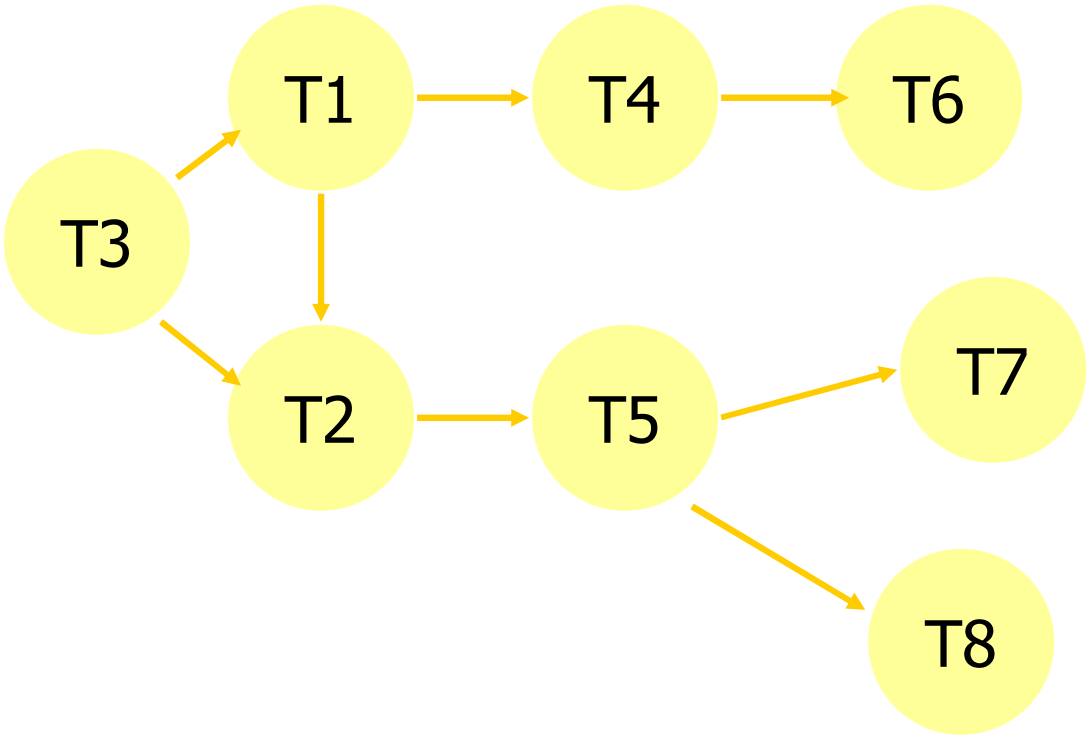
- Relación de precedencia:

  Se dice que una transacción Ti precede a otra Tj en una planificación P, si existen dos acciones ai y aj conflictivas (no permutables) de forma que ai se ejecuta por Ti antes de aj por Tj.

- Grafo de precedencia: es un grafo orientado cuyo conjunto de vértices es el conjunto de transacciones y existe un arco entre Ti y Tj si Ti precede a Tj.

Planificaciones y concurrencia

- Ejemplo de grafo de precedencia



Grafo de precedencia

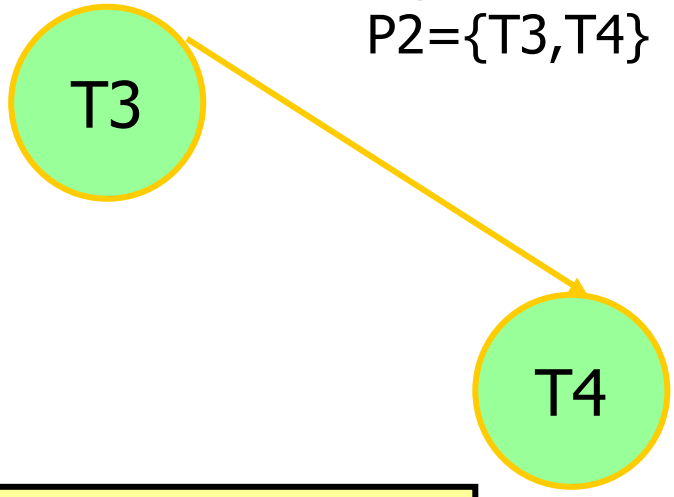
Planificaciones y concurrencia

- Ejemplo planificación P1

P1

T3	T4
lee(a) a:=a+1 esc(a)	
Lee(b) b:=b+1 Esc(b)	Lee(a) a:=a*2 Esc(a)
	Lee(b) b:=b*2 Esc(b)

El plan en serie
Equivalente es
P2={T3,T4}



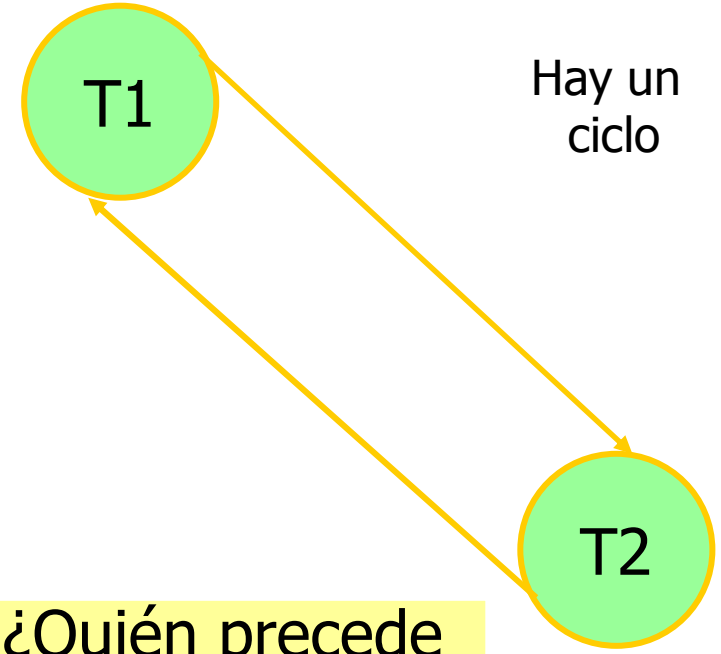
T3 precede a T4

Planificaciones y concurrencia

- Ejemplo 2:

P3

T1	T2
Lee(x)	
X:=x-N	
	lee(x)
	X:=x+M
escribe(x)	
lee(y)	
	escribe(x)
y:=y+N	
escribe(y)	



Hay un ciclo

¿Quién precede a quién?

Planificaciones y concurrencia

- Proposición: la condición para que una planificación sea serializable (por conflictos) es que el grafo de precedencias asociado no tenga ciclos.

- Ejemplo sobre tres transacciones:

T1
leer(x)
esc(x)
lee(y)
esc(y)

T2
lee(z)
lee(y)
esc(y)
lee(x)
esc(x)

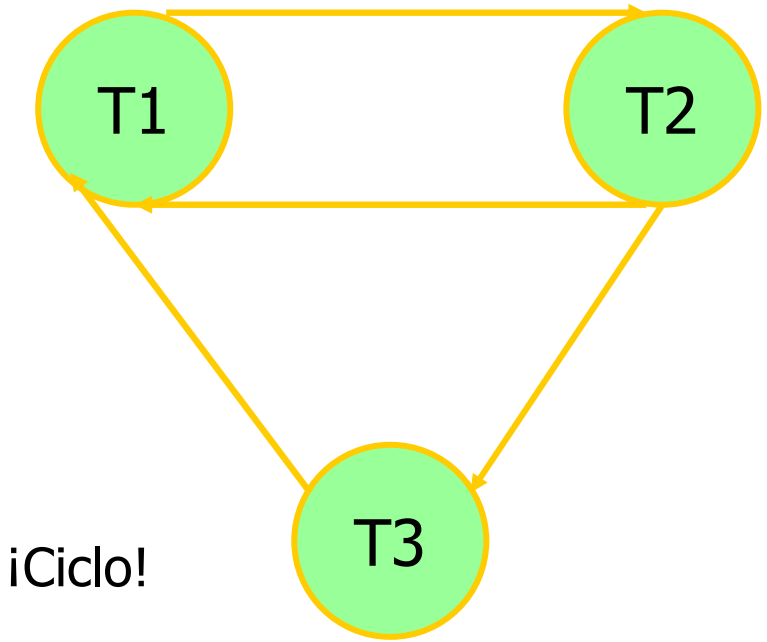
T3
lee(y)
lee(z)
esc(y)
esc(z)

Planificaciones y concurrencia

- Ejemplo planificación P4

P4

T1	T2	T3
	Lee(z) Lee(y) Esc(y)	
Lee(x) Esc(x)		Lee(y) Lee(z) Esc(y) Esc(z)
Lee(y) Esc(y)	Lee(x) Esc(x)	



¡Ciclo!

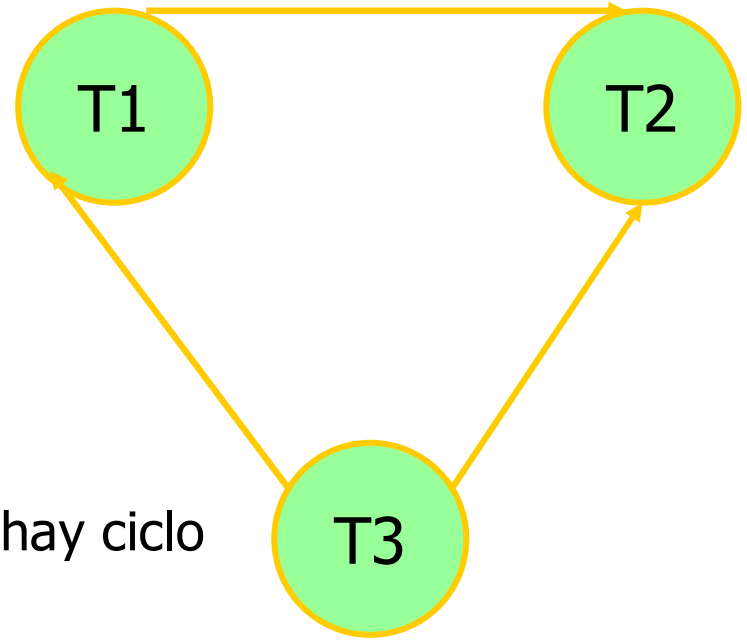
¡No existe plan en serie equivalente!

Planificaciones y concurrencia

- Ejemplo planificación P5:

P5

T1	T2	T3
Lee(x) Esc(x)		Lee(y) Lee(z)
	Lee(z)	Esc(y) Esc(z)
Lee(y) Esc(y)	Lee(y) Esc(y) Lee(x) Esc(x)	



No hay ciclo

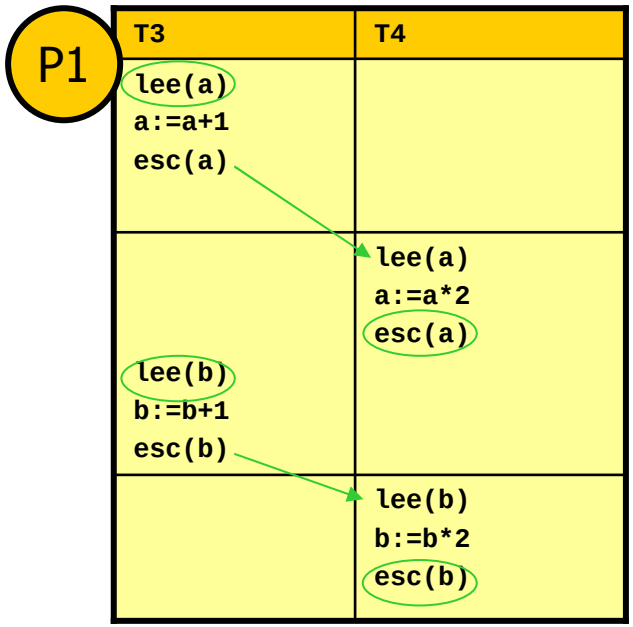
Es serializable, existe un plan en serie equivalente {T3,T1,T2}

Planificaciones y concurrencia

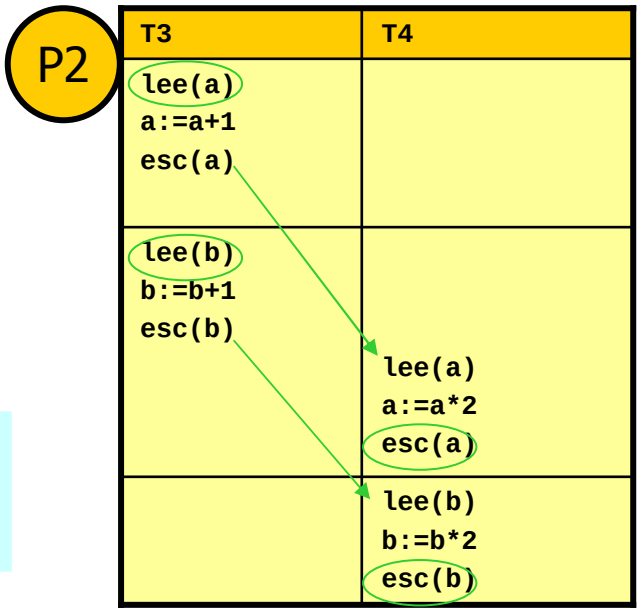
- Dos planificaciones también pueden ser **equivalentes por vistas**.
- Dos planificaciones P1 y P2 son equivalentes por vistas si cumplen las tres condiciones siguientes:
  1. si contienen el mismo conjunto de transacciones.
  2. si en P1 Ti lee un gránulo cuyo valor fue escrito por Tj (o si es el valor original del gránulo antes de empezar el plan), en P2 ocurre lo mismo.
  3. si en P1, Tk es la última operación que escribe un determinado gránulo, en P2 pasa lo mismo.
- Una **planificación es serializable por vistas** si es equivalente por vistas a una planificación en serie.

Planificaciones y concurrencia

- Equivalencia y seriabilidad por vistas. Ver el orden de:
  - escrituras que preceden a lecturas del mismo gránulo
  - primeras lecturas / últimas escrituras



P1 y P2 son equivalentes por vistas



Planificaciones y concurrencia

- Equivalencia y seriabilidad por vistas.

P3

T1	T2
lee(x)	
x:=x-N	
	lee(x)
	x:=x+M
escribe(x)	
lee(y)	
	escribe(x)
y:=y+N	
escribe(y)	

P3 y P7 NO son equivalentes por vistas

P7

T1	T2
lee(x)	
x:=x-N	
escribe(x)	
lee(y)	
y:=y+N	
escribe(y)	
	lee(x)
	x:=x+M
	escribe(x)

Planificaciones y concurrencia

- Equivalencia y seriabilidad por vistas. Ver el orden de:
  - escrituras que preceden a lecturas del mismo gránulo
  - primeras lecturas / últimas escrituras

P5

T1	T2	T3
Lee(x) Esc(x)		Lee(y) Lee(z)
	Lee(z)	Esc(y) Esc(z)
Lee(y) Esc(y)	Lee(y) Esc(y) Lee(x) Esc(x)	

P5 y P6 ¿son equivalentes por vistas?

P6

T1	T2	T3
		Lee(y) Lee(z) Esc(y) Esc(z)
Lee(x) Esc(x) Lee(y) Esc(y)		
	Lee(z) Lee(y) Esc(y) Lee(x) Esc(x)	

Planificaciones y concurrencia

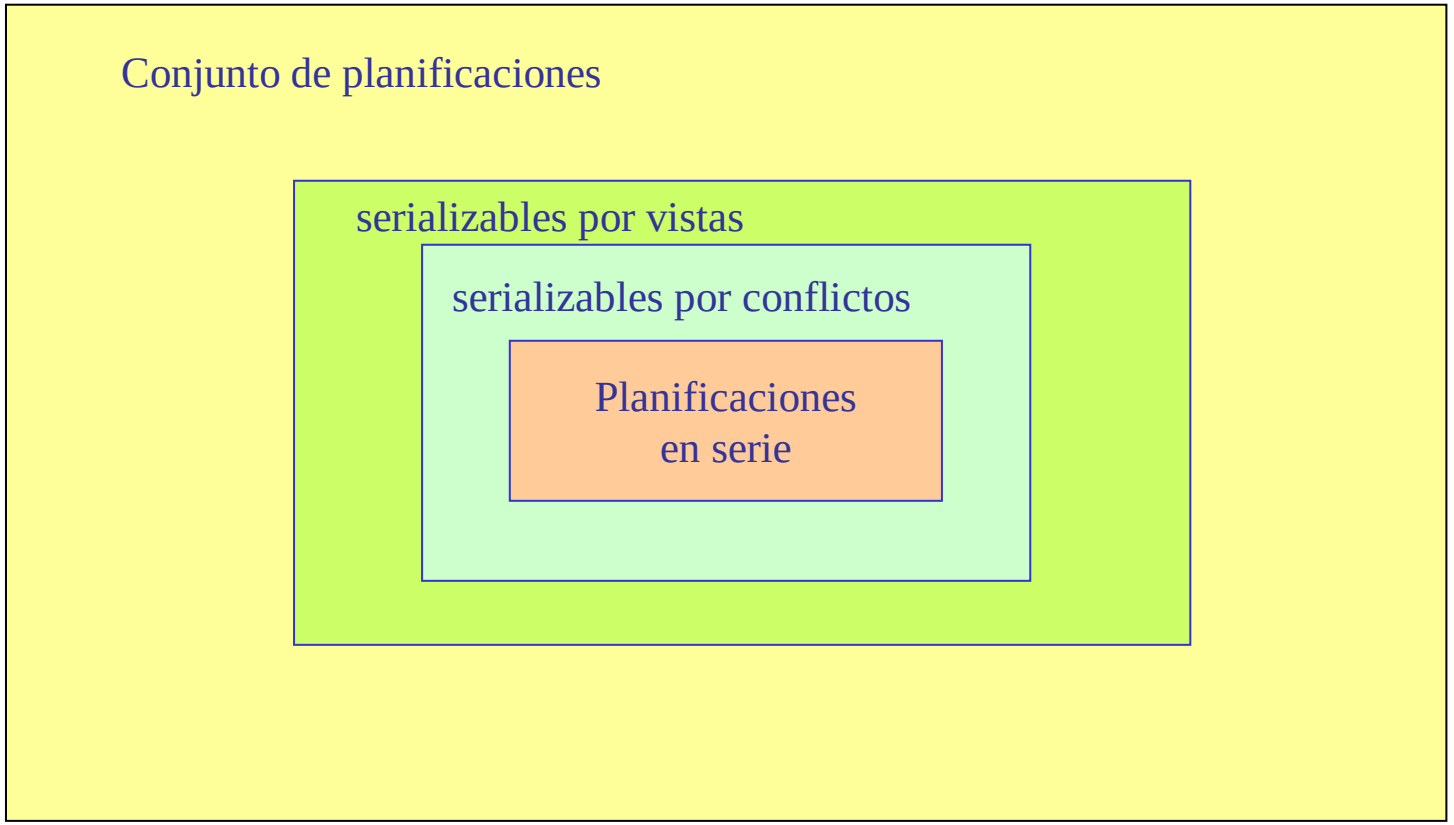
- La seriabilidad por conflictos y por vistas es similar si se cumple la **suposición de escritura restringida**,
  - si cualquier operación de escritura $w_i(X)$ en T_i va precedida de una operación de lectura $r_i(X)$ en T_i,
  - y si el valor escrito por $w_i(X)$ en T_i depende (de la manera que sea) del valor leído por $r_i(X)$ en T_i
- Las definiciones de seriabilidad por conflictos y por vistas son equivalentes si no se efectúa ninguna escritura ciega
  - ninguna escritura que no dependa de una lectura anterior.

Condición más débil que equivalencia por conflictos

```
P:T1:lee(x),T2:esc(x),T1:esc(x),T3:esc(x)
```

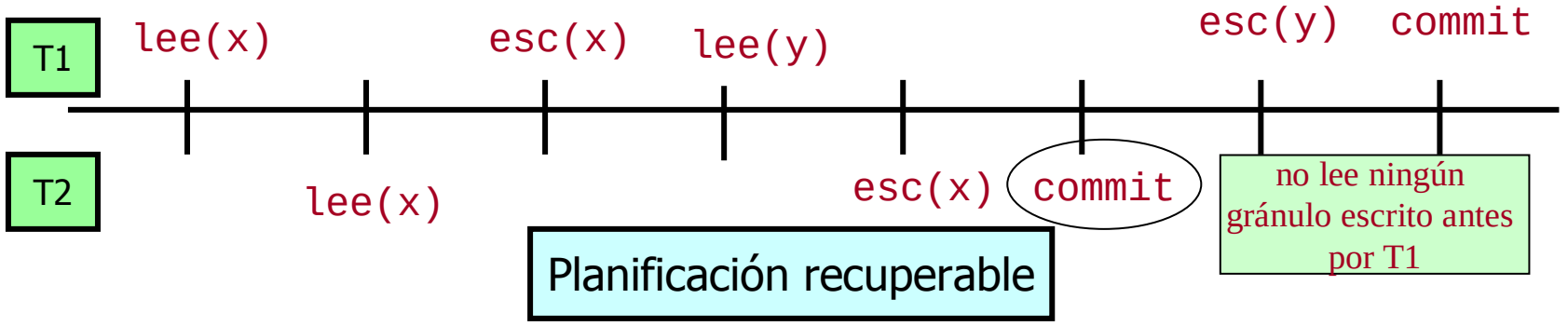
Escritura ciega

Diagrama del conjunto de planificaciones

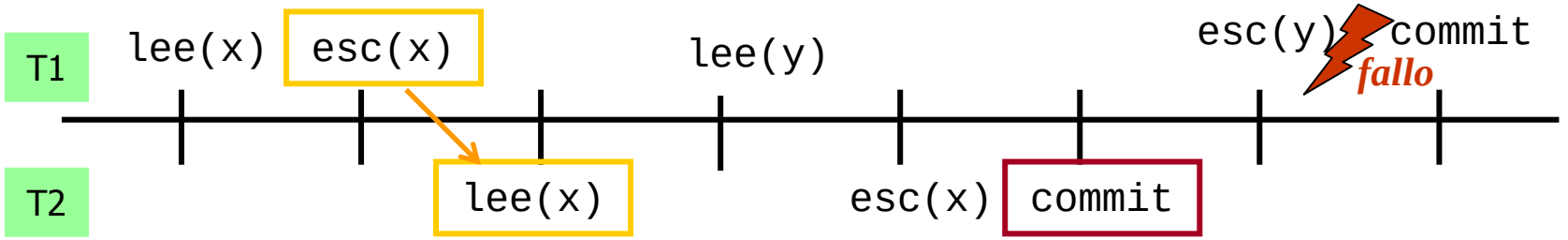


Planificaciones y recuperabilidad

- Clasificación de planificaciones:
  - **Recuperables:** una vez que una T ha confirmado nunca será necesario deshacerla.
  - **No recuperables:** no satisfacen la condición anterior (no se deben permitir).
- Una planificación P es recuperable si ninguna transacción T de P se confirma antes de que se hayan confirmado todas las transacciones T' que han escrito un elemento que T lee posteriormente

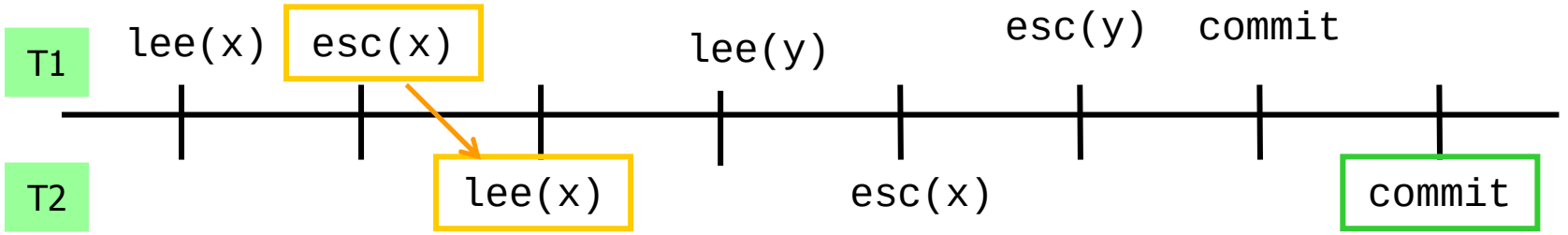


Planificaciones y recuperabilidad



Planificación no recuperable

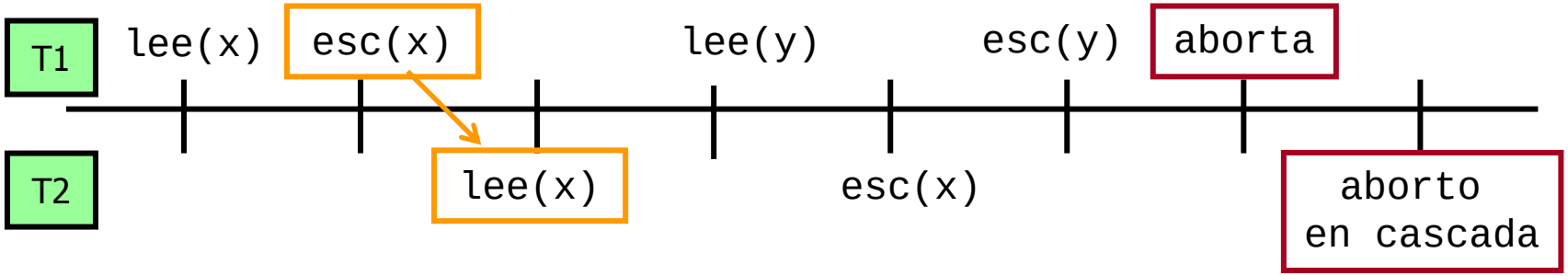
- La anterior es no recuperable porque T2 se confirma antes de que lo haga T1; si T1 fallara se dificultaría la recuperación
- Se transformaría en una planificación recuperable si posponemos la confirmación de T2 hasta después de la de T1



Planificación recuperable

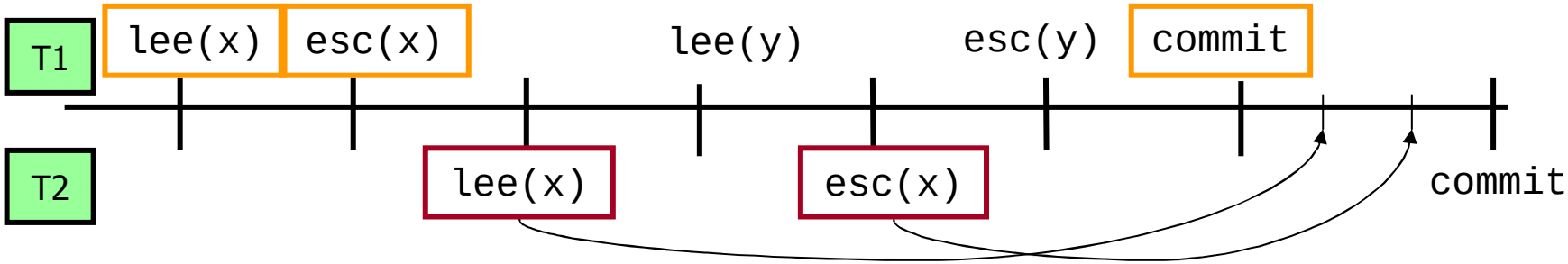
Planificaciones y recuperabilidad

- Puede ocurrir el fenómeno de **restauración en cascada** (**aborto en cascada**):
  una transacción no confirmada debe deshacerse porque leyó un elemento de una transacción fallida.



Planificaciones y recuperabilidad

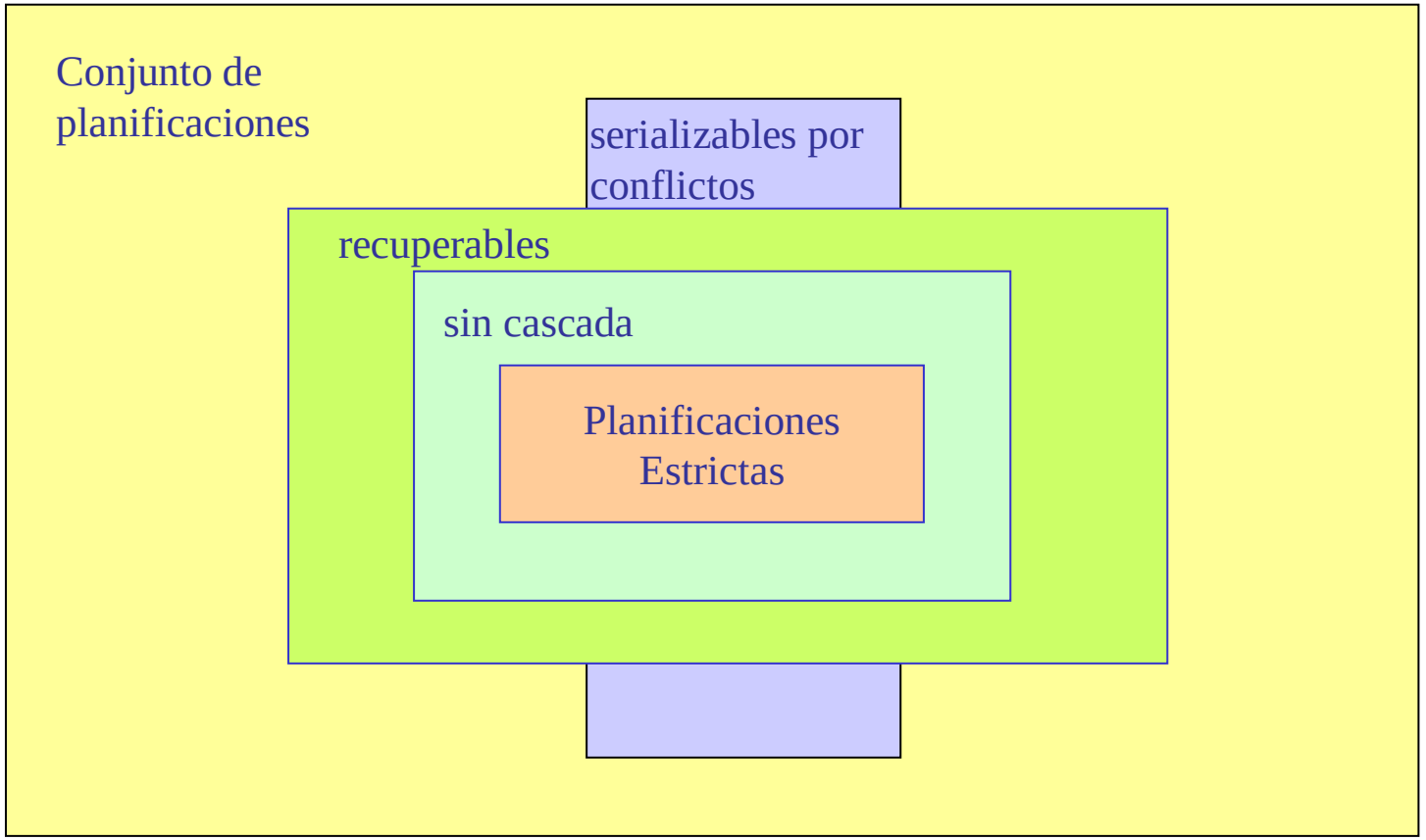
- El aborto en cascada puede consumir mucho tiempo por lo que se intenta evitar:
  - **Planificación sin abortos en cascada**: si toda transacción de la planificación solo lee elementos escritos por transacciones confirmadas. *Por ejemplo:*



  - **Planificaciones estrictas**: las transacciones no pueden leer ni escribir de un gránulo x hasta que se confirme la última transacción que escribió x (fáciles de recuperar).

Planificaciones y recuperabilidad

Diagrama del conjunto de planificaciones



Soporte de transacciones en SQL

Definición de transacciones SQL

- Inicio de transacción: es implícito en SQL
- Final de transacción: en SQL
  - `COMMIT`
  - `ROLLBACK`
- Características de una transacción SQL
  - mediante SET TRANSACTION
  - se puede fijar
    - Modo de acceso
    - Tamaño del área de diagnóstico
    - Nivel de aislamiento

Soporte de transacciones en SQL

- Características de una transacción especificables en SQL:

Modo de acceso:

```
READ ONLY
READ WRITE
```

Tamaño del área de diagnóstico:

```
DIAGNOSTICS SIZE <n>
```

Nivel de aislamiento:

```
ISOLATION LEVEL <isol>
<isol>::=READ UNCOMMITTED |READ COMMITTED|
REPEATABLE READ|SERIALIZABLE
```

Soporte de transacciones en SQL

- Violaciones posibles con aislamiento < SERIALIZABLE:
  - Lectura sucia: una transacción T1 puede leer la actualización de T2 que todavía no ha confirmado. Si T2 aborta, T1 habría leído un dato incorrecto.
  - Lectura no reproducible: Si una transacción lee dos veces un mismo dato y en medio una transacción lo modifica, verá valores diferentes para el dato.
  - Fantasmas: una transacción T1 puede leer un conjunto de filas (que cumplan una condición). Si una transacción T2 inserta una fila que también cumple la condición y T1 se repite verá un fantasma, una fila que previamente no existía.

Soporte de transacciones en SQL

Violaciones posibles según el nivel de aislamiento

	Lectura sucia	**Lectura no reproducible**	**Fantasmas**
Lectura no Confirmadas	**Sí**	**Sí**	**Sí**
Lectura Confirmadas	**No**	**Sí**	**Sí**
Lectura repetible	**No**	**No**	**Sí**
Serializable	**No**	**No**	**No**

Soporte de transacciones en SQL

- Ejemplo de transacción en SQL:

```
EXEC SQL WHENEVER SQLERROR GOTO UNDO;
EXEC SQL SET TRANSACTION
   READ WRITE
   ISOLATION LEVEL SERIALIZABLE;
EXEC SQL INSERT INTO EMPLEADO (ENOMBRE,DNO,SALARIO)
   VALUES ('Miguel López','2',35000);
EXEC SQL UPDATE EMPLEDO SET SALARIO=SALARIO*1,1 WHERE
   DNO=2;
EXEC SQL COMMIT;
FINAL:
UNDO:
   EXEC SQL ROLLBACK
```

Bibliografia utilizada

- Ramez A. Elmasri, Shamkant B. Navathe, “Fundamentos de Sistemas de Bases de datos”. Addison Wesley, tercera edición (cap.19)

- G. Weikum, G. Vossen, “Transactional information systems”, Morgan Kaufmann, 2002.

- G. Gardarin, P. Valduriez, “Relational Databases and Knowledge Bases”. Addison Wesley, 1989.

Ejercicios propuestos

- Pon un ejemplo de transacción y justifica que cumple cada una de las características deseables. Utiliza otra transacción, mezcla las acciones de las dos transacciones en el tiempo y mira lo que ocurre con su ejecución. ¿Se manifiesta alguno de los problemas estudiados?

- ¿Qué es una planificación? Indique las diferentes formas en que diferentes planificaciones pueden ser equivalentes.

- ¿Qué es una planificación en serie? ¿Por qué se considera correcta una planificación en serie? ¿Por qué se considera correcta una planificación serializable?

- Discute las diferencias entre serializable por conflictos y por vistas.

Ejercicios propuestos

- ¿Qué diferencia hay entre la suposición de escritura restringida y no restringida? ¿Cuál es más realista?

- Responder a las siguientes preguntas:
  ¿Cuántas planificaciones pueden generarse a partir de 2 transacciones con 3 acciones cada una de ellas?
  ¿Cuántas de ellas son en serie?
  ¿Cuántas son serializables en caso de que ninguna de las transacciones actúe sobre un gránulo usado por otra?
  ¿Cuántas planificaciones son serializables si ninguna transacción realiza escrituras?

Ejercicios propuestos

- Considera las siguientes planificaciones y de qué tipo de planificación es desde el punto de vista del control de concurrencia y de la recuperación:
  1. {T1:lee(x),T2:lee(x),T1:esc(x),T2:esc(x)}
  2. {T1:esc(x), T2:lee(y), T1:lee(y),T2:lee(x)}
  3. {T1:lee(x),T2:lee(y),T3:esc(x),T2:lee(x),T1:lee(y)}
  4. {T1:lee(x),T2:esc(x),T1:esc(x),T2:aborta, T1:confirma}
  5. {T1:esc(x),T2:lee(x),T1:esc(x),T2:confirma, T1:confirma}
- Dada la siguiente planificación indica si es serializable y de que tipo.

  {T1:lee(x),T2:esc(x),T1:esc(x),T3:esc(x)}
- Dado el diagrama de inclusión de tipos de planificaciones da un ejemplo para cada subconjunto de planificaciones.

Ejercicios propuestos

- ¿Cuáles de los siguientes planes son serializables por conflictos? Dibuja los grafos de precedencia y para cada plan serializable indica el plan en serie equivalente
  1. { T1:lee(x), T3:lee(x), T1:esc(x), T2:lee(x), T3:esc(x) }
  2. { T1:lee(x), T3:lee(x), T3:esc(x), T1:esc(x), T2:lee(x) }
  3. { T3:lee(x), T2:lee(x), T3:esc(x), T1:lee(x), T1:esc(x) }
  4. { T3:lee(x), T2:lee(x), T1:lee(x), T3:esc(x), T1:esc(x) }

- Pon un ejemplo de transacción en SQL seleccionando el nivel de aislamiento e indicando los problemas que pueden surgir con el nivel seleccionado.