# Concurrencia

**Bibliografía: Introducción a los Sistemas de Bases de Datos**
**Date, C.J.**

# Concurrencia

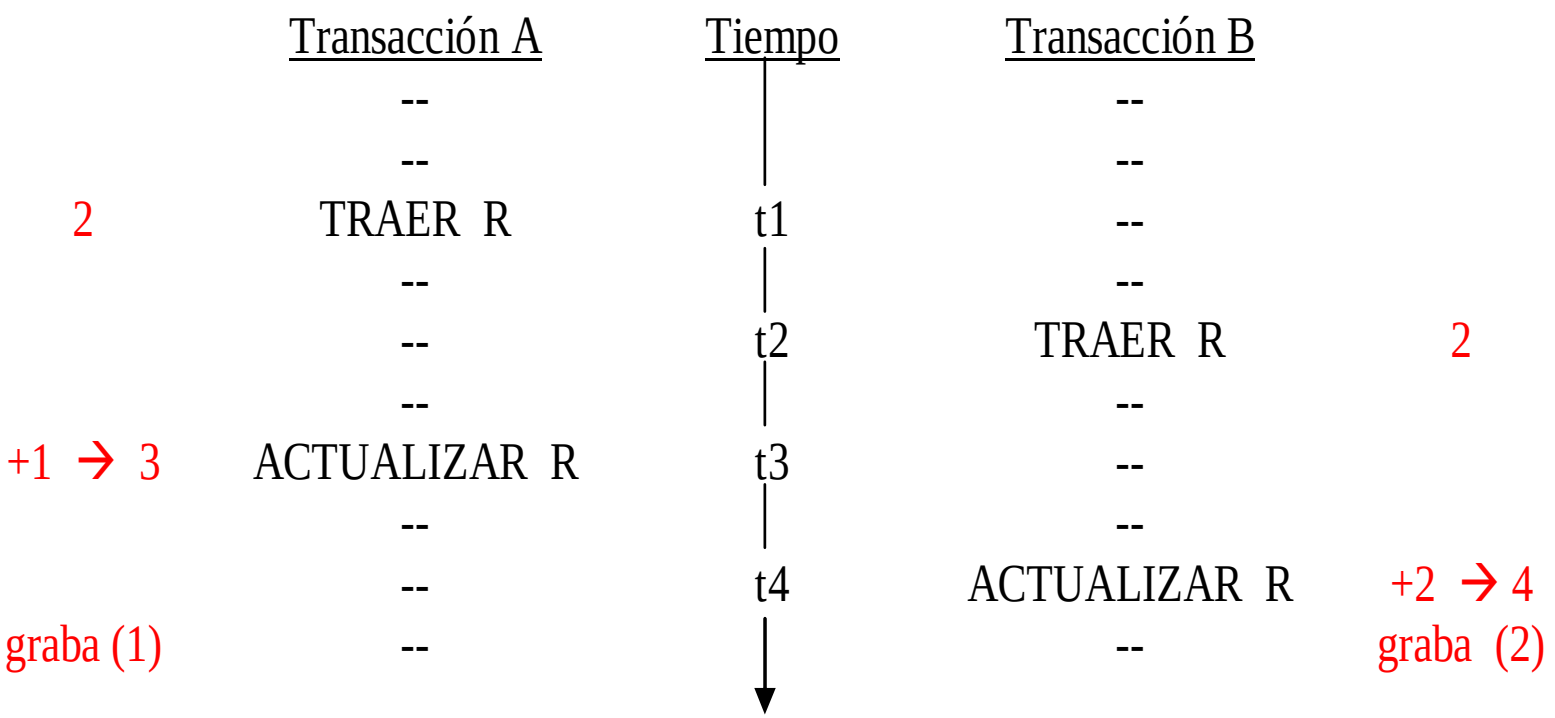
- La mayor parte de los DBMS son **sistemas para múltiples usuarios**
- Se permite a **cualquier cantidad de transacciones** tener **acceso a la misma BD al mismo tiempo**
- Se necesita un **mecanismo de control de concurrencia** para asegurar que ninguna transacción interfiera con las demás

  (➔ bloqueo)

# Tres problemas de concurrencia

- Los **problemas** que se pueden presentar en los cuales una **transacción correcta** puede producir un **resultado incorrecto** debido a una **interferencia de otra transacción son:**
  - El problema de la **modificación perdida**
  - El problema de la **dependencia no comprometida**
  - El problema del **análisis inconsistente**

# El problema de la modificación perdida

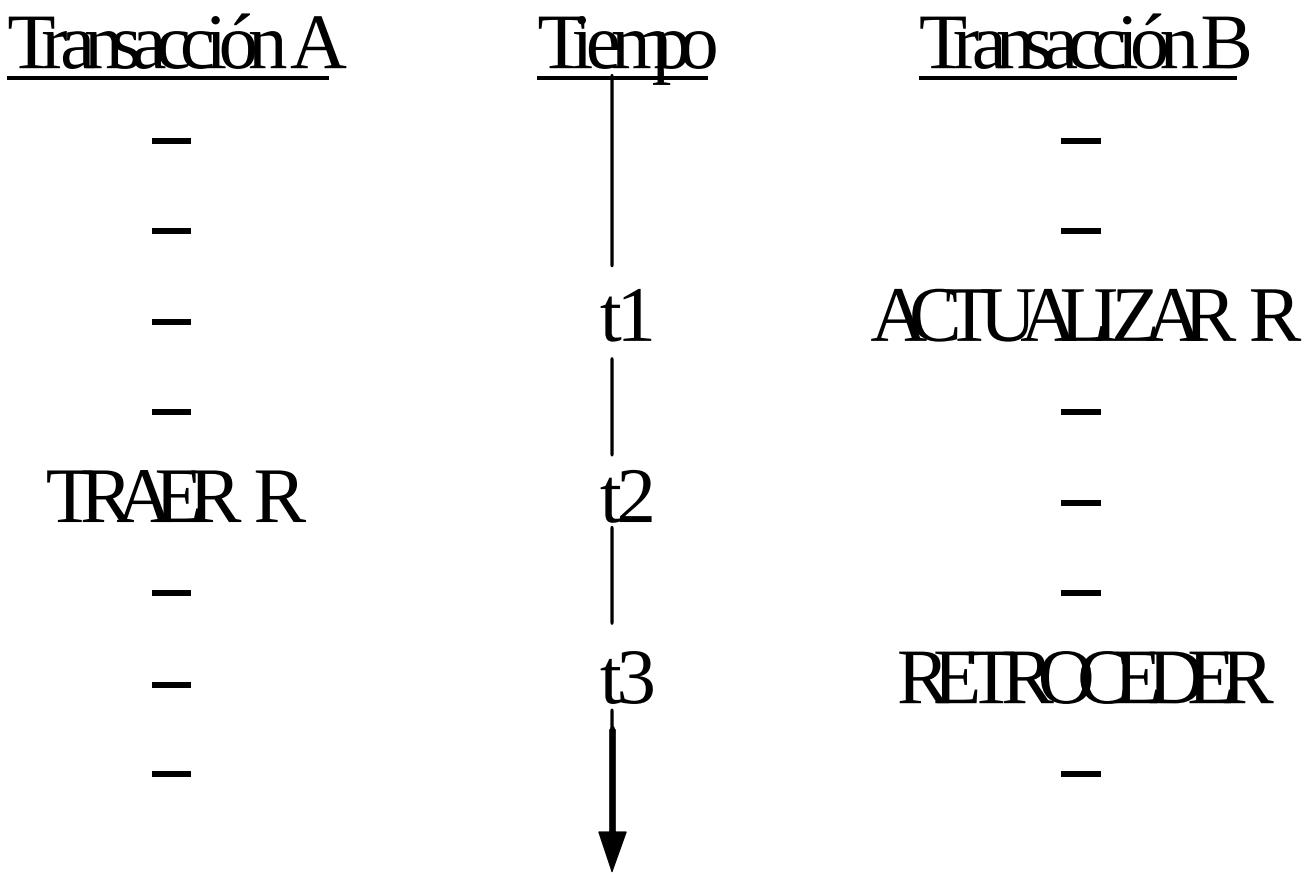


- La transacción A pierde una modificación en t4
- La transacción **A lee un registro** R en el momento **t1**
- La transacción **B lee ese mismo registro** R en el momento **t2**
- La transacción **A actualiza** ese registro **según los valores observados en t1 en t3**
- La transacción **B actualiza** el **mismo registro** (**según valores** vistos en **t2**, los cuales son también los observados en t1) en el momento **t4**

La **modificación de** la transacción **A**
**se pierde** en **t4** porque la
**B graba** su registro modificado
**encima del registro**
**modificado por A** sin verlo siquiera

# El problema de la dependencia no comprometida

- Se presenta cuando **se permite** a una transacción **leer (o modificar)** un registro que ha sido **actualizado por otra transacción** y esta última todavía **no** lo ha **comprometido**
  - Existe la posibilidad de que **nunca se comprometa y se anule**
  - Y la primera transacción habrá visto **datos** que **ya no existen** o nunca existieron


Transacción A
Tiempo
Transacción B
t1
ACTUALIZAR R
TRAER R
t2
t3
RETROCEDER

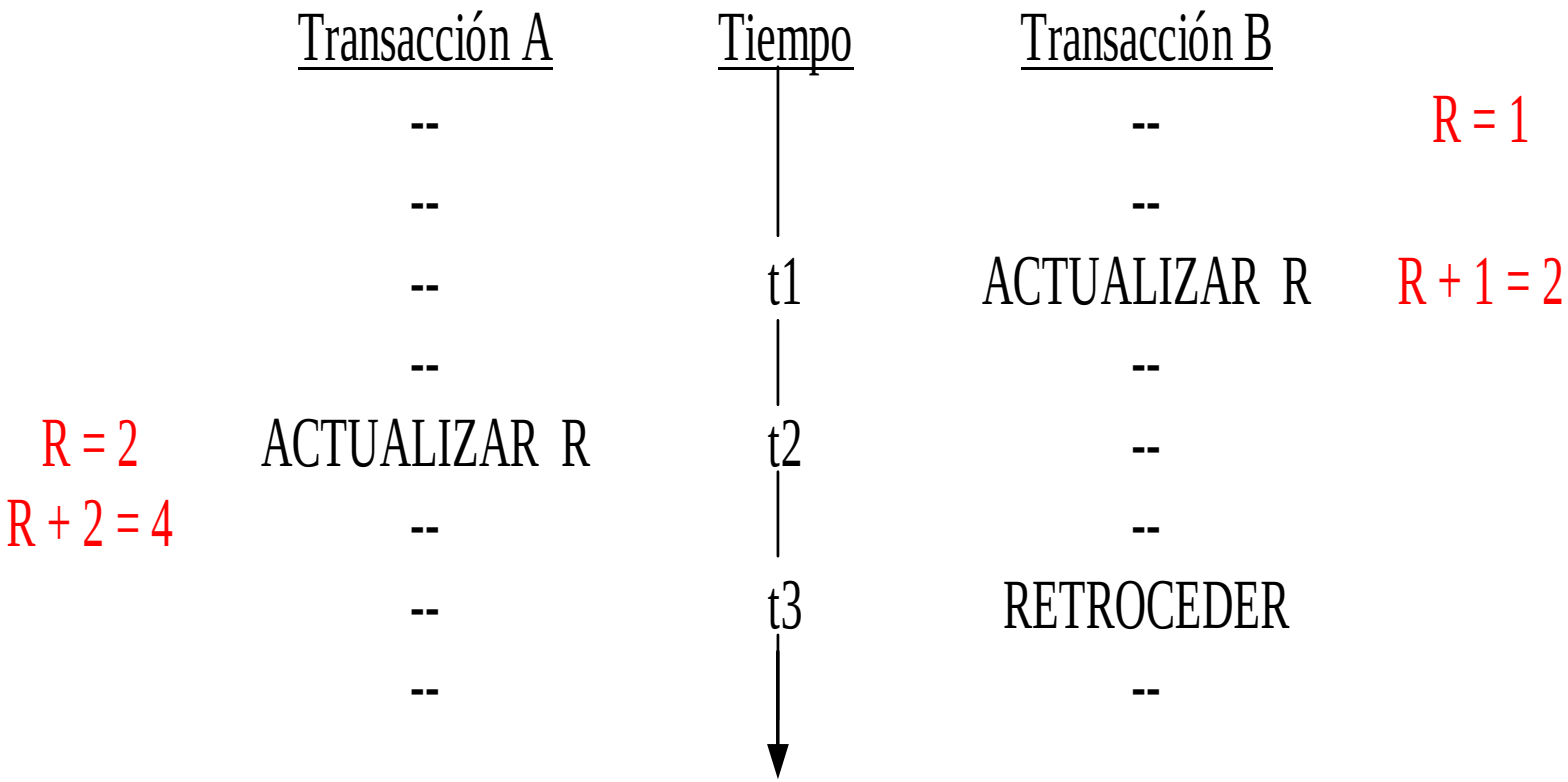
La transacción A se vuelve dependiente de una modificación no comprometida en t2

- **A** observa una **modificación no comprometida** en **t2**
- **A** opera sobre una **suposición falsa**:
  - **R** tiene el **valor** visto en **t2**,
  - mientras que **su valor** es el visto en **t1**

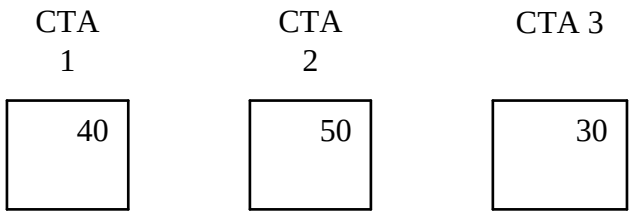
➔ **A** podría producir una **salida incorrecta**

- El retroceso de B podría ser por una caída del sistema y
- A podría haberse completado antes del ROLLBACK de B

**Observación**: **Peor aún** si **A actualiza una modificación no comprometida en t2** y **pierde esa actualización en t3:**



# El problema del análisis inconsistente




| Transacción A | Tiempo | Transacción B |
|---|---|---|
| -- | | -- |
| -- | | -- |
| TRAER CTA 1 (40): | t1 | -- |
| suma = 40 | | -- |
| | | -- |
| TRAER CTA 2 (50): | t2 | -- |
| suma = 90 | | -- |
| -- | | -- |
| -- | t3 | TRAER CTA 3 (30): |
| -- | | -- |
| -- | t4 | ACTUALIZAR CTA 3: |
| -- | | 30 → 20 |
| -- | | -- |
| -- | t5 | TRAER CTA 1 (40): |
| -- | | -- |
| -- | t6 | ACTUALIZAR CTA 1: |
| -- | | 40 → 50 |
| -- | | -- |
| -- | t7 | COMPROMETER |
| -- | | |
| TRAER CTA 3 (**20**): | t8 | |
| suma = **110**, **no** 120 | | |

Compromete todo antes que A vea CTA 3. Por lo tanto, A no depende de una modificación no comprometida

- La transacción A realiza un análisis inconsistente.
- Dos **transacciones** A y B operan sobre **registros de cuentas**:
  - **A suma saldos** de las cuentas y
  - **B transfiere** una cantidad de 10 de la cuenta 3 a la 1
- El **resultado** producido por **A** igual a 110 es **incorrecto**.
- Si lo grabara la base quedaría en un **estado inconsistente**
- **A** ha visto la **base** en un **estado inconsistente** ➔ hizo un **análisis inconsistente**

**Nota**:

A diferencia del ejemplo anterior, en este caso **A no depende** en absoluto **de una modificación no comprometida**, pues **B compromete todas sus modificaciones antes** que A vea CTA3.

# Bloqueo

- Cuando una **transacción** requiere la **seguridad** de que algún **objeto no cambie** de manera no predecible sin que ella se de cuenta, **adquiere un bloqueo** sobre ese objeto.

- Se **bloquea** el **acceso** a **otras transacciones** al objeto y **evita** que lo **modifiquen**

# Bloqueo

Hay **dos tipos de bloqueo**:

- X: **exclusivos**
- S: **compartidos** (Share)

# Bloqueo exclusivo

Si una transacción **A** tiene un **bloqueo exclusivo** X **sobre** el registro **R**,

- una solicitud de parte de otra transacción B de **cualquier tipo de bloqueo** sobre R hará que **B** entre en un **estado de espera**, **hasta que A libere el bloqueo**

# Bloqueo compartido

Si una transacción **A** tiene un **bloqueo compartido** S **sobre** el registro **R**:

- una solicitud de otra transacción B de **bloqueo X** sobre R hará que **B** entre en un **estado de espera**, hasta que **A libere el bloqueo**

- una solicitud de otra transacción B de **bloqueo S** sobre R será **concedida**. Es decir B tendrá también un bloqueo S sobre R.

# Matriz de compatibilidad de tipos de bloqueo

| B / A | X | S | -- |
|---|---|---|---|
| X | N | N | S |
| S | N | S | S |
| - | S | S | S |

X: eXclusivo
S: compartido (Share)

N: CONFLICTO

S: COMPATIBILIDAD

N ➔ La solicitud **no se concede** y B entra en **estado de espera**

S ➔ La solicitud **se concede**

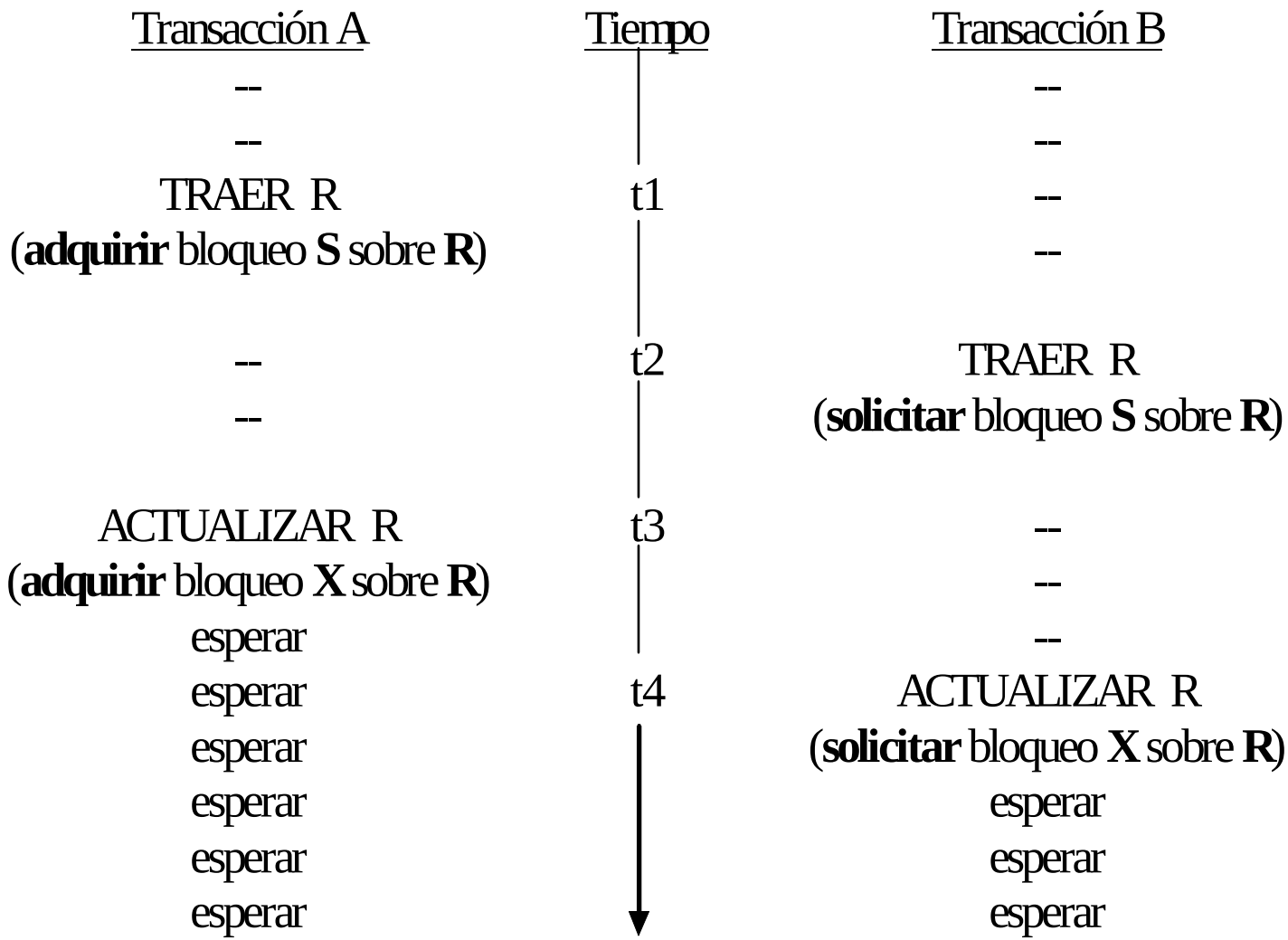
- Las solicitudes de bloqueos sobre registros por parte de las transacciones son **implícitas** en **condiciones normales**.
  - Cuando una transacción **lee** un registro, adquiere **automáticamente** un **bloqueo** del tipo **S** sobre él
  - Cuando una transacción **actualiza** un registro, adquiere **automáticamente** un **bloqueo** del tipo **X** sobre él

- Los **bloqueos X** se mantiene **hasta** el **siguiente punto de sincronización**

- El bloqueo resuelve los problemas anteriores

# El problema de la modificación perdida

- La instrucción **ACTUALIZAR** de **A** en **t3 no se acepta** porque es una **solicitud implícita de bloqueo X** sobre R y está en **conflicto** con el **S de B** y entra en espera
- Análogamente **B** entra en **espera en t4**
- **Resuelve la pérdida de información.**
- Pero **aparece nuevo problema**: **Bloqueo Mutuo**


Transacción A
Tiempo
Transacción B
--
--
TRAER R
(adquirir bloqueo S sobre R)
t1
--
--
--
--
t2
TRAER R
(solicitar bloqueo S sobre R)
ACTUALIZAR R
(adquirir bloqueo X sobre R)
esperar
esperar
esperar
esperar
esperar
t3
t4
--
--
--
ACTUALIZAR R
(solicitar bloqueo X sobre R)
esperar
esperar
esperar

- No se pierde ninguna modificación, pero hay un **bloqueo mutuo** en t4

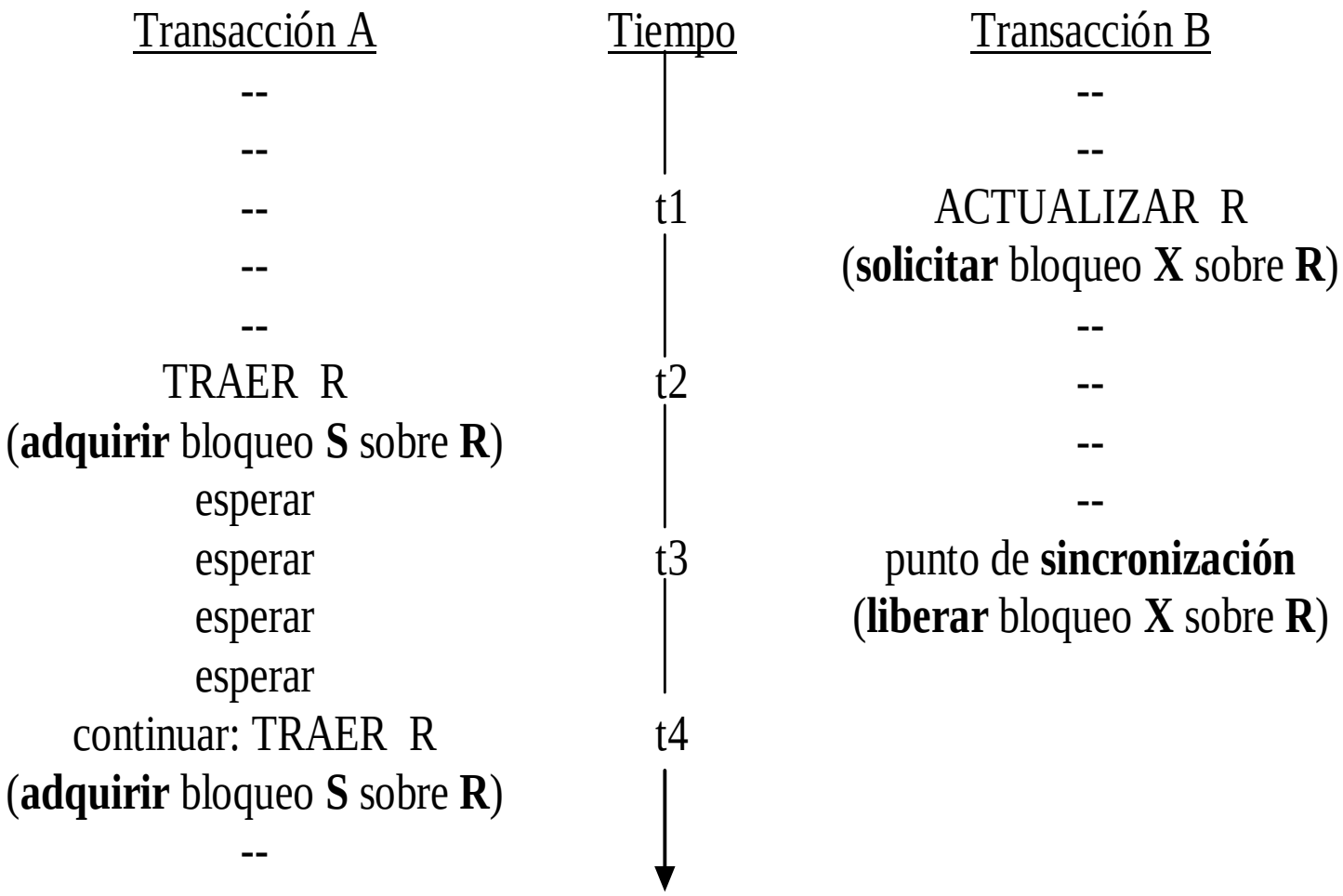
- **El problema de la dependencia no comprometida:**

  – La **operación de A** (leer o actualizar) en **t2 no se acepta** porque es una **solicitud** de bloqueo **S** sobre R y está en **conflicto con** el **bloqueo X** que tiene **B**

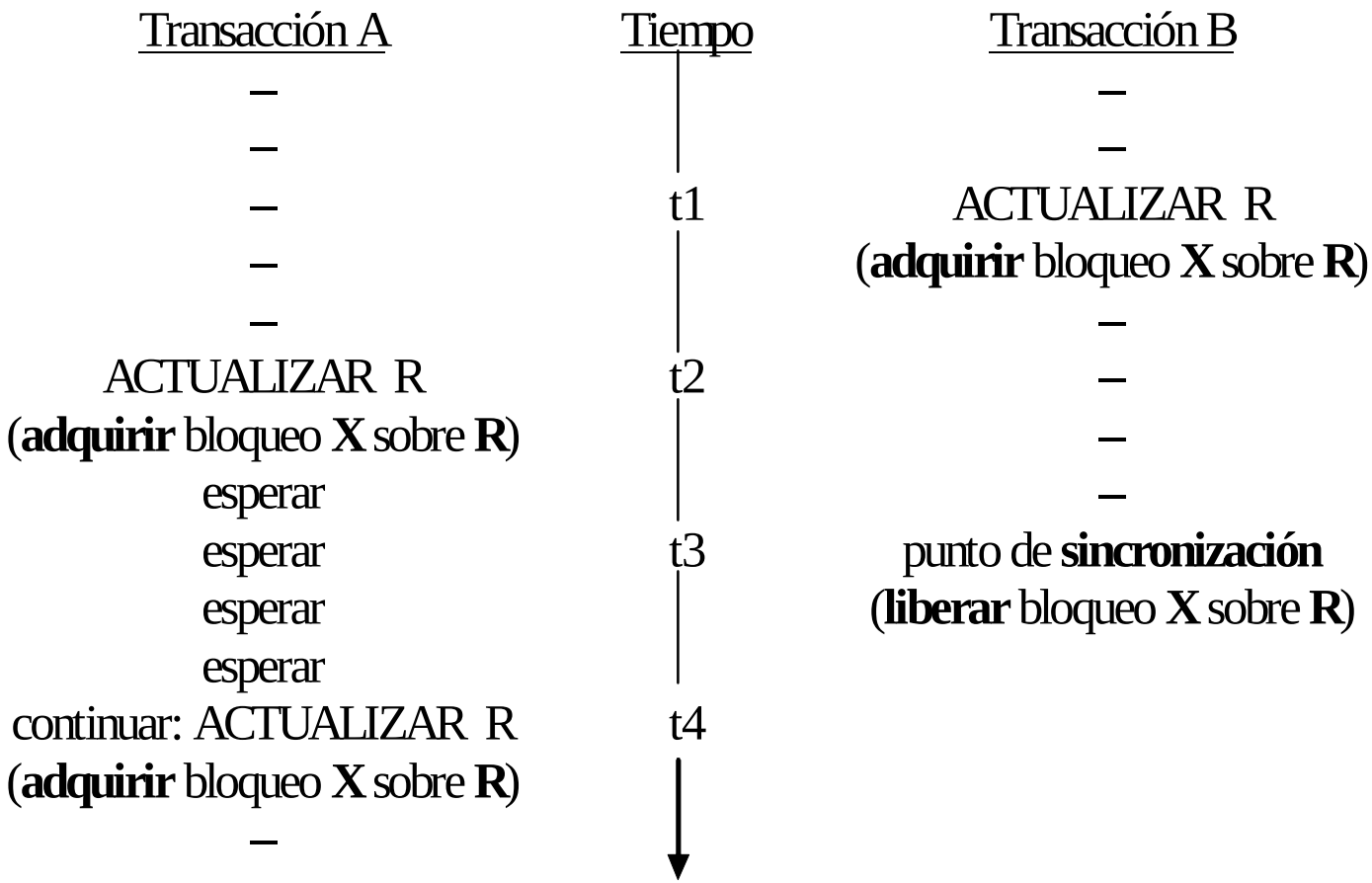
  – **A** entra en **espera hasta** que **B** llegue a un **punto de sincronización** (ROLLBACK o COMMIT)

  – Cuando **libere** el bloqueo, **A** podrá **continuar** sobre un valor comprometido de R según cómo haya terminado B

# Se evita que la transacción A **lea** una modificación no comprometida en t2



# Se evita que la transacción A **actualice** una modificación no comprometida



# El problema del análisis inconsistente

- La operación **MODIFICAR** de B en **t6 no se acepta** porque es **implícitamente** un **bloqueo X** sobre CTA 1 y está en **conflicto** con el **bloqueo S** que ya tiene A
- **B** entra en **espera**.
- La operación **TRAER** de A en **t7 no se acepta** porque es **bloqueo S** sobre CTA3 y está en **conflicto** con el **bloqueo X** que ya tiene
- **A** entra en **espera**.

➔ ocurre un **bloqueo mutuo**.

(*) ya tiene un bloqueo
**S** → entra en **espera**

| CTA 1 | CTA 2 | CTA 3 |
|---|---|---|
| 40 | 50 | 30 |

| Transacción A | Tiempo | Transacción B |
|---|---|---|
| -- | | -- |
| -- | | -- |
| TRAER CTA 1 (40): | t1 | -- |
| (**adquirir** bloqueo **S** sobre CTA 1) | | -- |
| suma = 40 | | -- |
| -- | | -- |
| TRAER CTA 2 (50): | t2 | -- |
| (**adquirir** bloqueo **S** sobre CTA 2) | | -- |
| suma = 90 | | -- |
| -- | t3 | TRAER CTA 3 (30): |
| -- | | (**adquirir** bloqueo **S** sobre CTA 3) |
| -- | | -- |
| -- | t4 | ACTUALIZAR CTA 3: |
| -- | | (**adquirir** bloqueo **X** sobre CTA 3) |
| -- | | 30 → 20 |
| -- | | -- |
| -- | t5 | TRAER CTA 1 (40): |
| -- | | (**adquirir** bloqueo **S** sobre CTA 1) |
| -- | | -- |
| -- | t6 | ACTUALIZAR CTA 1: |
| -- | | (**adquirir** bloqueo **X** sobre CTA 1) (*) |
| -- | | esperar |
| -- | | esperar |
| TRAER CTA 3 (**20**): | t7 | esperar |
| (**solicitar** bloqueo **S** sobre CTA 3) | | esperar |
| esperar | | esperar |
| esperar | | esperar |

# Bloqueo Mutuo

- **Bloqueo**: Resuelve problemas pero también los causa

- Bloqueo mutuo: Es una **situación** en la cual **dos o más transacciones** están en un **estado de espera simultáneo** y cada uno espera la liberación del otro para poder continuar.

# Bloqueo Mutuo

- El **sistema** debería **detectar** y **romper** estas situaciones, **eligiendo una** de las **transacciones** paralizadas y cancelándola

  ➔ se liberan sus bloqueos y continúa la otra.

- Enviará un código de retorno al programa y será responsabilidad del programa manejarla en una forma elegante.

# Ejemplo de bloqueo mutuo

| Transacción A | Tiempo | Transacción B |
|---|---|---|
| -- | | -- |
| -- | | -- |
| **solicitar** bloqueo **X** sobre **R1** | t1 | -- |
| -- | | -- |
| -- | | |
| -- | t2 | **solicitar** bloqueo **X** sobre **R2** |
| -- | | -- |
| | | -- |
| **solicitar** bloqueo **X** sobre **R2** | t3 | -- |
| esperar | | -- |
| esperar | | -- |
| esperar | t4 | **solicitar** bloqueo **X** sobre **R1** |
| esperar | | esperar |
| | | esperar |