Modelos y bases de datos Recuperación y concurrencia

CEIS

2023-2

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

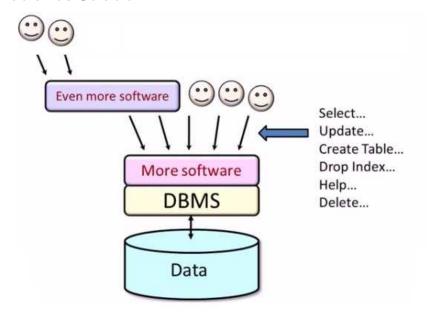
Problemas

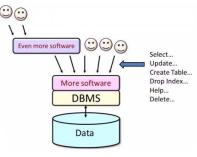
Solución

Recuperación

Problemas

Solución



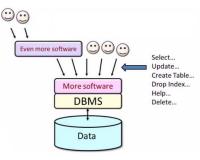


"MOOC Databases. Standford."

:(Dos aplicaciones que están ejecutando simultáneamente usan los mismos datos

(El sistema falla cuando se está ejecutando

Retirar dinero en un cajero



"MOOC Databases, Standford,"

:(Dos aplicaciones que están ejecutando simultáneamente usan los mismos datos

Puede quedar la base de datos con datos incorrectos

Concurrencia

¿Cómo garantizamos una ejecución concurrente correcta?

 El sistema falla cuando se está ejecutando

Puede quedar la base de datos en un estado desconocido (correcto o incorrecto)

Recuperación

¿Cómo volvemos a un estado correcto y conocido?

ADMINISTRACION DE TRANSACCIONES

Retirar dinero en un cajero



Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

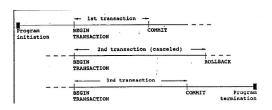
Problemas Solución

Recuperación

Problemas Solución

Definición

Contexto

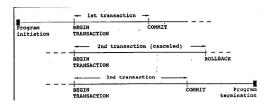


Propiedades. ACID

Definición

Es una secuencia de operaciones que transforman un estado consistente en otro estado consistente. Unidad de trabajo lógica.

Contexto



La ejecución de un programa es la ejecución de una secuencia de transacciones que pueden ser **concurrentes**

Propiedades. ACID

- A Atomicidad
- C Consistencia
- I Aislamiento
- D Durabilidad



Transferencia

```
CREATE OR REPLACE
PROCEDURE TRANSFERENCIA(origen IN CHAR, destino IN CHAR, valor IN NUMBER)
IS
BEGIN
UPDATE CUENTAS SET saldo=saldo+valor WHERE numero=destino;
UPDATE CUENTAS SET saldo=saldo-valor WHERE numero=origen;
END;
/
```

Hay un chequeo de saldo>=0

▶ ¿Es correcto?

Transferencia

```
CREATE OR REPLACE
PROCEDURE TRANSFERENCIA(origen IN CHAR, destino IN CHAR, valor IN NUMBER)
IS
BEGIN

UPDATE CUENTAS SET saldo=saldo+valor WHERE numero=destino;
IF (SOL%ROWCOUNT=0) THEN

RAISE_APPLICATION_ERROR(-20001, 'No existe la cuenta destino');
END IF;
UPDATE CUENTAS SET saldo=saldo-valor WHERE numero=origen;
IF (SOL%ROWCOUNT=0) THEN

RAISE_APPLICATION_ERROR(-20001, 'No existe la cuenta origen');
END IF;
END;
//
```

Hay un chequeo de saldo>=0

- ► ¿Es correcto?
- ▶ ¿Qué pasa si el sistema se cae después de realizarse la primera instrucción?

Transferencia

```
CREATE OR REPLACE
PROCEDURE TRANSFERENCIA(origen IN CHAR, destino IN CHAR, valor IN NUMBER)
IS
BEGIN

UPDATE CUENTAS SET saldo=saldo+valor WHERE numero=destino; leer,actualizar,escribir
IF (SQL*RCWCOUNT=0) THEN

RAISE_APPLICATION_ERROR(-20001,'No existe la cuenta destino');
END IF;

UPDATE CUENTAS SET saldo=saldo-valor WHERE numero=origen; leer,actualizar,escribir
IF (SQL*RCWCOUNT=0) THEN

RAISE_APPLICATION_ERROR(-20001,'No existe la cuenta origen');
END IF;
END;
//
```

Hav un chequeo de saldo>=0

- ▶ ;Es correcto?
- ▶ ¿Qué pasa si el sistema se cae después de realizarse la primera instrucción?
- ▶ ¿Qué puede pasar si hay dos transacciones concurrentes?

Elementos

UN INICIO

1. BEGIN TRANSACTION

Indica el inicio de la transacción

DOS POSIBLES FINALES

COMMIT

Indica la finalización de una transacción satisfactoria Los cambios ahora son permanentes y visibles a todos los usuarios

ROLLBACK

Indica la finalización de una transacción no satisfactoria Los cambios no son válidos, nadie los verá

Puntos

- De sincronización Las instrucciones COMMIT y ROLLBACK establecen puntos de sincronización; son puntos en que la base de datos debería estar en un estado de consistencia
- De verificación Escritura a disco, a determinados intervalos de tiempos prescritos



Transferencia

```
CREATE OR REPLACE
PROCEDURE TRANSFERENCIA(origen IN CHAR, destino IN CHAR, valor IN NUMBER)
IS
BEGIN

UPDATE CUENTAS SET saldo=saldo+valor WHERE numero=destino;
IF (SQL%RCWCOUNT=0) THEN

RAISE_APPLICATION_ERROR(-20001,'No existe la cuenta destino');
END IF;
UPDATE CUENTAS SET saldo=saldo-valor WHERE numero=origen;
IF (SQL%RCWCOUNT=0) THEN

RAISE_APPLICATION_ERROR(-20001,'No existe la cuenta origen');
END IF;
END;
//
```

- ► ; BEGIN TRANSACTION?
- ► ¿COMMIT?
- ► ;ROLLBACK?

Transferencia

```
CREATE OR REPLACE
PROCEDURE TRANSFERENCIA(origen IN CHAR, destino IN CHAR, valor IN NUMBER)
IS
BEGIN
  -- BEGIN TRANSACTION
  UPDATE CUENTAS SET saldo=saldo+valor WHERE numero=destino:
  IF (SQL%ROWCOUNT=0) THEN
     RAISE APPLICATION ERROR(-20001, 'No existe la cuenta destino');
  END IF:
  UPDATE CUENTAS SET saldo-saldo-valor WHERE numero-origen;
  IF (SOL%ROWCOUNT=0) THEN
     RAISE_APPLICATION_ERROR(-20001, 'No existe la cuenta origen');
  END IF:
  COMMIT:
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK:
    RAISE APPLICATION_ERROR(-20999, SQLERRM);
END:
```

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

Corrección

De un programa

De base de dase de datos

De conjunto de transacciones

Corrección

De un programa

:) Si cumple la precondición debe cumplir la poscondición

De base de dase de datos

:) Si cumple todas las restricciones de integridad definidas

De conjunto de transacciones

:) Si es serializable

Produce el mismo resultado que una ejecución serial de las mismas transacciones

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

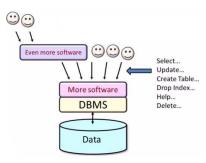
Problemas

Solución

Recuperación

Problemas

Solución



"MOOC Databases Standford"

:(Dos aplicaciones que están ejecutando simultáneamente usan los mismos datos

Puede quedar la base de datos con datos incorrectos

Concurrencia

 $\label{eq:concurrente} \mbox{ξ C\'omo garantizamos una ejecuci\'on concurrente} \mbox{$\mbox{$correcta$?}$}$

ADMINISTRACION DE TRANSACCIONES



Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

1. Actualización perdida

Lectura no repetible

2. Dependencia no confirmada

Lectura sucia

3. Análisis inconsistente

Lectura no repetible

4

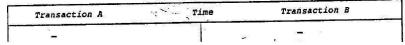
Fantasmas

Actualización perdida

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

¿Cuáles serían posibles resultados de una ejecución correcta?

Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000



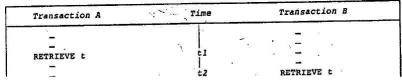
Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

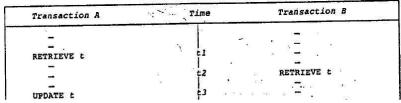


Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

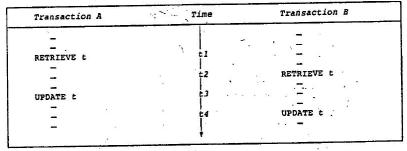
Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

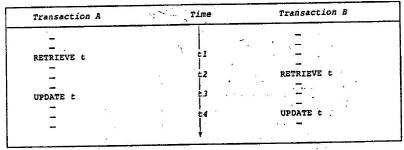


Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

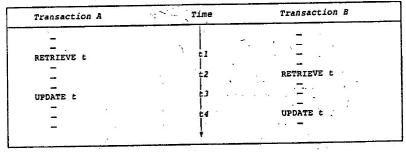
Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

¿Es correcto?

Actualización perdida



- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

¿Es correcto? ¡No es correcto! Lectura no repetible

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

Idea

Cuando una transacción quiere asegurar que el objeto en el que está interesada (normalmente una tupla) no cambiará mientras la está usando adquiere un **bloqueo** sobre ese objeto.

Tipos de bloqueo

- Exclusivo (X)
- ► Compartido (S)

Idea

Cuando una transacción quiere asegurar que el objeto en el que está interesada (normalmente una tupla) no cambiará mientras la está usando adquiere un **bloqueo** sobre ese objeto.

Tipos de bloqueo

- Exclusivo (X)
 - Si la transacción A pone un bloqueo exclusivo X sobre la tupla t,
 - se rechazará la petición de cualquier otra transacción B para un bloqueo de cualquier tipo sobre t
- Compartido (S)
 - Si la transacción A pone un bloqueo compartido S sobre la tupla t,
 - se rechazará la petición de cualquier otra transacción B para un bloqueo X sobre t
 - se otorgará una petición de cualquier otra transacción para un bloqueo S sobre t

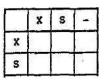
Idea

Cuando una transacción quiere asegurar que objeto en el que está interesada (normalmente una tupla) no cambiará mientras la está usando adquiere un **bloqueo** sobre ese objeto.

Tipos de bloqueo

- ► Exclusivo (X)
- ► Compartido (S)

Completar tabla



Tlene

Idea

Cuando una transacción quiere asegurar que objeto en el que está interesada no cambiará mientras la está usando adquiere un **bloqueo** sobre ese objeto.

Granularidad del bloqueo

- TuplaNormalmente es una tupla
- ▶ Tabla
- Base de datos

Transacciones

Propiedadades. ACID. [I] Aislamiento. Niveles.

1. READ UNCOMMITTED: Lectura no registrada

2. READ COMMITTED: Lectura registrada

3. REPEABLE READ : Lectura repetible

4. SERIALIZABLE : Serializable

Transacciones

Propiedadades. ACID. [I] Aislamiento. Niveles.

- READ UNCOMMITTED : Lectura no registrada sin bloqueos
- READ COMMITTED : Lectura registrada Bloqueo exclusivo para actualizar
- REPEABLE READ : Lectura repetible
 Bloqueo exclusivo para actualizar y compartido para leer
- 4. SERIALIZABLE : Serializable
 Bloqueo exclusivo para actualizar y compartido para leer
 Bloqueos de rango en las consultas (no cambiar WHERE)

Bloqueo

Protocolo: Lectura repetible

Este es el protocolo normal de acceso a los datos

Operaciones - bloqueo

Para leer

Una transacción que desea recuperar una tupla primero debe adquirir un bloqueo compartido S sobre la tupla

- (S) Compartido = De lectura
- Para escribir

Una transacción que desea actualizar una tupla primero debe adquirir un bloqueo exclusivo X sobre la tupla

(X) Exclusivo = De escritura

Actualización perdida

. Transaction A	Time	Transaction B
T 7 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3		

- Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

¿Cuáles serían posibles resultados de una ejecución correcta?

Actualización perdida

. Transaction	A		Time	Transaction B
RETRIEVE t		e die	t1	
(acquire S	lock on t	: 1	1	· - :

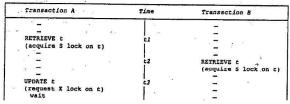
- Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Actualización perdida

Transaction A	Time	Transaction B
		_
RETRIEVE t	t1	·
(acquire S lock on t)		* <u>-</u>
		-
•	t2	RETRIEVE t
	1	(acquire 5 lock on c)
-	i .	

- Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Actualización perdida



- Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Actualización perdida

. Transaction A	6 7 4	Time	Transaction B
· <u>-</u> · · · ·		T :	_
· -		40.0	_
RETRIEVE t		t1	
(acquire S lo	ck on t)	1	* <u>=</u>
· . · -	3.50	1	_
-		£2	RETRIEVE &
		1	(acquire 5 lock on c)
_			(4544226 6 2007 611 2)
UPDATE &		£3	
(request X lo	ck on tl	ĩ	<u>-</u>
wait			=
wait		£4	UPDATE ¢
wait		ř	(request X lock on t)
wait	•	1	wait
Wait		1	wait
wait		1	wait
		1	Walt

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Actualización perdida

. Transaction A	6 7 4	Time	Transaction B
· <u>-</u> · · · ·		T :	_
· -		40.0	_
RETRIEVE t		t1	
(acquire S lo	ck on t)	1	* <u>=</u>
· . · -	3.50	1	_
-		£2	RETRIEVE &
		1	(acquire 5 lock on c)
_			(4544226 6 2007 611 2)
UPDATE &		£3	
(request X lo	ck on tl	ĩ	<u>-</u>
wait			=
wait		£4	UPDATE ¢
wait		ř	(request X lock on t)
wait	•	1	wait
Wait	10	1	wait
wait		1	wait
		1	Walt

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Actualización perdida

Transaction A	Time	Transaction B
	T is	
	10.00	_
RETRIEVE t	t1	
(acquire S lock on t)	1	·
· . · -	1	<u>=</u>
· ·	t2	RETRIEVE t
	1	(acquire 5 lock on c)
-	1 .	-
UPDATE t	t.3	_
(request X lock on t)	1	
wait		
wait	t4	UPDATE t
wait	1	(request X lock on t)
wait		wait
wait		wait
wait	1	wait
	i	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

¿Es correcto?

Actualización perdida

. Transaction A	6 7 4	Time	Transaction B
· <u>-</u> · · · ·		T :	_
· -		40.0	_
RETRIEVE t		t1	
(acquire S lo	ck on t)	1	* <u>=</u>
· . · -	3.50	1	_
-		£2	RETRIEVE &
		1	(acquire 5 lock on c)
_			(4544226 6 2007 611 2)
UPDATE &		£3	
(request X lo	ck on tl	ĩ	<u>-</u>
wait			=
wait		£4	UPDATE ¢
wait		ř	(request X lock on t)
wait	•	1	wait
Wait	10	1	wait
wait		1	wait
		1	Walt

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

¿Es correcto? Bloqueo mortal

Bloqueo mortal

¿Cómo se soluciona?

¿Cómo se detecta?

Bloqueo mortal

¿Cómo se soluciona?

Seleccionar una de las transacciones del bloqueo, **la víctima** y deshacerla, liberando sus bloqueos,

¿Cómo se detecta?

- Mantener grafo de espera ¿Qué información guardamos?
 ¿Qué buscamos?
- 2. Control de tiempo
 ¿Qué información guardamos?
 ¿Qué buscamos?

Agenda

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

Aspectos finales

Agenda

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

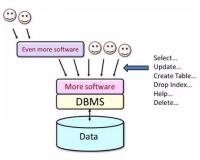
Recuperación

Problemas

Solución

Aspectos finales

Problemas-Solución



"MOOC Databases Standford"

:(Dos aplicaciones que están ejecutando simultáneamente usan los mismos datos

Puede quedar la base de datos con datos incorrectos

Concurrencia

¿Cómo garantizamos una ejecución concurrente correcta?

(El sistema falla cuando se está ejecutando

Puede quedar la base de datos en un estado desconocido (correcto o incorrecto)

Recuperación

¿Cómo volvemos a un estado correcto y conocido?

ADMINISTRACION DE TRANSACCIONES

Fallas locales

Fallas globales

Fallas locales

Falla de operación

Fallas globales

- ► Falla del sistema
- ► Falla del medio

Agenda

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problema:

Solución

Aspectos finales

Solución

Redundancia

Garantizar que cualquier parte de la información que contiene el sistema puede ser recuperada a partir de otra información guardada redundantemente en otro lugar del sistema

- Bitacora
 Mantiene registro de los cambios que cada transacción ha hecho a la base de datos
- ► Copias de respaldo

 Archivar la base de datos a un segundo disco

Solución

Punto de sincronización

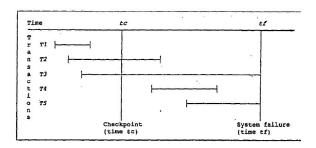
► Las instrucciones COMMIT y ROLLBACK establecen puntos de sincronización; son puntos en que la base de datos debería estar en un estado de consistencia

Puntos de verificación

 Escritura a disco, a determinados intervalos de tiempos prescritos

Soluciíon

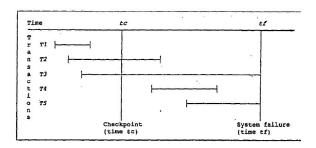
¿Cuáles son las acciones?



- ▶ ¿Qué queda listo?
- ▶ ¿Qué se debe rehacer?
- ▶ ¿ Qué se debe deshacer?

Soluciíon

¿Cuáles son las acciones?



- ¿Qué queda listo?
 Todo el trabajo realizado por las transacciones terminaron antes del punto de verificación
- ¿Qué se debe rehacer?
 Todo el trabajo realizado por las transacciones que terminaron satisfactoriamente antes de la caída
 Recuperación hacia adelante
- ¿ Qué se debe deshacer?
 Todo el trabajo realizado por las transacciones que no terminaron antes de la caída
 Recuperación hacia atrás

Agenda

Introducción

Problemas Solución

Transacciones

Corrección

Concurrencia

Problemas

Solución

Recuperación

Problemas

Solución

Aspectos finales

Example

Transaction A	.; ` _ Time	Transaction B
		2-1
=		3

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- ► La transacción B es un retiro de 500.000



Example

Transaction A	Time	Transaction B
<u>-</u>		·
RETRIEVE t	t I	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000



Example

Transaction A	Time	Transaction B
_		i ⇒ 1
-	, I,	
RETRIEVE t	ţ1	¥ ²⁰ 0 ⊊
	.1_	
	t2	RETRIEVE t

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000



Example

Transaction A	Time	Transaction B
		-
RETRIEVE t	۱ ا	
=	 t2	RETRIEVE t
UPDATE E	. [

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Example

Transaction A	Time	Transaction B
_		-
- .	× 1,	· · · · · · ·
RETRIEVE t	, ţ1	· ***
_	1	•
	t2	RETRIEVE t
		· · · · · · · · · · · · · · · · · · ·
UPDATE t	Ė3	
DEDRIE L	1	. .
*	. t4	UPDATE t
	ĭ'	=
- 4		- · · -
10	Y	- 65

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Example

Transaction A	Time	Transaction B
_		, i= 10
-	, 1	· · · · · · · · · · · · · · · · · · ·
RETRIEVE t	t1	· · · · · · · · · · · · · · · · · · ·
	1	
_	t2	RETRIEVE t
- 		· · · · · · · · · · · · · · · · · · ·
	<u> </u>	· • · · · · · · · · · · · · · · · · · ·
UPDATE t	Г	
-	t4	UPDATE t
-	Ęª.	
- :		* · · -
	•	6

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

. Transaction A	Time	Transaction B
	7	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

. Transaction	A ·	Time	Transaction B
_		1 1	
			_
RETRIEVE t		t1	
(acquire S)	lock on t)	1	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Transaction A	Time	Transaction B
-	T a	
		_
RETRIEVE t	t1	*
(acquire S lock on t)	1	* <u>-</u>
·	1	<u>=</u>
	ĖZ	RETRIEVE t
'	1	(acquire 5 lock on c)
-	1 .	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Transaction A	Time	Transaction B
- · · · · · · · · · · · · · · · · · · ·	1 1	
		_
RETRIEVE t	t1	*
(acquire S lock on t)	1	* <u>*</u> * * *
·	1	
	£2	RETRIEVE &
	77	(acquire 5 lock on c
_	1	(acquire a rock on c
UPDATE &		·-
		1 -1 2
(reduese v fock ou E)	1 .	1. 1
wait	1	_

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Transaction A	Time	Transaction B
_		_
		_
RETRIEVE t	t1	
(acquire S lock on t)	1	* <u>-</u> 1 1
	1	
· · ·	£2	RETRIEVE &
	ï	(acquire 5 lock on c
-	1	(acquire a lock on E
UPDATE t	. 7	
(request X lock on t)	ř	
wait		
wait	£4	UPDATE ¢
wait		
wait	1	(request X lock on t
	i	wait
wait		wait -
wait	1	wait

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Actualización perdida

Transaction A	Time	Transaction B
		_
		_
RETRIEVE t	t1	-
(acquire S lock on t)	1	· <u>-</u>
· . :-	1	_
	£2	RETRIEVE t
	í i	(acquire 5 lock on c
-	i .	(
UPDATE t	£.3	
(request X lock on t)	i i	_
wait		<u> </u>
wait	±4	UPDATE ¢
wait	Ĩ.	(request X lock on t
wait		wait
wait .	1	wait -
wait	1	wait
	1	WOTE

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consignación de 250.000
- La transacción B es un retiro de 500.000

Bloqueo mortal

Transaction A	Time	Transaction B
Transaction II	2 43114	TTENDECETON D
NAME OF THE PROPERTY OF THE PARTY OF THE PAR		

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta el saldo
- La transacción B trata de hacer un retiro de 500.000 pero se arrepiente

Time	Transaction B
1	
t1	UPDATE t
	Time

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta el saldo
- La transacción B trata de hacer un retiro de 500.000 pero se arrepiente

Transaction A		Time	Transaction B
_			
-	25	1	-
-		¢1	UPDATE t
-		Ĭ	
RETRIEVE t			

- Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta el saldo
- La transacción B trata de hacer un retiro de 500.000 pero se arrepiente

Transaction A		Time	Transaction B
PV0			-
-	33		_
-		t1	UPDATE t
-			≃
RETRIEVE t	•	£2 . ·	
-	•		= -
-		ĖJ	ROLLBACK
=		1	
		1	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta el saldo
- ▶ La transacción B trata de hacer un retiro de 500.000 pero se arrepiente

Dependencia no confirmada

Transaction A		Time	Transaction B
			_
-	33	- 1	<u></u>
-		t1	UPDATE t
		ì	<u> -</u>
RETRIEVE t		£2 . •	
-	•		≒ =
		ĖJ	ROLLBACK
-		1	
		1 .	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta el saldo
- La transacción B trata de hacer un retiro de 500.000 pero se arrepiente

¡ No es correcto!. Lectura sucia



Transaction A	Time '	Transaction B
	1.1	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta de saldo
- La transacción B es un retiro de 500.000

Transaction A	Time '	Transaction B
· · ·	1	
-		-
-	ŧ1	UPDATE t
=		(acquire X lock on t)

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta de saldo
- La transacción B es un retiro de 500.000

Transaction A	Time '	Transaction B
	. 1	=
-		-
	t1	UPDATE t
=	. 1	(acquire X lock on t)
-	1	
RETRIEVE t	c.2	
(request S lock on t)		-

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta de saldo
- ► La transacción B es un retiro de 500.000

Transaction A	Time		Transaction B
	. 1		.=x
·	1	X	-
	t1		UPDATE t
-	-		(acquire X lock on t
-	1		
RETRIEVE t	t2		·
(request S lock on t)		r	
wait	- 1	×.	-
wait	£3	*	COMMIT / ROLLBACK
wait			(release X lock on t

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta de saldo
- La transacción B es un retiro de 500.000

Transaction A	Time	Transaction B
· ;	. 1	
-		-
	t1	UPDATE t
-	1	(acquire X lock on t
-	1	· –
RETRIEVE t	t2	-
(request S lock on t)		
wait	1	7
wait	t.3	COMMIT / ROLLBACK
wait		(release X lock on t
resume : RETRIEVE t	¢4	
(acquire S lock on t)	1	
-		

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta de saldo
- La transacción B es un retiro de 500.000

Dependencia no confirmada

-	. !	
-		-
	t1	UPDATE È
=		(acquire X lock on t
-		· -
RETRIEVE t	t2	-
(request S lock on t)		d 150
wait	1 "	7
wait	t.3	COMMIT / ROLLBACK
wait		(release X lock on t
resume : RETRIEVE t	C4	
(acquire S lock on t)	1	
-	L	

- ► Tenemos una cuenta t con saldo 1.000.000
- La transacción A es una consulta de saldo
- La transacción B es un retiro de 500.000

¡Es correcto!

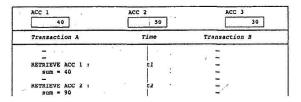


ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
Transaction A	Time	Transaction B
		_
RETRIEVE ACC 1 :	L.	, =
sum = 40	i :	=

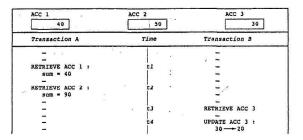
- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1



- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
Transaction A	Time	Transaction B
		-
=	1	, -
RETRIEVE ACC 1 :	t1	-
sum = 40	1 :	=
_		-
RETRIEVE ACC 2 :	£2	- ✓
sum = 90		
	1	
	+3	RETRIEVE ACC 3

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1



- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
Transaction A	Time	Transaction B
		-
-		, -
RETRIEVE ACC 1 :	t1	-
sum = 40	1 :	
_		-
RETRIEVE ACC 2 :	£2 ·	- , ≥
sum = 90		
2	1	=
_	t.3	RETRIEVE ACC 3
=		-
_	t4	UPDATE ACC 3 :
_	1	30 20
- 75.00	1	-
-	t.5	RETRIEVE ACC 1
	1	

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
Transaction A	Time	Transaction B
- : :		-
	1	, -
RETRIEVE ACC 1 :	ΕI	-
sum = 40	:	-
_	1	-
RETRIEVE ACC 2 :	c2 ·	- ,≥
sum = 90		<u>~</u> "
=	1	=
_	c3	RETRIEVE ACC 3
~		
-	t4	UPDATE ACC 3 :
_	1	30 20
_		=
_	- A:	RETRIEVE ACC 1
-		-
_	£6	UPDATE ACC 1 :
	I I	40 50

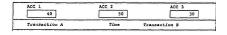
- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1 40	ACC 2	ACC 3
Transaction A	Time	Transaction B
- : :		-
=	1	, –
RETRIEVE ACC 1 :	t1	-
sum = 40	;	_
=		-
RETRIEVE ACC 2 :	£2	- ,≥
sum = 90		- *
-		<u>-</u>
_	. t3	RETRIEVE ACC 3
	1 .	_
_	t4	UPDATE ACC 3 :
_	1	30 20
_	P 23:	-
	t.5	RETRIEVE ACC 1
_	3 T	
_	£6	UPDATE ACC 1 :
	. 1	40 50
		_
	£7'	COMMIT

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1 .	ACC 2	ACC 3
Transaction A	Time	Transaction B
- : :		-
-		, -
RETRIEVE ACC 1 :	tI	-
sum = 40		_
_		-
RETRIEVE ACC 2 :	t2	- /
sum = 90		-
_	1	_
_	. t3	RETRIEVE ACC 3
=	1 .	
_	t4	UPDATE ACC 3 :
-		30 20
_	mat.	A CONTRACTOR OF THE PARTY OF TH
-	t5	RETRIEVE ACC 1
-		was a second base of the second second
	£6	UPDATE ACC 1 :
-		40 50
		-
	£7'	COMMIT
_		
RETRIEVE ACC 3 :	t8	
sum = 110, not 120		
	1	

- La transacción A está calculando el saldo de las cuentas
- ► La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1



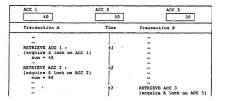
- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
_		
-	1.7	_
RETRIEVE ACC 1 :	tl ·	_
(acquire S lock on ACC 1)		_
sum = 40	1	-

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
-		
-	1.7	_
ETRIEVE ACC 1 :	tI .	-
acquire S lock on ACC 1)	1	_
sum = 40	1	-
-	1	_
ETRIEVE ACC 2 :	t2	-
acquire S lock on ACC 2)	1	_
sum - 90		-

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1



- La transacción A está calculando el saldo de las cuentas
- ► La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
_		<u>-</u>
-	1.7	-
RETRIEVE ACC 1 :	tl ·	-
(acquire S lock on ACC 1)		_
sum = 40		-
-	1	_
RETRIEVE ACC 2 :	£2	-
(acquire S lock on ACC 2)	T.	_
sum = 90		_
-	1	-
- 8	t3	RETRIEVE ACC 3
-	1	(acquire S lock on ACC 3
14	1	-
-	±4	UPDATE ACC 3
	1	(acquire X lock on ACC 3
-	1	30

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
-		
-	1.6	_
RETRIEVE ACC 1 :	¢1 .	-
(acquire S lock on ACC 1)		_
sum = 40	1	=
and the same of th	1	_
RETRIEVE ACC 2 :	t.2	_
(acquire S lock on ACC 2)	1	_
sum - 90	1	to <u>-</u>
-	1	-
- *	t3	RETRIEVE ACC 3
_	- 1	(acquire S lock on ACC 3
·_	1	
_	±4	UPDATE ACC 3
= -		(acquire X lock on ACC 3
-	- 1	30 20
-	1	-
_	£.5	RETRIEVE ACC 1
-	- 1	(acquire S lock on ACC 1

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
-		
_	1.4	-
RETRIEVE ACC 1 :	¢1 .	-
(acquire S lock on ACC 1)	1 "	-
sum = 40	1	<u> </u>
-	1	-
RETRIEVE ACC 2 :	t.2	_
(acquire S lock on ACC 2)	ľ	_
sum - 90	1	5)
-	1	-
- *	t3	RETRIEVE ACC 3
-	1	(acquire S lock on ACC 3
	1	
-	±4	UPDATE ACC 3
-	1	(acquire X lock on ACC 3
-	1	30 20
-	1	=
_	t.5	RETRIEVE ACC 1
-	- 1	(acquire S lock on ACC 1
-	1	
_	£6	UPDATE ACC 1
=		(request X lock on ACC 3
-		wait

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

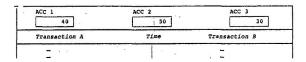
ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
-		_
-	1.4	-
RETRIEVE ACC 1 :	¢1 .	-
(acquire S lock on ACC 1)	1	_
sum = 40	1	-
-	1	_
RETRIEVE ACC 2 :	t.2	-
(acquire S lock on ACC 2)	ľ	_
sum = 90		_
-	- 1	-
- ×	¢3	RETRIEVE ACC 3
-	- 1	(acquire S lock on ACC 3
·-	1	_
-	±4	UPDATE ACC 3
-		(acquire X lock on ACC 3
-	1	30 20
-	į.	-
_	t.5	RETRIEVE ACC 1
-	1	(acquire S lock on ACC 1
-	1	
-	£6	UPDATE ACC 1
-	1	(request X lock on ACC 3
	1	wait
RETRIEVE ACC 3 :	£7	wait
(request S lock on ACC 3)	- 1	wait .
wait	1	wait
wait	1	wait

- La transacción A está calculando el saldo de las cuentas
- La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1

ACC 1	ACC 2	ACC 3
40	50	30
Transaction A	Time	Transaction B
-		_
_	1.4	-
RETRIEVE ACC 1 :	¢1 .	-
(acquire S lock on ACC 1)	1	-
sum = 40	1	-
-	1	_
RETRIEVE ACC 2 :	t.2	-
(acquire S lock on ACC 2)	ľ	_
sum - 90	1 :	-
-	1	-
- × .	t3	RETRIEVE ACC 3
-	- ((acquire S lock on ACC 3
·-	1	_
-	±4	UPDATE ACC 3
-	1	(acquire X lock on ACC 3
-	1	30 20
-	t	-
-	t.5	RETRIEVE ACC 1
-	- 1	(acquire S lock on ACC 1
-	L.	
-	£6	UPDATE ACC 1
-		(request X lock on ACC 3
J. J. T	1	wait
RETRIEVE ACC 3 :	£7	wait
(request S lock on ACC 3)	1	wait .
wait	1	wait
wait	1	wait

- La transacción A está calculando el saldo de las cuentas
- ► La transacción B está transfiriendo 10 de la cuenta ACC3 a la cuenta ACC1 BLOQUEO MORTAL

;?



- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- ► La transacción B está adicionando una nueva cuenta ACC4 100

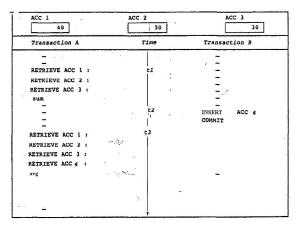
ACC 1	ACC 2	ACC 3
Transaction A	Time	Transaction B
- 1.1		_
-	1 .	, -
RETRIEVE ACC 1 :	t1	-
RETRIEVE ACC 2 :	:	-
		-
RETRIEVE ACC 3 :	25 Jan 2	- , r
sum		

- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- ► La transacción B está adicionando una nueva cuenta ACC4 100

ACC 1	ACC 2	ACC 3
Transaction A	Time	Transaction B
- 1. 1		-
	1	, -
RETRIEVE ACC 1 :	£1	-
RETRIEVE ACC 2 :		_
RETRIEVE ACC 3 :		-
		5 .4
sum	1	-
	£2	INSERT ACC 4
-		
-	1	COMMIT

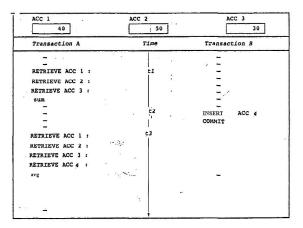
- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- La transacción B está adicionando una nueva cuenta ACC4 100

? ;



- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- ► La transacción B está adicionando una nueva cuenta ACC4 100

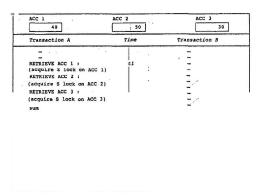
? خ



- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- ► La transacción B está adicionando una nueva cuenta ACC4 100 ¡No es correcto! Fantasmas.

ACC 1 40	ACC 2	ACC 3
	50	30
Transaction A	Time	Transaction B
- 11		-
	1 .	1 2

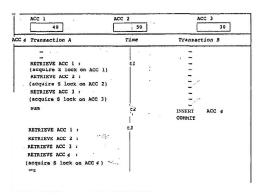
- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- La transacción B está adicionando una nueva cuenta ACC4 100



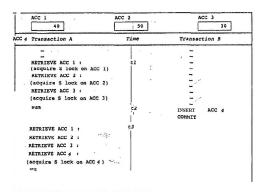
- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- La transacción B está adicionando una nueva cuenta ACC4 100

40 .	ACC 2	ACC 3
Transaction A	Time	Transaction B
		-
RETRIEVE ACC 1 :	13	/ 2
acquire S lock on ACC 1)	ĭ* :	Ē
ETRIEVE ACC 2 :		-
acquire S lock on ACC 2)		- J. C.
ETRIEVE ACC 3 :		~
acquire S lock on ACC 3)		-
sum		200
run		

- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- La transacción B está adicionando una nueva cuenta ACC4 100



- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- La transacción B está adicionando una nueva cuenta ACC4 100



- La transacción A está calculando primero la suma y luego el promedio de las cuentas
- La transacción B está adicionando una nueva cuenta ACC4 100 ¡No es correcto! Fantasmas.