

CC3201-1

BASES DE DATOS

OTOÑO 2023

Clase 10: Transacciones y ACID

Aidan Hogan

[aidhog@gmail.com](mailto:aidhog@gmail.com)

# Una cuenta bancaria ...

## Ingreso

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Deposito inicial	2020-01-21	20:02:02	300000	300000	TRCXGU8JSHD
7873698669	C0°0°L Designs	2020-02-06	09:15:33	50000	325000	TRCCIA2J8A0

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	225000	344,42

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,0001533
USD	CLP	652,2750000

## Cliente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

# Una cuenta bancaria ... integridad

## Ingreso

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Deposito inicial	2020-01-21	20:02:02	300000	300000	TRCXGU8JSHD
7873698669	C0°0°L Designs	2020-02-06	09:15:33	50000	325000	TRCCIA2J8A0

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	225000	344,42

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,0001533
USD	CLP	652,2750000

## Cliente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

# Restricciones sobre varias tablas

## Ingreso

cuenta	comentario	fecha	hora	monto	saldo	id
7873698669	Deposito inicial	2020-01-21	20:02:02	300000	300000	TRCXGU8JSHD
7873698669	C0°0°L Designs	2020-02-06	09:15:33	50000	325000	TRCCIA2J8A0

## Gasto

cuenta	comentario	fecha	hora	monto	saldo	id
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

## Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

```
CREATE TABLE Cuenta (  
  número INTEGER PRIMARY KEY,  
  rut VARCHAR (12) NOT NULL,  
  tipo VARCHAR (12) NOT NULL,  
  saldo_clp BIGINT NOT NULL,  
  saldo_usd FLOAT NOT NULL,  
  CHECK (  
    ( SELECT SUM(monto) FROM Ingreso WHERE cuenta=número )  
    - ( SELECT SUM(monto) FROM Gasto WHERE cuenta=número )  
    = saldo_clp ) )
```



¿A. P. A.? ...

INSERT INTO Ingreso ...

INSERT INTO Gasto ...

UPDATE Cuenta  
SET saldo\_clp ...

¿A. S.? ...

TRANSACCIONES

# Transacciones

Una **transacción** es

un **conjunto de operaciones** que

se ejecutan de **manera atómica**

(es decir, **como si fuera una sola operación**)

# Vacaciones ...



# Transacciones: START TRANSACTION/COMMIT

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD
7873698669	Noruega	2020-02-12	02:14:20	400000	-175000	TRCLK9K24KS

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	-175000	-268,29

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,001533
USD	CLP	652,275000

## Cliente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

```
START TRANSACTION;  
INSERT INTO Gasto VALUES  
  (7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');  
UPDATE Cuenta SET saldo_clp=-175000, saldo_usd=-268.29 WHERE número=7873698669;  
COMMIT;
```

**START TRANSACTION** (o a veces **BEGIN**) inicia la transacción  
**COMMIT** realiza/guarda los cambios

# Transacciones (por defecto)

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD
7873698669	Noruega	2020-02-12	02:14:20	400000	-175000	TRCLK9K24KS

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	-175000	-268,29

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,001533
USD	CLP	652,275000

## Cliente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

```
INSERT INTO Gasto VALUES
```

```
(7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');
```

```
-- COMMIT;
```

```
UPDATE Cuenta SET saldo_clp=-175000, saldo_usd=-268.29 WHERE número=7873698669;
```

```
-- COMMIT;
```

Si no hay una transacción explícita, por defecto, Postgres hace un **COMMIT** después de cada sentencia (pero se puede cambiar la configuración)

# Transacciones: ROLLBACK

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	225000	344,94

## Cliente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,001533
USD	CLP	652,275000

```
START TRANSACTION;  
INSERT INTO Gasto VALUES  
(7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');  
ROLLBACK;
```

**ROLLBACK** deshace/borra los cambios desde el inicio de la transacción

# Transacciones: SAVEPOINT

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD
7873698669	Noruega	2020-02-12	02:14:20	400000	-175000	TRCLK9K24KS

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	-175000	-268,29

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,001533
USD	CLP	652,275000

## Cliente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

```
START TRANSACTION;
```

```
INSERT INTO Gasto VALUES
```

```
(7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');
```

```
UPDATE Cuenta SET saldo_clp=-175000, saldo_usd=-268.25 WHERE número=7873698669;
```

```
SAVEPOINT CompraNoruega;
```

```
INSERT INTO Gasto VALUES
```

```
(7873698669, 'BOSE', '2020-02-12', 200000, -375000, 'TRCASD8PNAK');
```

```
ROLLBACK TO SAVEPOINT CompraNoruega;
```

```
COMMIT;
```

**ROLLBACK** puede deshacer/borrar los cambios desde un punto específico con **SAVEPOINT**

# Una transacción con valores dinámicos

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	225000	344,94

## Ciente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,001533
USD	CLP	652,275000

```
START TRANSACTION;  
INSERT INTO Gasto VALUES  
  (7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');  
UPDATE Cuenta SET saldo_clp=(saldo_clp-400000) WHERE número=7873698669;  
UPDATE Cuenta SET saldo_usd=(A.saldo_clp/valor)  
  FROM ( SELECT valor FROM Divisa WHERE d1='USD' AND d2='CLP') T,  
  ( SELECT saldo_clp FROM Cuenta WHERE número=7873698669 ) A  
  WHERE número=7873698669;  
COMMIT;
```

¿Valor final de `saldo_usd` en Cuenta?

# Una transacción con valores dinámicos

Gasto							
cuenta	comentario	fecha	hora	monto	saldo	id	
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A	
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS	
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D	
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD	
7873698669	Noruega	2020-02-12	02:14:20	400000	-175000	TRCLK9K24KS	

Cuenta				
número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	-175000	-268,28

Divisa		
d1	d2	valor
CLP	USD	0,001533
USD	CLP	652,275000

Cliente			
rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

```
START TRANSACTION;
INSERT INTO Gasto VALUES
(7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');
UPDATE Cuenta SET saldo_clp=(saldo_clp-400000) WHERE número=7873698669;
UPDATE Cuenta SET saldo_usd=(A.saldo_clp/valor)
FROM ( SELECT valor FROM Divisa WHERE d1='USD' AND d2='CLP') T,
( SELECT saldo_clp FROM Cuenta WHERE número=7873698669 ) A
WHERE número=7873698669;
COMMIT;
```

¿Valor final de `saldo_usd` en Cuenta? **-268,28** (Se lee el valor actual de la misma transacción)

# Una transacción con CHECK

## Gasto

<u>cuenta</u>	<u>comentario</u>	<u>fecha</u>	<u>hora</u>	<u>monto</u>	<u>saldo</u>	<u>id</u>
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

## Cuenta

<u>número</u>	<u>rut</u>	<u>tipo</u>	<u>saldo_clp</u>	<u>saldo_usd</u>
7873698669	32.000.273-K	Estacional	225000	344,94

## Ciente

<u>rut</u>	<u>nombre</u>	<u>fono</u>	<u>dirección</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

## Divisa

<u>d1</u>	<u>d2</u>	<u>valor</u>
CLP	USD	0,001533
USD	CLP	652,275000

```
START TRANSACTION;
```

```
INSERT INTO Gasto VALUES
```

```
(7873698669, 'Noruega', '2020-02-12', '02:14:20', 400000, -175000, 'TRCLK9K24KS');
```

```
UPDATE Cuenta SET saldo_clp=(saldo_clp-400000) WHERE número=7873698669;
```

```
UPDATE Cuenta SET saldo_usd=(A.saldo_clp/valor)
```

```
FROM ( SELECT valor FROM Divisa WHERE d1='USD' AND d2='CLP') T,
```

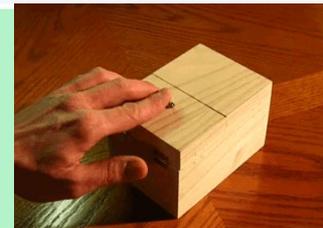
```
( SELECT saldo_clp FROM Cuenta WHERE número=7873698669 ) A
```

```
WHERE número=7873698669;
```

```
COMMIT;
```

```
CREATE TABLE Cuenta (...
```

```
CHECK ( saldo_clp = saldo_usd * (SELECT valor FROM Divisa WHERE d1='USD' AND d2='CLP') ) )
```



¿Funciona?

¡No!

# Transacciones / Restricciones: IMMEDIATE

Cuenta				
número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

Banco de Chile

```
CREATE TABLE Cuenta (  
  ...,  
  CONSTRAINT Cuenta_PK  
    PRIMARY KEY (número)  
)
```

```
START TRANSACTION;  
  INSERT INTO Cuenta VALUES  
    (7873698669, '32.000.273-K', 'Estacional',  
     -175000, -268.29);  
  DELETE FROM Cuenta  
    WHERE número=7873698669 AND saldo_clp=225000;  
COMMIT;
```

Por defecto, se aplica la restricción inmediatamente después de cada sentencia

# Transacciones / Restricciones: DEFERRABLE

Cuenta				
<u>número</u>	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

Cuenta				
<u>número</u>	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	-175000	-268,29

```
CREATE TABLE Cuenta (  
  ...,  
  CONSTRAINT Cuenta_PK  
    PRIMARY KEY (número)  
    DEFERRABLE  
)
```

```
START TRANSACTION;  
SET CONSTRAINT Cuenta_PK DEFERRED;  
INSERT INTO Cuenta  
  (7873698669, '32.000.273-K', 'Estacional',  
   -175000, -268.29);  
DELETE FROM Cuenta  
  WHERE número=7873698669 AND saldo_clp=225000;  
COMMIT;
```

**DEFERRABLE** define una restricción que se puede diferir hasta un **COMMIT**

**DEFERRED** difiere la restricción hasta el **COMMIT** en la transacción actual

# Transacciones / Restricciones: DEFERRABLE

Cuenta				
<u>número</u>	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

Cuenta				
<u>número</u>	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	-175000	-268,29

```
CREATE TABLE Cuenta (  
  ...,  
  CONSTRAINT Cuenta_PK  
    PRIMARY KEY (número)  
  DEFERRABLE INITIALLY DEFERRED  
)
```

**DEFERRABLE INITIALLY DEFERRED**  
define una restricción que sea diferida por defecto hasta un **COMMIT**

```
START TRANSACTION;  
  INSERT INTO Cuenta  
    (7873698669, '32.000.273-K', 'Estacional',  
     -175000, -268.29);  
  DELETE FROM Cuenta  
    WHERE número=7873698669 AND saldo_clp=225000;  
COMMIT;
```

Atomicidad, Coherencia, Aislamiento, Durabilidad  
(*Atomicity, Consistency, Isolation, Durability*)

# LAS GARANTÍAS DE ACID

No hay un solo usuario ...



... hay que tener cuidado con la concurrencia

# Una cuenta con varios usuarios



**Banco de Chilly**

N° Cuenta : 7873698669  
Saldo (CLP) : 225000  
Límite de crédito : 200000  
Disponible : 425000



```
CREATE TABLE Cuenta ( ..., CHECK ( saldo_clp > -200000 ) )
```

COMPRA(Islas de Caimán,300000)

COMPRA(Noruega,400000)

```
INSERT INTO Gasto ...
```

```
INSERT INTO Gasto ...
```

```
UPDATE Cuenta ...
```

```
UPDATE Cuenta ...
```

*¿Qué será el resultado final? ...*

# Caos



# Esta vez con transacciones ...



**Banco de Chilly**

N° Cuenta : 7873698669  
Saldo (CLP) : 225000  
Límite de crédito : 200000  
Disponible : 425000



```
CREATE TABLE Cuenta ( ..., CHECK ( saldo_clp > -200000 ) )
```

COMPRA(Islas de Caimán,300000)

COMPRA(Noruega,400000)

```
START TRANSACTION
INSERT INTO Gasto ...
UPDATE Cuenta ...
COMMIT;
```

```
START TRANSACTION
INSERT INTO Gasto ...
UPDATE Cuenta ...
COMMIT;
```

*¿Qué será el resultado final?*

Se rechazará una de las transacciones.

# Garantías de ACID

- **Atomicidad:**
  - La ejecución de cada transacción es atómica:
    - Se realizan todas las acciones o no se realiza ninguna
- **Coherencia:**
  - Cada transacción debe preservar la integridad
    - La base de datos satisface todas las restricciones después de una transacción
- **Aislamiento (*Isolation*):**
  - Una transacción no puede afectar otra
- **Durabilidad:**
  - Una vez que haya un **COMMIT**, la base de datos debe guardar los cambios

# ACID: Atomicidad

```
CREATE TABLE Balance (  
  cuenta BIGINT PRIMARY KEY,  
  total_gasto BIGINT,  
  total_ingreso BIGINT,  
  saldo BIGINT,  
  CHECK (total_ingreso - total_gasto = saldo)  
)
```

```
START TRANSACTION  
  UPDATE Balance SET saldo=saldo-10 WHERE Cuenta=7873698669 ;  
  UPDATE Balance SET total_gasto=total_gasto+10 WHERE Cuenta=7873698669 ;  
COMMIT;
```

## Atomicidad

No se puede actualizar el saldo sin actualizar el gasto directamente después.  
(Si alguna actualización falla, ambas fallan.)

# ACID: Coherencia (Consistencia)

```
CREATE TABLE Balance (  
  cuenta BIGINT PRIMARY KEY,  
  total_gasto BIGINT,  
  total_ingreso BIGINT,  
  saldo BIGINT,  
  CHECK (total_ingreso - total_gasto = saldo)  
)
```

START TRANSACTION

```
UPDATE Balance SET saldo=saldo-100 WHERE Cuenta=7873698669 ;
```

```
UPDATE Balance SET total_gasto=total_gasto+10 WHERE Cuenta=7873698669 ;
```

```
COMMIT;
```



## Coherencia

Si el resultado de la transacción no satisface todas las restricciones, fallará.

# ACID: Aislamiento (*Isolation*)

```
CREATE TABLE Balance (  
  cuenta BIGINT PRIMARY KEY,  
  total_gasto BIGINT,  
  total_ingreso BIGINT,  
  saldo BIGINT,  
  CHECK (total_ingreso - total_gasto = saldo) DEFERRABLE INITIALLY DEFERRED  
)
```

```
START TRANSACTION T1  
UPDATE Balance  
  SET saldo=saldo-10 (1)  
  WHERE Cuenta=7873698669 ;  
UPDATE Balance  
  SET total_gasto=total_gasto+100  
  WHERE Cuenta=7873698669 ; (3)  
COMMIT;  (4) ROLLBACK;
```

```
START TRANSACTION T2  
UPDATE Balance  
  SET saldo=saldo+100 (2)  
  WHERE Cuenta=7873698669 ;  
UPDATE Balance  
  SET total_ingreso=total_ingreso+100  
  WHERE Cuenta=7873698669 ; (5)  
COMMIT; (6)
```

## Aislamiento

Una transacción no puede interferir con otra transacción.

En (4), hay que tener cuidado con el ROLLBACK: no se puede restaurar el valor de saldo antes del paso (1) porque el valor ya fue cambiado por (2).

# ACID: Durabilidad

```
CREATE TABLE Balance (  
  cuenta BIGINT PRIMARY KEY,  
  total_gasto BIGINT,  
  total_ingreso BIGINT,  
  saldo BIGINT,  
  CHECK (total_ingreso - total_gasto = saldo)  
)
```

```
START TRANSACTION  
  UPDATE Balance SET saldo=saldo-10 WHERE Cuenta=7873698669 ;  
  UPDATE Balance SET total_gasto=total_gasto+10 WHERE Cuenta=7873698669 ;  
COMMIT;
```



## Durabilidad

Una vez que haya un **COMMIT** exitoso, se persisten los cambios.

(Normalmente la persistencia aquí significa en el disco duro. Sin persistencia, en el caso de que la máquina falle y toda la evidencia de los cambios esté en la memoria principal, el sistema de base de datos **olvidará los cambios silenciosamente**.)

# ¿Cuándo **no** tenemos ACID?

- **Atomicidad:**
  - × Una transacción se ejecuta solo a medias pero afecta el estado de la base de datos
- **Coherencia:**
  - × Al ejecutar la transacción, la base de datos no satisface las restricciones de integridad
- **Aislamiento (*Isolation*):**
  - × El resultado final de dos transacciones no es equivalente a ejecutar cada transacción en serie
- **Durabilidad:**
  - × La base de datos se actualiza momentáneamente y luego vuelve al estado anterior.

ACID implementado por muchos sistemas



# ACID implementado por muchos sistemas

- Registros

- Se escriben todas las operaciones al disco duro
- Permiten recordar y deshacer/rehacer operaciones

- Bloqueos

- Cuando una transacción **actualice** x (sea una fila, una tabla, etc.), adquiere un **bloqueo exclusivo** sobre x tal que tenga acceso exclusivo a x
- Cuando una transacción **lea** x (sea una fila, una tabla, etc.), adquiere un **bloqueo no exclusivo** sobre x tal que otras transacciones puedan leer x pero no puedan actualizar x

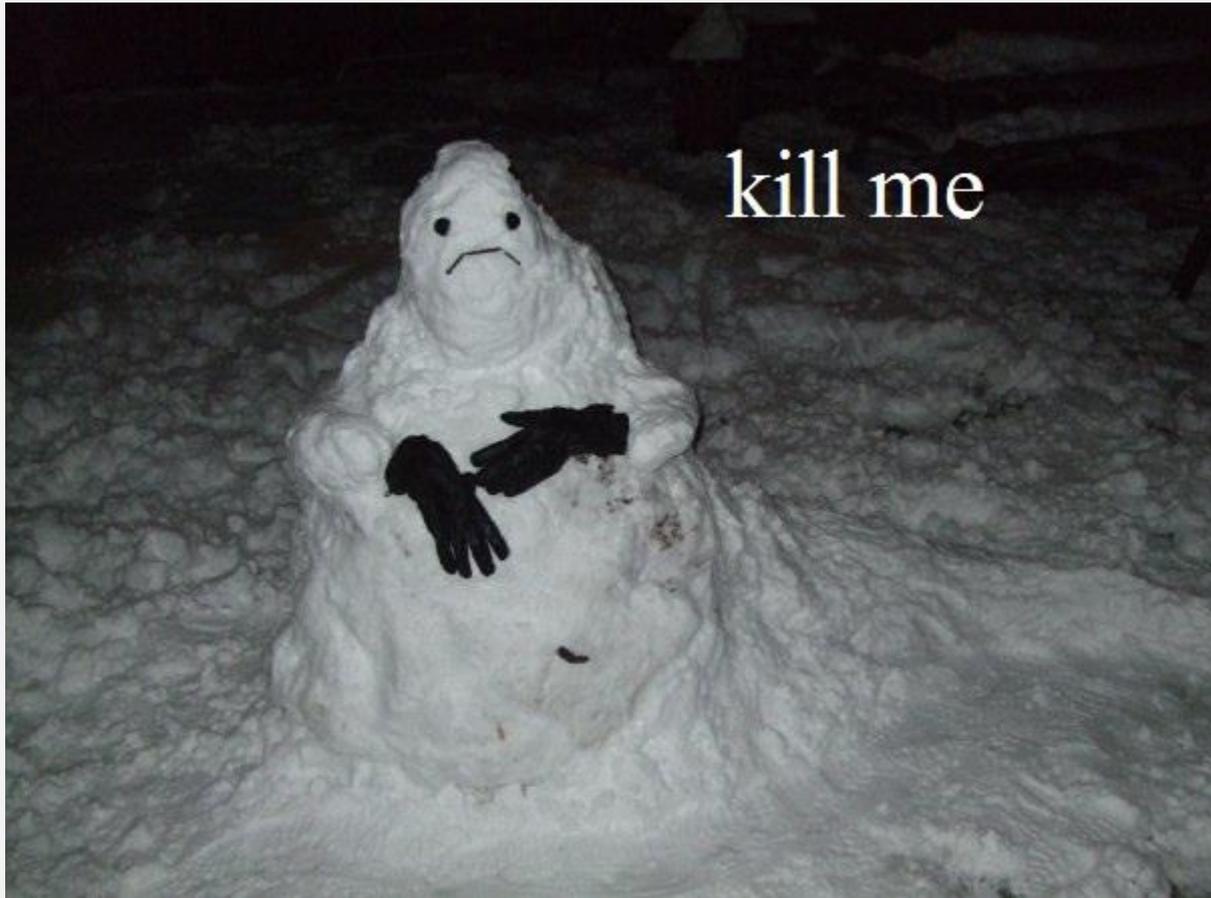
Entonces con las garantías de ACID ...



... todo está tranquilo.

\* asumiendo que no hay errores de usuario, hardware o software

... pero si uno tiene que *implementar* ACID



... es más difícil ...

# Modelando una transacción

**LEER**( $X$ ): leer un objeto  $X$  de la base de datos a memoria principal

**ESCRIBIR**( $X$ ): escribir un objeto de memoria principal a la base de datos

Un objeto  $X$  puede ser un valor, una fila, una tabla ...

*Ejemplo: Una transferencia bancaria ...*

**LEER**( $A$ )

$A \leftarrow A - 100$

**ESCRIBIR**( $A$ ) 

**LEER**( $B$ )

$B \leftarrow B + 100$

**ESCRIBIR**( $B$ ) 

T

← Dejamos el **COMMIT** implícito al final de la transacción.

# ¿Cuándo **no** tenemos ACID?

- **Atomicidad:**
  - × Una transacción se ejecuta solo a medias pero afecta el estado de la base de datos
- **Coherencia:**
  - × Al ejecutar la transacción, la base de datos no satisface las restricciones de integridad
- **Durabilidad:**
  - × La base de datos se actualiza momentáneamente y luego vuelve al estado anterior.

# Modelando una transacción

- **Atomicidad:**
  - × Una transacción se ejecuta solo a medias pero afecta el estado de la base de datos

*Ejemplo: Una transferencia bancaria ...*

```
LEER(A) T  
A ← A - 100  
ESCRIBIR(A)   
LEER(B)  
B ← B + 100  
ESCRIBIR(B) 
```

No se pueden realizar dos escrituras *exactamente* al mismo tiempo

IMPLEMENTANDO ACID:  
REGISTROS (*LOGGING*)

# Mantener un registro de la transacción

*Ejemplo: Una transferencia bancaria ...*

LEER( $A$ )

$A \leftarrow A - 100$

ESCRIBIR( $A$ ) 

LEER( $B$ )

$B \leftarrow B + 100$

ESCRIBIR( $B$ ) 

T

T begin

T leer A 400

T escribir A 300

T leer B 200

T escribir B 300

T commit



./registro.log

# Si hay un problema, revertir el registro

La información en el registro debe bastar para revertir el estado de la base de datos sin ambigüedad

*Ejemplo: Una transferencia bancaria ...*

LEER( $A$ ) T  
 $A \leftarrow A - 100$   
ESCRIBIR( $A$ )   
LEER( $B$ )  
 $B \leftarrow B + 100$   
ESCRIBIR( $B$ ) 



```
T begin
T leer A 400
T escribir A 300
T leer B 200
T escribir B 300
T rollback
```



./registro.log

# Registros ayudan con ...

- **Atomicidad:** 
  - × Una transacción se ejecuta solo a medias pero afecta el estado de la base de datos
- **Coherencia:** 
  - × Al ejecutar la transacción, la base de datos no satisface las restricciones de integridad
- **Aislamiento (*Isolation*):** 
  - × El resultado final de dos transacciones no es equivalente a ejecutar cada transacción en serie
- **Durabilidad:** 
  - × La base de datos se actualiza momentáneamente y luego vuelve al estado anterior.

¿Qué problemas podemos evitar con alguna forma de registro? ...

# Concurrencia/Aislamiento: un problema abierto

```
CREATE TABLE Balance (  
  cuenta INTEGER PRIMARY KEY,  
  total_gasto BIGINT,  
  total_ingreso BIGINT,  
  saldo BIGINT,  
  CHECK (total_ingreso - total_gasto = saldo)  
)
```

```
START TRANSACTION T1  
UPDATE Balance  
  SET saldo=saldo-10 (1)  
  WHERE Cuenta=7873698669 ;  
UPDATE Balance  
  SET total_gasto=total_gasto+100  
  WHERE Cuenta=7873698669 ; (3) ❌  
COMMIT; (4) ROLLBACK;
```

```
START TRANSACTION T2  
UPDATE Balance  
  SET saldo=saldo+100 (2)  
  WHERE Cuenta=7873698669 ;  
UPDATE Balance  
  SET total_ingreso=total_ingreso+100  
  WHERE Cuenta=7873698669 ; (5)  
COMMIT; (6)
```

## Aislamiento

Una transacción no puede interferir con otra transacción.

En (4), hay que tener cuidado con el ROLLBACK: no se puede restaurar el valor de saldo antes del paso (1) porque el valor ya fue cambiado por (2).

Preguntas?

