

Prácticas de

Administración de Bases de Datos

Grado en Ingeniería Informática

**PRÁCTICA 6**

Gestión y control de la concurrencia (Transacciones)

**Ejercicios**

1. Inicia una sesión y ejecuta las siguientes sentencias. Observa el resultado.



**CREATE TABLE T (id INT PRIMARY KEY, s VARCHAR(30), si NUMBER(5)); INSERT INTO T (id, s) VALUES (1, 'first');**

**COMMIT;**

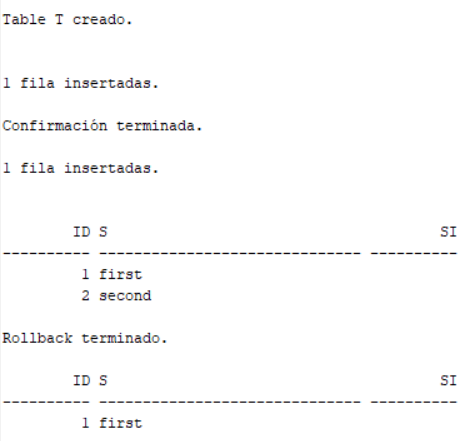
**INSERT INTO T (id, s) VALUES (2, 'second'); SELECT \* FROM T;**

**ROLLBACK;**

**SELECT \* FROM T;**

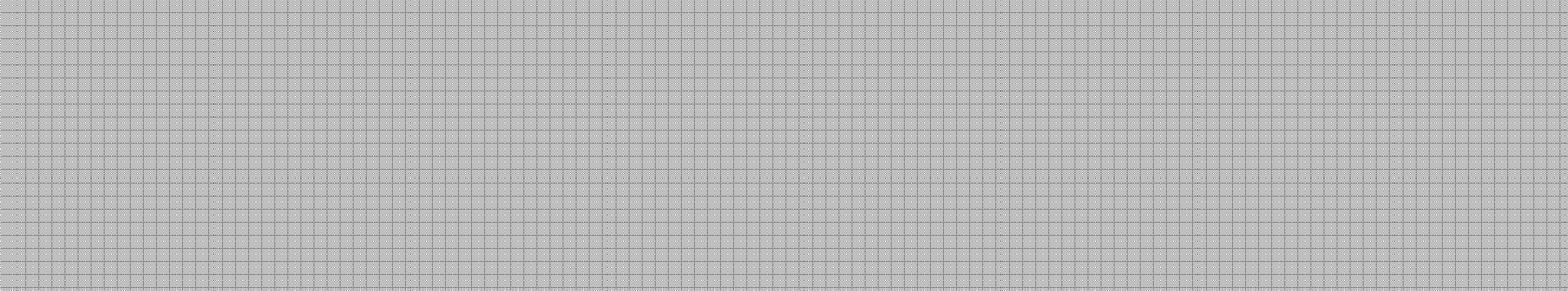
Se crea la tabla T en el acto sin necesidad de commit (commit implícito). Mientras, en el caso de sentencias INSERT, se necesita hacer commit del cambio introducido (commit explícito).

Se puede observar como al ejecutar las sentencias, cuando se hace rollback del segundo insert (2, ‘second’), este deja de aparecer en la tabla T.



Añade las siguientes sentencias. ¿Cuál ha sido el efecto de ROLLBACK?

¿Ocurrió algún COMMIT implícito?



**INSERT INTO T (id, s) VALUES (2, 'Will this be committed?'); CREATE TABLE T2 (id INT);**

**INSERT INTO T2 (id) VALUES (1); SELECT \* FROM T2;**

**ROLLBACK;**

Las sentencias proporcionan la siguiente salida:

Imagen que contiene Gráfico

Descripción generada automáticamente

El efecto de rollback ha sido no hacer commit de la sentencia INSERT INTO T2 (id) VALUES (1); Esto es debido a que al realizarse la creación de la tabla T2, esta sentencia provoca un commit implícito tanto antes como después.

Todas las transacciones que contengan sentencias DML (INSERT, UPDATE, etc.) son persistidas en cuanto se ejecuta una sentencia DDL (CREATE, etc.). Posteriormente la propia sentencia DDL se trata como una transacción nueva, persistiéndose al final de la operación.

Si ejecutamos un SELECT \* FROM T2 observamos que la única sentencia no persistida es la comentada con anterioridad:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Con SELECT \* FROM T:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Añade las siguientes sentencias. ¿Qué puede observarse?



**DELETE FROM T WHERE id > 1 ; COMMIT;**

**INSERT INTO T (id, s) VALUES (2, 'The test starts by this'); SELECT (1/0) FROM DUAL;**

**UPDATE T SET s = 'foo' WHERE id = 9999; DELETE FROM T WHERE id = 7777;**

**COMMIT;**

**SELECT \* FROM T;**

Se obtiene la siguiente salida:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

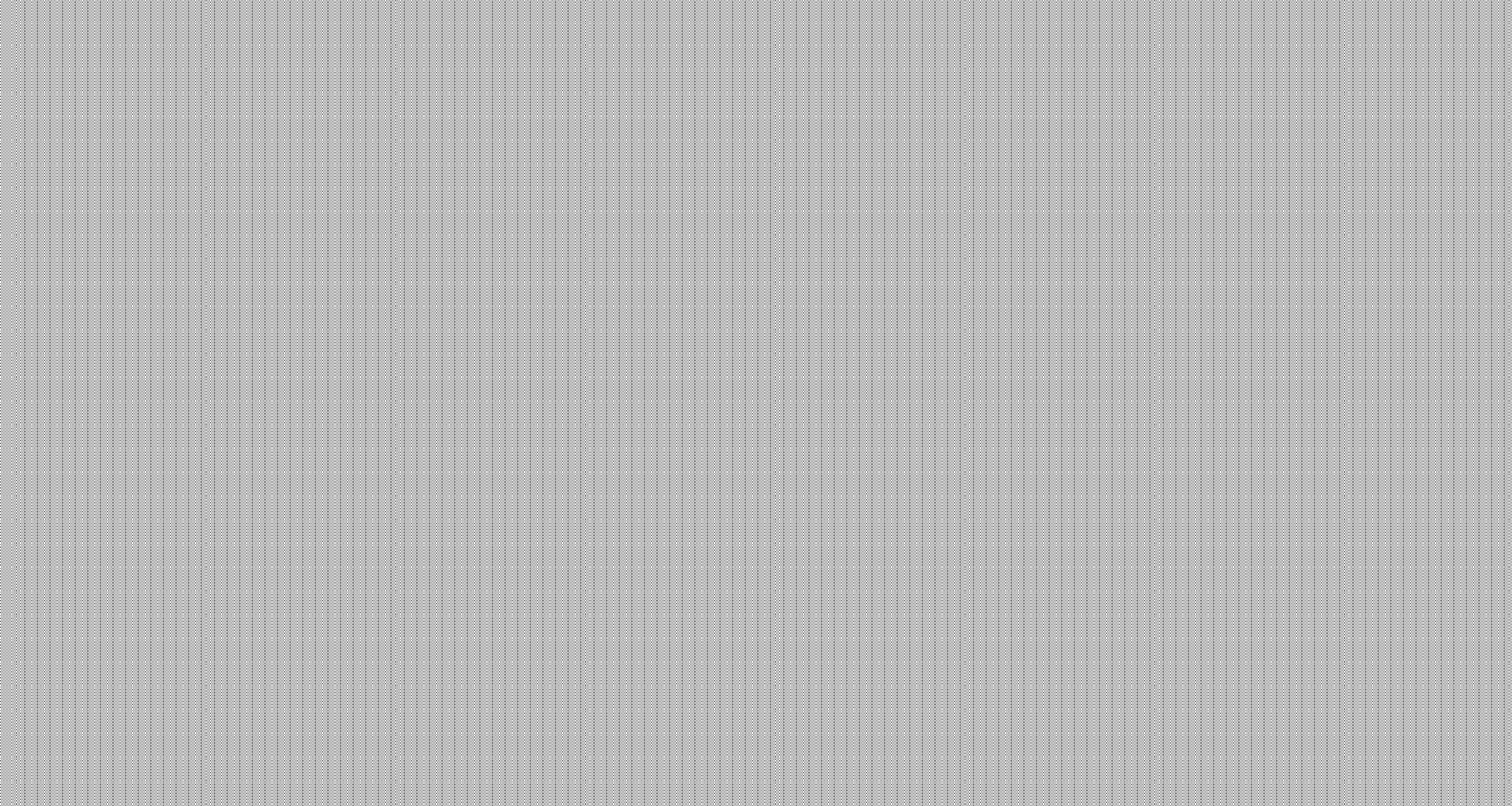
Descripción generada automáticamente

La sentencia SELECT (1/0) FROM DUAL falla, pero al ser tratada de manera atómica por la BDD Oracle, es tan solo esta única sentencia la que no se persiste en base de datos o no se ejecuta, sin causar la perdida de trabajo de sentencias anteriores. Lo mismos es aplicable a las sentencias UPDATE T SET s = ‘foo’ WHERE id = 9999 y DELETE FROM T WHERE id = 7777 que fallan al no haber ningún elemento con id 9999 en la tabla T, ni ninguno con id 7777.

Cuando el SELECT (1/0) FROM DUAL falla, esto deja abierta la posibilidad de que para la sentencia que la precede, el usuario realice un commit o un rollback según se desee.

Es por ello que debido al commit final, se persiste la sentencia INSERT INTO T (id, s) VALUES (2, 'The test starts by this');

Ahora añade las siguientes sentencias. ¿Qué se observa?



**UPDATE T SET s = 'Otro test' WHERE id = 2; INSERT INTO T (id, s)**

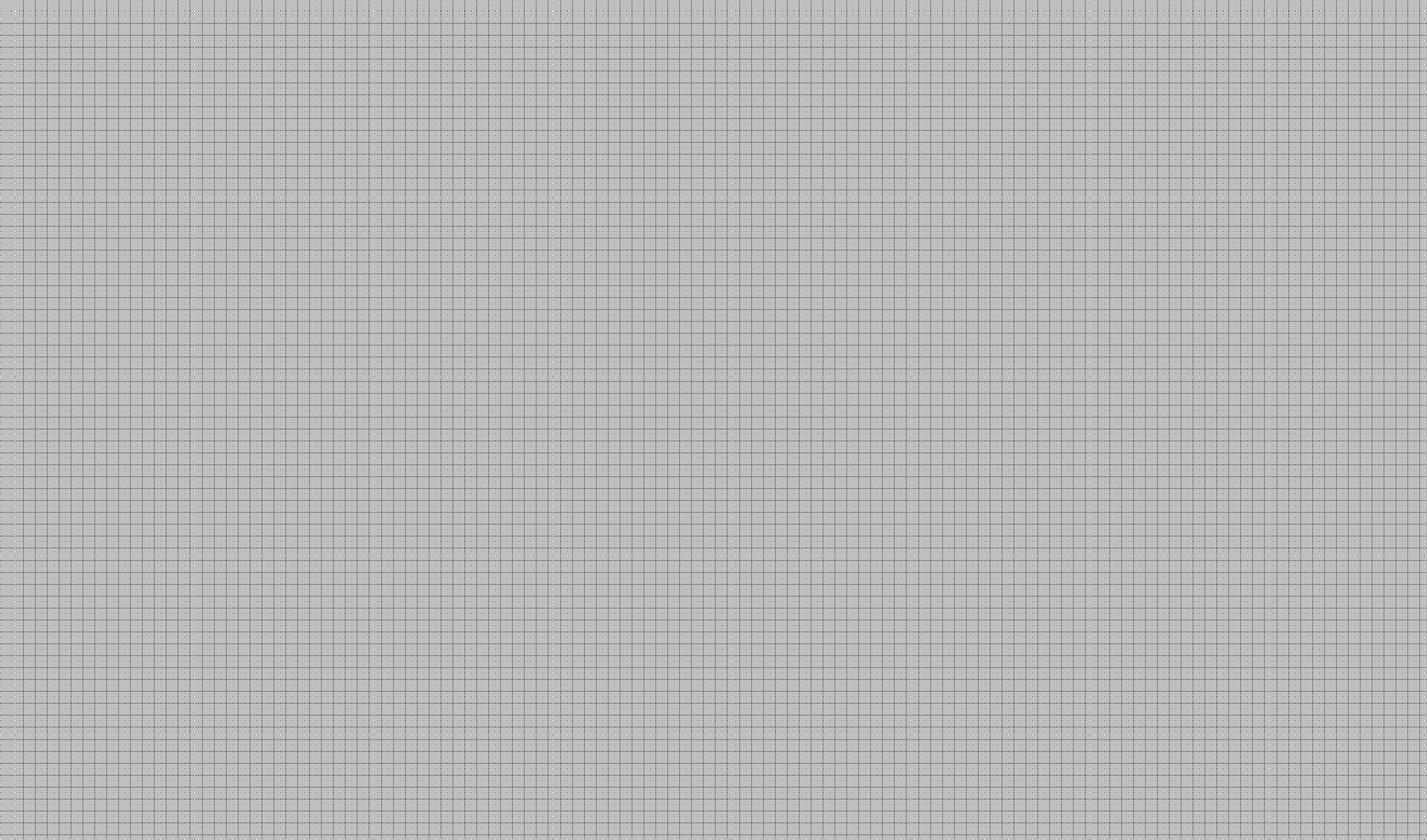
**VALUES (2, 'Soy un duplicado'); INSERT INTO T (id, s)**

**VALUES (3, 'Intentando insertar una cadena muy larga'); INSERT INTO T (id, s, si)**

**VALUES (4, 'Se puede un 123456?', 123456); COMMIT;**

**SELECT \* FROM T;**

1. Ejecuta las siguientes sentencias. Entendiendo que se trata de transferencias de una cuenta a otra.. ¿hay algún problema en lo que se expresa a continuación? ¿Cómo podría solucionarse?



**DROP TABLE Accounts; CREATE TABLE Accounts (**

**acctID INTEGER NOT NULL PRIMARY KEY,**

**balance INTEGER NOT NULL**

**CONSTRAINT unloanable\_account CHECK (balance >= 0)**

**);**

**INSERT INTO Accounts (acctID, balance) VALUES (101, 1000); INSERT INTO Accounts (acctID, balance) VALUES (202, 2000); COMMIT;**

**SELECT \* FROM Accounts;**

**UPDATE Accounts SET balance = balance - 100 WHERE acctID = 101; UPDATE Accounts SET balance = balance + 100 WHERE acctID = 202; COMMIT;**

**SELECT \* FROM Accounts;**

**UPDATE Accounts SET balance = balance - 2000 WHERE acctID = 101; UPDATE Accounts SET balance = balance + 2000 WHERE acctID = 202; COMMIT;**

**SELECT \* FROM Accounts;**

1. Abre dos sesiones en dos ventanas SQL\*Plus, correspondientes a los clientes A y

B. Ejecuta las correspondientes sentencias para cada cliente. ¿Observas alguna anomalía? ¿Alguna sugerencia?



**-- En cualquier cliente:**

**DROP TABLE Accounts; CREATE TABLE Accounts (**

**acctID INTEGER NOT NULL PRIMARY KEY,**

**balance INTEGER NOT NULL**

**CONSTRAINT unloanable\_account CHECK (balance >= 0)**

**);**

**INSERT INTO Accounts (acctID, balance) VALUES (101, 1000); INSERT INTO Accounts (acctID, balance) VALUES (202, 2000);**

**COMMIT;**

**SELECT \* FROM Accounts;**

**-- Cliente A comienza:**

**SELECT balance FROM Accounts WHERE acctID = 101;**

**-- se simula la lectura de 1000 en una variable**

**-- Cliente B comienza:**

**SELECT balance FROM Accounts WHERE acctID = 101;**

**-- se simula la lectura de 1000 en una variable**

**-- Cliente A continúa:**

**UPDATE Accounts SET balance = (1000 - 200) WHERE acctID = 101;**

**-- Cliente B continúa:**

**UPDATE Accounts SET balance = (1000 - 500) WHERE acctID = 101;**

**-- Cliente A continúa:**

**SELECT acctID, balance FROM Accounts WHERE acctID = 101; COMMIT;**

**-- Cliente B continúa:**

**SELECT acctID, balance FROM Accounts WHERE acctID = 101; COMMIT;**

**Ahora ejecuta las siguientes sentencias. ¿Qué ocurre?**



**-- En cualquier cliente:**

**DELETE FROM Accounts;**

**INSERT INTO Accounts (acctID,balance) VALUES (101,1000); INSERT INTO Accounts (acctID,balance) VALUES (202,2000); COMMIT;**

**-- El cliente A transfiere 100 euros de la cuenta 101 a la 202**

**-- El cliente B transfiere 200 euros de la cuenta 202 a la 101**

**-- Cliente A:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UPDATE Accounts** | **SET** | **balance** | **=** | **balance** | **-** | **100** | **WHERE** | **acctID** | **=** | **101;** |
| **-- Cliente B: UPDATE Accounts** | **SET** | **balance** | **=** | **balance** | **-** | **200** | **WHERE** | **acctID** | **=** | **202;** |
| **-- Cliente A: UPDATE Accounts** | **SET** | **balance** | **=** | **balance** | **+** | **100** | **WHERE** | **acctID** | **=** | **202;** |
| **-- Cliente B: UPDATE Accounts** | **SET** | **balance** | **=** | **balance** | **+** | **200** | **WHERE** | **acctID** | **=** | **101;** |

**Ahora ejecuta las siguientes sentencias. ¿Observas alguna anomalía?**



**DELETE FROM Accounts;**

**INSERT INTO Accounts (acctID,balance) VALUES (101,1000); INSERT INTO Accounts (acctID,balance) VALUES (202,2000); COMMIT;**

**-- Cliente A:**

**SELECT \* FROM Accounts WHERE balance > 500;**

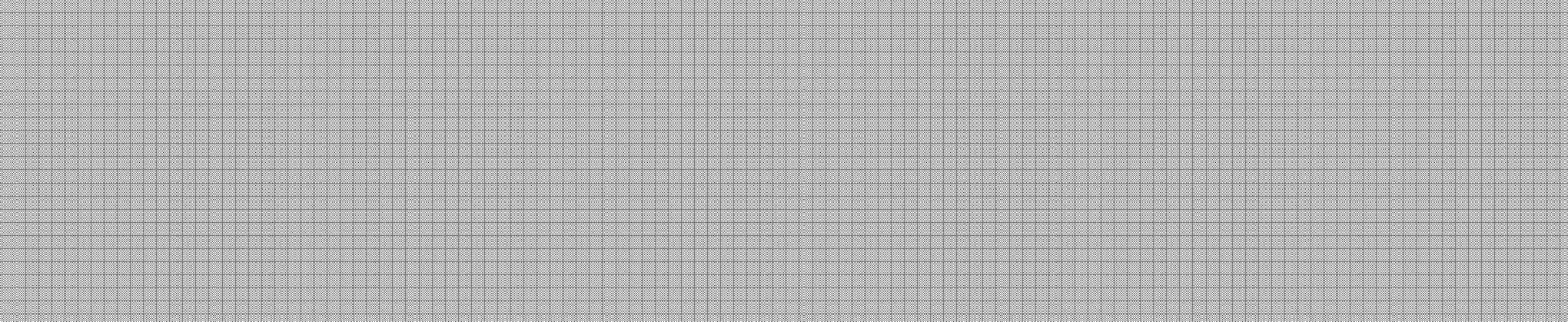
**-- Cliente B:**

**UPDATE Accounts SET balance = balance - 500 WHERE acctID = 101; UPDATE Accounts SET balance = balance + 500 WHERE acctID = 202; COMMIT;**

**-- Cliente A:**

**SELECT \* FROM Accounts WHERE balance > 500; COMMIT;**

1. Ejecuta las siguientes sentencias y analiza su resultado.



**DELETE FROM Accounts;**

**INSERT INTO Accounts (acctID,balance) VALUES (101,1000); INSERT INTO Accounts (acctID,balance) VALUES (202,2000); COMMIT;**

**SELECT \* FROM Accounts;**

**A continuación se ejecutan las siguientes instrucciones, Obtenga conclusiones..**



**SELECT NAME, STATUS FROM V$TRANSACTION; SET TRANSACTION NAME 'ingresos\_1';**

**UPDATE Accounts SET balance = balance + 250 WHERE acctID = 101; SAVEPOINT tras\_ingreso\_250;**

**SELECT NAME, STATUS FROM V$TRANSACTION;**

**UPDATE Accounts SET balance = balance + 100 WHERE acctID = 202; SAVEPOINT tras\_ingreso\_100;**

**ROLLBACK TO SAVEPOINT tras\_ingreso\_250; SELECT \* FROM Accounts;**

**SELECT NAME, STATUS FROM V$TRANSACTION;**

**UPDATE Accounts SET balance = balance + 300 WHERE acctID = 202; ROLLBACK;**

**SELECT \* FROM Accounts;**

**SELECT NAME, STATUS FROM V$TRANSACTION;**

**SET TRANSACTION NAME 'ingresos\_2'; SELECT NAME, STATUS FROM V$TRANSACTION;**

**UPDATE Accounts SET balance = balance + 150 WHERE acctID = 101; UPDATE Accounts SET balance = balance + 200 WHERE acctID = 202;**

**SELECT NAME, STATUS FROM V$TRANSACTION; COMMIT;**

**SELECT \* FROM Accounts;**

**SELECT NAME, STATUS FROM V$TRANSACTION;**

**A continuación se ejecutan las siguientes instrucciones, ¿se produce algún problema?. Obtenga conclusiones..**



**SET TRANSACTION NAME 'uno';**

**UPDATE Accounts SET balance = balance + 250 WHERE acctID = 101; SET TRANSACTION NAME 'dos';**