2º Grado en Ingeniería Informática Facultad de Informática

Asignatura: **Bases de Datos**

Tema 11. Aspectos Básicos de Procesamiento de Transacciones

EJERCICIOS. Concurrencia de transacciones. Soluciones.

Objetivo

- Comprender algunos conceptos básicos relacionados con la concurrencia de transacciones en los sistemas de bases de datos, a través de la resolución de diversos ejercicios.

Modalidad

Clase de problemas.

Contenidos

Considera el siguiente esquema relacional de base de datos

EMPLEADO(nss, dni, nombre, apellido, fechanacim, ciudad, est_civil, salario, nssjefe, dep, numfamiliares)

DEPARTAMENTO(coddep, nombre, nssdire)

FAMILIAR(nssemp, numero, nombre, fechanacim, parentesco)

PROYECTO(codproy, titulo, lugar, dep) -- proyectos puestos en marcha por los departamentos

DEDICACION(empleado, proyecto, horas) -- horas que cada empleado dedica a cada proyecto

Este boletín consiste en el planteamiento y resolución de una serie de ejercicios de **concurrencia en Sistemas de Bases de datos**, relacionados con los **niveles de aislamiento** y el uso de las **técnicas basadas en bloqueos**.

---

1. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan .

T1 (READ UNCOMMITED)	T2 (READ UNCOMMITED)
	SELECT SUM(salario) suma1 FROM empleado;
UPDATE empleado SET salario = salario *0.95 WHERE nss NOT IN (SELECT empleado FROM dedicacion) GROUP BY nssemp HAVING COUNT(*) > 2);	
	SELECT SUM(salario) suma2 FROM empleado;
COMMIT;	
	SELECT SUM(salario) suma3 FROM empleado;
	…

Indique la opción correcta:

a) El valor de suma1 es igual al de suma2 y al de suma3 (suma1=suma2=suma3)

b) El valor de suma1 es diferente al de suma2, y el valor de suma2 es igual al de suma3 (suma1<>suma2=suma3).

c) El valor de suma1 es igual al de suma2, y el valor de suma2 es diferente al de suma3 (suma1=suma2<>suma3).

2. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan.

T1 (READ COMMITED)	T2 (READ COMMITED)
	SELECT SUM(salario) suma1 FROM empleado;
UPDATE empleado SET salario = salario *0.95 WHERE nss NOT IN (SELECT empleado FROM dedicacion) GROUP BY nssemp HAVING COUNT(*) > 2);	
	SELECT SUM(salario) suma2 FROM empleado;
COMMIT;	
	SELECT SUM(salario) suma3 FROM empleado;
	…

Indique la opción correcta:

a) El valor de suma1 es igual al de suma2 y al de suma3 (suma1=suma2=suma3)

b) El valor de suma1 es diferente al de suma2, y el valor de suma2 es igual al de suma3 (suma1<>suma2=suma3).

c) El valor de suma1 es igual al de suma2, y el valor de suma2 es diferente al de suma3 (suma1=suma2<>suma3).

Lo que pasa: T2 ejecuta su SELECT y al estar en el nivel READ COMMITED, los bloqueos de lectura se liberan inmediatamente. T1 puede bloquear para lectura y realizar su UPDATE, pero no libera los bloqueos. Cuando T2 trata de ejecutar el SELECT (suma2), no puede y queda en espera hasta que T1 hace el COMMIT y libera bloqueos. Así que T2 cuando ejecuta el SELECT que calcula suma2 obtiene un valor diferente al de suma1, porque lo hace con los valores nuevos modificados por el UPDATE de T1; seguidamente T2 ejecuta el SELECT que calcula suma3, que obtiene el mismo valor que suma2, porque usa los mismos valores.

3. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan.

	T1 (REPEATABLE READ)	**T2 (REPEATABLE READ)**
1		SELECT est_civil FROM empleado WHERE nss = 333;
2	UPDATE empleado SET est_civil = 'D' WHERE nss = 333;	
3		SELECT est_civil FROM empleado WHERE nss = 333;
4		COMMIT;
5	…	

Indique la opción correcta:

a) T1 no puede ejecutar el UPDATE hasta después de que T2 realiza el COMMIT, por lo que los resultados de las SELECT de los pasos 1 y 3 son idénticos.

b) T1 sí ejecuta el UPDATE en el paso 2, así que los resultados de la SELECT de los pasos 1 y 3 son diferentes.

c) T1 sí ejecuta el UPDATE en el paso 2, pero T2 no ve los cambios realizados por T1, así que suma1 y suma2 tienen el mismo valor

Lo que pasa: T2 ejecuta su SELECT y al estar en el nivel REPEATABLE READ, los bloqueos de lectura permanecen hasta el fin de T2, por lo que cuando T2 trata de ejecutar el UPDATE, no puede y queda en espera hasta que T2 finalice y libere bloqueos. Así que T2 vuelve a ejecutar el SELECT (paso 3) que obtiene el mismo valor que antes. Cuando T2 se confirma, entonces T1 puede realizar el UPDATE (paso 5).

4. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan.

	T1 (REPEATABLE READ)	**T2 (REPEATABLE READ)**
1	SELECT numero, nombre, parentesco FROM familiar WHERE nssemp = 987;	
2		INSERT INTO familiar (nssemp, numero, nombre, fechanacim, parentesco) VALUES (987, 2, 'Bonifacio', '20/08/2006', 'HIJO');
3		COMMIT;
4	SELECT numero, nombre, parentesco FROM familiar WHERE nssemp = 987;	
5	…	

Indique la opción correcta:

a) En el paso 4, el resultado de la SELECT de T1 es diferente al del paso 1. En el paso 4 la SELECT muestra también las filas insertadas por T2 en el paso 2.

b) Los resultados de la SELECT de los pasos 1 y 4 son idénticos. T1 no ve los cambios (inserciones) hechos por T2.

c) T2 no puede ejecutar el INSERT hasta que T1 finalice, por lo que se queda a la espera y no se ejecutan ni el paso 2 ni el 3.

Lo que pasa: T1 ejecuta su SELECT y se bloquean las filas de FAMILIAR con el nssemp 987. T2 inserta varias filas en FAMILIAR, nuevas. Estas filas no están bloqueadas por T1 lógicamente, por lo que se insertan sin problema.
T2 se confirma y libera bloqueos.
Cuando T1 vuelve a ejecutar la SELECT (paso 4), se ven las nuevas filas en FAMILIAR: son filas fantasmas.

5. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan.

	T1 (READ UNCOMMITED)	**T2 (READ UNCOMMITED)**
1	SELECT numfamiliares -- *tiene 2* FROM empleado WHERE nss = 123;	
2	UPDATE empleado SET numfamiliares = numfamiliares + 1 WHERE nss = 123;	
3		SELECT numfamiliares FROM empleado WHERE nss = 123;
4		UPDATE empleado SET numfamiliares = numfamiliares + 3 WHERE nss = 123;
5	ROLLBACK;	
6		COMMIT;

Considera que inicialmente (en el paso 1) el valor de "numfamiliares" es 2 para el empleado con nss 123. ¿Cuál es el valor de "numfamiliares" tras la ejecución del paso 6? Indique la opción correcta:

a) T2 en el paso 3 sí ve los cambios del UPDATE realizado por T1 en el paso 2. Así que el valor es 6.

b) T2 ejecuta su UPDATE en el paso 3 sin ver el cambio realizado por T1 en el paso 2. Por esto, el valor es 5.

c) Al deshacerse las operaciones de T1 en el paso 4, el "numfamiliares" vuelve a su valor original 2.

Lo que pasa: T1 ejecuta su SELECT y su UPDATE sin problemas. Al estar en modo READ UNCOMMITED, los bloqueos se liberan inmediatamente tras cada sentencia, así que en el paso 3 T2 lee el nº de familiares para el 123 CON el cambio hecho por T1, por lo que T2 lee el valor de "numfamiliares+1" que modificó T1. Luego, paso 4, T2 modifica el valor sin problema a "numfamiliares+1+3". Cuando T1 hace ROLLBACK, deshace su operación sobre numfamiliares, pero cuando T2 se confirma, el valor 6 que deja en numfamiliares es ERRÓNEO debido a que ha tenido en cuenta el +1 que en realidad nunca se hizo (al deshacer T1).

Así que si al principio vale 2, al final es 2+1+3=6.

6. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan.

	T1 (READ COMMITED)	**T2 (READ COMMITED)**
1	SELECT numfamiliares -- *tiene 2* FROM empleado WHERE nss IN (123, 111);	
2		SELECT numfamiliares FROM empleado WHERE nss = 123;
3		UPDATE empleado SET numfamiliares = numfamiliares + 3 WHERE nss = 123;
4	UPDATE empleado SET numfamiliares = numfamiliares + 1 WHERE nss IN (123, 111);	
5		COMMIT;
6	COMMIT;	

Considera que inicialmente (en el paso 1) el valor de "numfamiliares" es 2 para el empleado con nss 123. ¿Cuál es el valor de "numfamiliares" tras la ejecución del paso 6? Indique la opción correcta:
a) El valor es 5. [T2 no ve los cambios hechos por T1, y la actualización de T2 sobrescribe la de T1]
b) El valor es 3. [T1 no ve los cambios hechos por T2, y la actualización de T1 sobrescribe la de T2]
c) El valor es 6. [El UPDATE de T1 se realiza después de que T2 se confirme]

Lo que pasa: T1 ejecuta su SELECT sin problemas. Al estar en modo READ COMMITED, se liberan los bloqueos y T2 puede ejecutar su SELECT en el paso 2 sin problemas. También puede bloquear para escritura la fila del empleado 123 y ejecutar su UPDATE en el paso3, por lo que numfamiliares = 2+3 y se mantiene el bloqueo.
T1 no puede ejecutar su UPDATE en el paso 4, porque la fila está bloqueada. Queda en espera.
Cuando T2 se confirma, libera los bloqueos y entonces T1 puede bloquear para escritura y hacer el UPDATE, sumando 1 más. Así que, si al principio vale 2, al final es 2+3+1=6

7. Considera la ejecución de las transacciones que aparecen en la siguiente planificación y ten en cuenta el nivel de aislamiento en el que se ejecutan.

	T1 (READ COMMITED)	T2 (SERIALIZABLE)
1	SELECT nombre, est_civil FROM empleado WHERE nss = 111;	
2	UPDATE empleado SET est_civil = 'C' WHERE nss = 111;	
3		UPDATE empleado SET est_civil = 'P' WHERE nss = 111;
4	COMMIT;	
5		…

Indique la opción correcta:

a) T2 no puede ejecutar su UPDATE en el paso 3, y debe esperar a que T1 finalice. Cuando T2 lo ejecuta (en el paso 5), el valor de "est_civil" para el empleado 111 se establece a 'P'.

b) T2 no puede ejecutar el UPDATE en el paso 3 y cuando T1 realiza su COMMIT ocurre un error de concurrencia: se incumple el nivel de aislamiento *serializable* para T2. En el paso 5, el valor de "est_civil" para el empleado 111 queda a 'C'.

c) T2 ejecuta sin problema su UPDATE en el paso 3. El valor de "est_civil" para el empleado 111 es 'C', puesto que T2 aún no se ha confirmado y no ha guardado sus cambios en disco.

Lo que pasa: T1 ejecuta su SELECT y su UPDATE sin problemas y mantiene su bloqueo. En el paso 3, T2 trata de modificar el valor, y se queda esperando a que T1 se confirme porque se trata de una fila bloqueada por T1 (la fila 111: conflicto de escritura). Cuando T1 se confirma, entonces el UPDATE de T2 NO se puede realizar, porque T1 confirmó sus cambios sobre la misma fila DESPUES de que T2 comenzara. Así que se devuelve un error para T2. Esto es debido a que T2 se ejecuta en modo SERIALIZABLE.