**Universidad Autónoma Gabriel René Moreno**

**Facultad de Ciencias de la Computación.**

**Carrera de Ingeniería Informática**



**TRABAJO PRÁCTICO**

**Materia**

**NOMBRE: PAMELA LEONESA MARTINEZ ROMERO**

**“BASE DE DATOS II”**

**TEMA 4**

Docente:

***Ing. Ubaldo Perez Ferreira***

Santa Cruz de la Sierra, enero 2015

# **Trabajo Practico 1 - Tema 4**

**1. Cual la diferencia entre un SGBD Mono Usuario y un SGBD Multiusuario**

Un SGBD multiusuario permite que más de un usuario utilice el sistema a la vez, en caso contrario se llama monousuario. En un SGBD multiusuario, los datos almacenados en la BD son accedidos y modificados de forma concurrente por los programas de los usuarios.

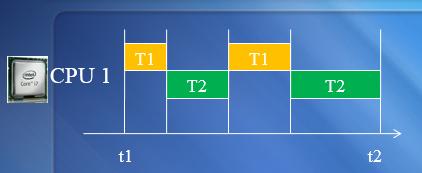
**2. Explique el concepto de Multiprogramación en los Sistema Operativos**

Se denomina multiprogramación a una técnica por la que dos o más procesos pueden alojarse en la memoria principal y ser ejecutados concurrentemente por el procesador o CPU.

La multiprogramación es la técnica que permite que dos o más programas ocupen la misma unidad de memoria principal y que sean ejecutados al mismo tiempo.

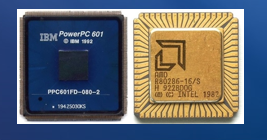
**3. Explique los conceptos de Concurrencia Intercalada y Paralela**

Concurrencia Intercalada:

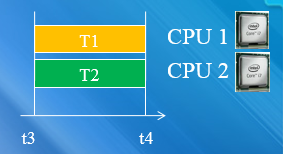


Si sólo hay **una** **CPU**, el **Sistema Operativo de multiprogramación** reparte el tiempo de CPU entre las transacciones:   
**ejecución concurrente intercalada**

* + modelo que supondremos



Concurrencia paralela:



Si el equipo tiene **varias CPU**, es posible el procesamiento simultáneo (paralelo) de transacciones.

**4. Explique el concepto de Transacciones Concurrentes**

Son varias transacciones introducidas por usuarios, que se ejecutan de manera concurrente, pueden leer/modificar el mismo elemento de datos X almacenados en la Base de Datos

Una transacción es una unidad de programa que accesa y posiblemente actualiza varios elementos de datos.

**5. Dadas dos transacciones T1 y T2, explique las dos formas posible que tiene el SGBD para ejecutar ambas transacciones.**

T1

T2

leer (X);

X:= X-N;

escribir(X);

leer(Y);

Y:=Y+N;

escribir(Y);

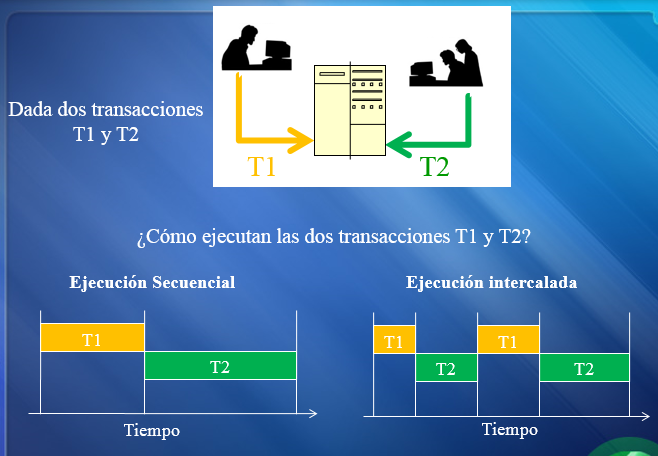
leer(X);

X:= X+M;

escribir(X);

Leer (X): Transfiere el dato X de la base de datos a una memoria intermedia local perteneciente a la transacción que ejecuta la operación leer.

Escribir(X): Transfiere el dato X de la memoria intermedia local a la base de la transacción que ejecuta la operación Escribir.



**6. Cites algunas razones por la cual es importante la ejecución concurrente de dos o más transacciones.**

* **Razones para permitir la concurrencia:**
  + Aumentar la productividad: número de transacciones ejecutadas por minuto.
  + Aumentar la utilización de la CPU (menos tiempo ocioso) y Control del disco.
  + Reducir el tiempo medio de respuesta de transacciones (las ‘pequeñas’ no esperan a las ‘grandes’).

***Ejecuciones Concurrentes***

* Una ejecución concurrente provoca conflictos
* Es más sencillo exigir que las transacciones se ejecuten secuencialmente.

**7. Las transacciones concurrentes deben controlar dos operaciones. ¿Cuáles son?**

**Leer(X): Transfiere** el dato X de la base de datos a una memoria intermedia local perteneciente a la transacción que ejecuta la operación leer.

**Escribir(X): Transfiere** el dato X de la memoria intermedia local a la base de la transacción que ejecuta la operación Escribir.

Ejemplo: Sea Ti una transacción para transferir 10.000 Bs. de la cuenta ‘A’ a la Cuenta ‘B’. Se puede definir dicha transacción como:

Ti: 1. Leer(A);

2. A= A – 10.000;

3. Escribir(A);

4. leer (B);

5. B= B + 10.000;

6. Escribir (B);

**8. Defina el concepto de lectura de valor Fantasma en las Transacciones**

Una *lectura fantasma* ocurre cuando, durante una transacción, se ejecutan dos consultas idénticas, y los resultados de la segunda son distintos de los de la primera.

Esto puede ocurrir cuando no se realizan *bloqueos de rango* al realizar una operación [*SELECT*](http://es.wikipedia.org/w/index.php?title=SELECT&action=edit&redlink=1)*... WHERE*.

La anomalía de las *lecturas fantasma* es una caso particular de las *lecturas no repetibles* cuando la transacción 1 repite una consulta acotada en rango [*SELECT*](http://es.wikipedia.org/w/index.php?title=SELECT&action=edit&redlink=1)*... WHERE* y, entre ambas operaciones la transacción 2 crea (i.e. [INSERT](http://es.wikipedia.org/w/index.php?title=INSERT&action=edit&redlink=1)) nuevas filas (en la misma tabla) que entran dentro de esa cláusula *WHERE*.

|  |  |
| --- | --- |
| **Transacción 1** | **Transacción 2** |
| */\* Consulta 1 \*/*  **SELECT** \* **FROM** usuarios  **WHERE** edad **BETWEEN** 10 **AND** 30; |  |
|  | */\* Consulta 2 \*/*  **INSERT** **INTO** usuarios **VALUES** ( 3, 'Bob', 27 );  COMMIT; |
| */\* Consulta 1 \*/*  **SELECT** \* **FROM** usuarios  **WHERE** edad **BETWEEN** 10 **AND** 30; |  |

Nótese que la transacción 1 ejecuta la misma consulta dos veces. Si se mantuviera el mayor nivel de aislamiento, los resultados de ambas consultas coincidirían, de hecho es lo que se pide a una base de datos operando al nivel de aislamiento SERIALIZABLE. Sin embargo, a niveles de aislamientos menores, pueden obtenerse resultados distintos.

Al nivel de aislamiento SERIALIZABLE, la consulta 1 bloquearía todos los registros con edades comprendidas entre 10 y 30, de modo que la consulta 2 quedaría bloqueada hasta que se cometiera la transacción 1. En modo REPEATABLE READ, el rango de 10 a 30 no se bloquearía, permitiendo la inserción de modo que la segunda ejecución de la consulta 1 sí incluirá la nueva fila en sus resultados.

**9. Defina el concepto de lectura de valor Sucio en las Transacciones**

Una lectura sucia ocurre cuando se le permite a una transacción la lectura de una fila que ha sido modificada por otra transacción concurrente pero todavía no ha sido cometida.

Las lecturas sucias funcionan de modo similar a las lecturas no repetibles; sin embargo la segunda transacción no necesita ser cometida para que la primera dé un resultado diferente. Lo único que se puede prevenir en el nivel de aislamiento LECTURAS NO COMETIDAS es que las actualizaciones aparezcan en desorden en el resultado; esto es, que las primeras actualizaciones siempre aparecerán antes que las actualizaciones posteriores.

En el ejemplo, la transacción 2 cambia una fila, pero no comete los cambios. La transacción 1 entonces lee los datos sin cometer. Si ahora la transacción 2 deshace sus cambios (ya leídos por la transacción 1) o realiza otros cambios, entonces los datos que ha recuperado la transacción 1 serán erróneos.

|  |  |
| --- | --- |
| **Transacción 1** | **Transacción 2** |
| */\* Query 1 \*/*  **SELECT** edad **FROM** usuarios **WHERE** id = 1;  */\* leerá 20 \*/* |  |
|  | */\* Consulta 2 \*/*  **UPDATE** usuarios **SET** edad = 21 **WHERE** id = 1;  */\* No se hace commit \*/* |
| */\* Query 1 \*/*  **SELECT** edad **FROM** usuarios **WHERE** id = 1;  */\* leerá 21 \*/* |  |
|  | **ROLLBACK**; */\* LECTURA SUCIA basada en bloqueo \*/* |

Pero no existe ningún usuario que tenga la edad de 21.

**10. Cite por lo menos 3 ejemplos que ocasiona ejecutar Transacciones Concurrentes.**

Ejemplo: Sea Ti una transacción para transferir 10.000 Bs. de la cuenta ‘A’ a la Cuenta ‘B’. Se puede definir dicha transacción como:

Ti: 1. Leer(A);

2. A= A – 10.000;

3. Escribir(A);

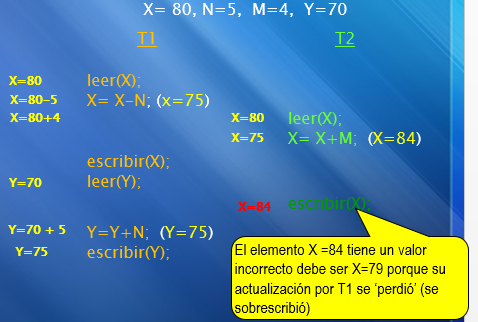
4. leer (B);

5. B= B + 10.000;

6. Escribir (B);

Ejemplo: sistema de bases de datos que permite hacer y anular reservas de asientos en vuelos de diferentes compañías aéreas.

* + Se almacena **un registro por cada vuelo**, que incluye, entre otras cosas, el número de asientos reservados en el vuelo
  + Sean **dos transacciones T1 y T2 concurrentes**:
    - **T1 transfiere N reservas realizadas en un vuelo X a otro vuelo Y**
    - **T2 reserva M plazas en el vuelo X**



**11. En los SGBD ¿Cual es función del Planificador de Transacciones?**

El planificador crea agendas, secuencias ordenadas de las operaciones de Lectura/Escritura tomadas por una o más transacciones

**12. Que es el Protocolo de Control de Concurrencia**

Para evitar que se destruya la consistencia de la base de datos los SGBD utilizan mecanismo denominados Protocolo de Control de Concurrencia.

**13. Cuál es el objetivo del Protocolo de Control de Concurrencia**

Planificar las transacciones de forma que no ocurran interferencias entre ellas, y así evitar la aparición de los problemas mencionados anteriormente.

**14. Si los SGBD permiten ejecutar de manera intercalas las instrucciones de T1 y T2, cite por lo menos tres formas de ejecutar ambas transacciones.**

T1

T2

leer (X);

X:= X-N;

escribir(X);

leer(Y);

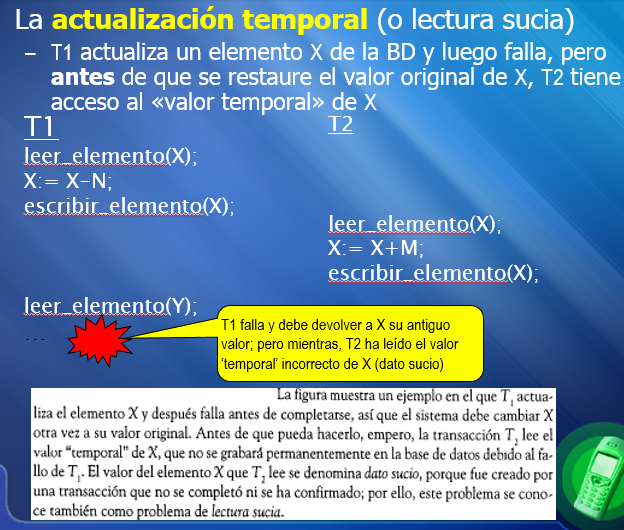
Y:=Y+N;

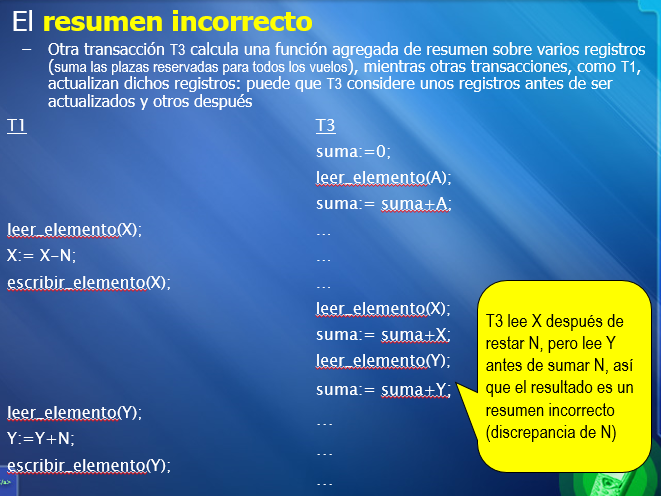
escribir(Y);

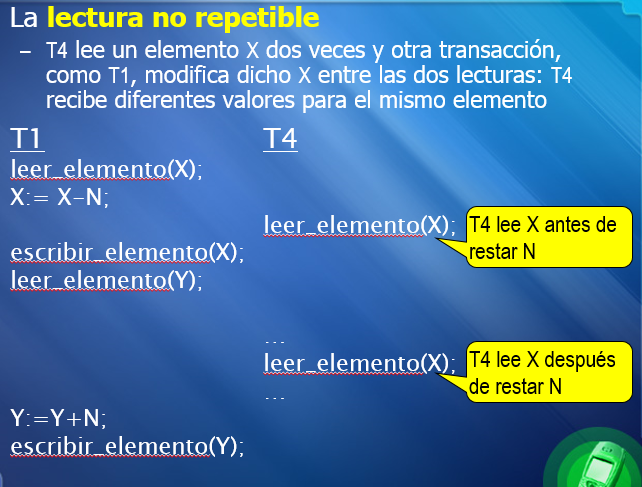
leer(X);

X:= X+M;

escribir(X);







**15. Que es una Planificación de Transacciones y que condiciones debe cumplirse.**

Cada transacción comprende una secuencia de operaciones que incluyen acciones de lectura y escritura en la BD, que finaliza con una confirmación (commit) o anulación (rollback)

**16. La tabla 1 especifica la abreviación de las operaciones que participan en una Transacción, usando la tabla 1 se escriben las planificaciones PA y PB, escriba otras posibles planificaciones PC, PD y PE. (El subíndice de cada operación indica a la transacción que pertenece)**

PA: l1(X) ; e1(X) ; l1(Y) ; e1(Y) ; c1 ; l2(X) ; e2(X) ; c2 ;

PB: l2(X) ; e2(X) ; c2 ; l1(X) ; e1(X) ; l1(Y) ; e1(Y) ; c1 ;

PC: l1(X) ; l2(X) ; e1(X) ; l1(Y) ; e2(X) ; c2 ; e1(Y) ; c1 ;

PD: l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ;

PE: l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ;

operación

abreviatura

leer

l

escribir

e

commit

c

rollback

r

Tabla 1. Abreviación de las operaciones

T1

T2

leer (X);

X:= X-N;

escribir(X);

leer(Y);

Y:=Y+N;

escribir(Y);

leer(X);

X:= X+M;

escribir(X);

**17. Para las transacciones T1 y T2 ¿Cuál de las siguientes planificaciones son correctas y cuáles no? Indicar en cada caso por qué.**

PA: l1(X) ; e1(X); l1(Y); e1(Y); c1 ; l2(X) ; e2(X) ; c2 ; CORRECTA

PB: e1 (Y); l1(X) ; l1(Y); e1(X) ; c1 ; e2(X) ; l2(X) ; c2 ; INCORRECTA

Porque las operaciones T1 y T2 no se ejecutan en el mismo orden que la transacción original

PC: l2(X); c2; l1(X); e1(X);e1(Y); c1 ; INCORRECTA

Porque faltan operaciones de T1 y T2.

PD: l2(X); e2(X); c2; l1(X);e1(X);l1(Y); e1(Y) ; c1 ; CORRECTA

T1

T2

leer (X);

X:= X-N;

escribir(X);

leer(Y);

Y:=Y+N;

escribir(Y);

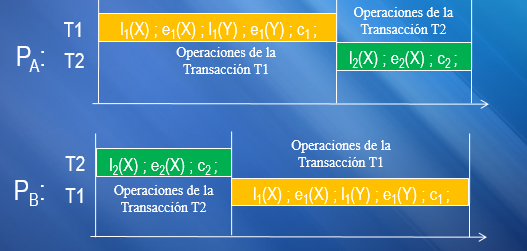
leer(X);

X:= X+M;

escribir(X);

**18. Defina el concepto de Planificación Serie**

Una **planificación serie P** es aquella en la que *las operaciones de cada transacción se ejecutan consecutivamente* sin que se intercalen operaciones de otras transacciones.



**19. Defina el concepto de Planificación No Serie**

Una Planificación No Serie P es aquella en la que las operaciones de un conjunto de transacciones concurrentes se ejecutan intercaladas.

**20. Cual la diferencia entre Planificación Serie y Planificación No serie**

Toda planificación serie es **correcta** ⏵BD consistente

Pero no se garantiza que los resultados de todas las ejecuciones en serie de las mismas transacciones sean idénticos en general, son inaceptables en la práctica (ineficiencia) transacciones

La Planificaciones No Serie permiten llevar la BD a un estado al que pueda llegarse mediante una ejecución en Serie.

**21. Para las transacciones T1 y T2 ¿Cuál de las siguientes planificaciones son Serie y cuáles no?. Indicar en cada caso por qué.**

PA : l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ; SERIE

PB: l1(X) ; e1(X) ; l2(X) ; e2(X) ; l1(Y) ; c2 ; e1(Y) ; c1 ; SERIE

PC: l1(X) ; e1(X) ; l2(X) ; e2(X) ; l1(Y) ; e1(Y) ; c2 ; c1 ; NO SERIABLE

PD: l1(X) ; e1(X) ; l2(X) ; e2(X) ; l1(Y) ; e1(Y) ; c1 ; c2 ; NO SERIEABLE PORQUE SE EJECUTAN INTERCALADAS

PE: l1(X) ; e1(X) ; l2(X) ; l1(Y) ; e2(X) ; e1(Y) ; c1 ; c2 ; SERIE PORQUE SE EJECUTAN CONSECUTIVAMENTE

PF: l1(X) ; e1(X) ; l2(X) ; l1(Y) ; e1(Y) ; e2(X) ; c1 ; c2 ; SERIE

PG: l1(X) ; e1(X) ; l2(X) ; l1(Y) ; e1(Y) ; c1 ; e2(X) ; c2 ; SERIE

PH: l1(X) ; e1(X) ; l1(Y) ; l2(X) ; e1(Y) ; c1; e2(X) ; c2 ;NO SERIABLE

PI: l1(X) ; e1(X) ; l1(Y) ; e1(Y) ; l2(X) ; c1 ; e2(X) ; c2 ; NO SERIABLE

PJ: l1(X) ; e1(X) ; l1(Y) ; e1(Y) ; c1 ; l2(X) ; e2(X) ; c2 ; NO SERIABLE

T1

T2

leer (X);

X:= X-N;

escribir(X);

leer(Y);

Y:=Y+N;

escribir(Y);

leer(X);

X:= X+M;

escribir(X);

**22. Que es un Plan Equivalente**

Un plan equivalente es aquel que produce los mismos efectos en la BD.

**23. Defina el concepto de Planificación Serializable.**

Una planificación P (no serie) es serializable si es equivalente a alguna planificación serie de la misma n transacciones.

**24. Existen dos manera de definir la Equivalencia entre Transacciones ¿Cuáles son?**

* + Equivalencia por conflictos
  + Equivalencia de vistas

**25. Explique ¿Cuándo dos operaciones de una Planificación están en conflicto? Cite ejemplos.**

**¿Cuándo dos operaciones de un plan están en conflicto?**

En una planificación, dos **operaciones** están **en conflicto** si

* + pertenecen a **diferentes** **transacciones**,
  + tienen acceso al **mismo elemento** leer(X),
  + y al menos una de ellas es **escribir**(X)

P: l1(X) ; e2(X) ;

P: w1(X) ; e2(X) ;

P: w1(X) ; l2(X) ;

Operaciones en conflicto en las planificaciones PC y PD:

**PC: l1(X) ; l2(X) ; e1(X) ; l1(Y) ; e2(X) ; c2 ; e1(Y) ; c1;**

**PD: l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ;**

**26. Dada la siguientes Planificaciones, indicar cuales operaciones están en conflicto, para cada caso explicar por qué.**

PA: l1(X) ; l2(X) ; e1(X) ; l1(Y) ; e2(X) ; c2 ; e1(Y) ; c1; CONFLICTO

PORQUE PERTENECEN A DIFERENTES TRANSACCIONES

PB: l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ;

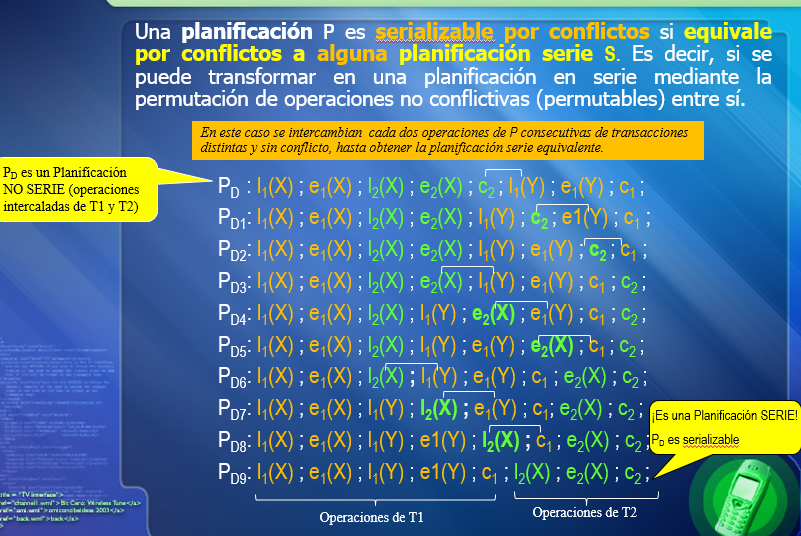
**27. ¿Cuándo un Plan No Serie es equivalentes por conflictos?**

Dos planes son equivalentes por conflictos si el orden de cualesquiera dos operaciones en conflicto es el mismo en ambos planes.

**28. Dada la siguientes Planificación No Serie PD, validar paso a paso si la misma es equivalente por Conflicto.**

PD : l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ; SI ES POR CONFLICTO

**29. Describa los pasos del Algoritmo (usando grafos) que permite determinar si un Plan es serializable por conflictos.**



**30. Dados los Planes PA, PB y PC dibujar los grafos para probar la serializabilidad por conflictos usando el Algoritmo.**

PA: l1(X) ; e1(X) ; l1(Y) ; e1(Y) ; c1 ; l2(X) ; e2(X) ; c2 ;

PB: l1(X) ; e1(X) ; l2(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ;

PC: l1(X) ; l2(X) ; e1(X) ; e2(X) ; c2 ; l1(Y) ; e1(Y) ; c1 ;

