UNIVERSIDAD NACIONAL DE LA MATANZA

DEPARTAMENTO DE INGENIERÍA
E INVESTIGACIONES TECNOLÓGICAS

INGENIERIA EN INFORMATICA

BASE DE DATOS

Práctica de Transacciones

Jefe de Cátedra: *Ing. Verónica Ichazo*

Docentes a cargo de curso:
Ing. Alfonso Palomares
Ing. Natalia Crespo
Ing. Guillermo Giannotti

Docentes a cargo de práctica:
Ing. Matías López
Ing. Fernando Ybarra
Ing. Javier Rebagliatti
Ing. Sebastián Deuteris
Ing. Hernán Jalil

2019

Ejercicio 1

Realice el Grafo de Precedencia. Indique si la planificación es serializable. Justifique.

	Transacción 1	Transacción 2	Transacción 3	Transacción 4
1				LL(A)
2	LE(B)			
3			LL(C)	
4			LL(B)	
5		LE(B)		
6		LL(A)		
7				LE(B)
8	LL(A)			

Ejercicio 2

Dada la siguiente ejecución entrelazada, indique si es serializable. Justificar.

	Transacción A	**Transacción B**	**Transacción C**
1	LL(E)		
2			LL(E)
3	LE(D)		
4			LE(F)
5		LL(F)	
6			LE(D)
7		LE(E)	

Ejercicio 3

Indique si el siguiente plan es serializable por conflictos: Arme el grafo de precedencia.

	T1	T2	T3
1		LL(Z)	

2		LL(Y)	
3		LE(Y)	
4			LL(Y)
5			LL(Z)
6	LL(X)		
7	LE(X)		
8			LE(Y)
9			LE(Z)
10		LL(X)	
11	LL(Y)		
12	LE(Y)		
13		LE(X)	

EJERCICIO 4

Indique si el siguiente plan es serializable por conflictos: Arme el grafo de precedencia.

	T1	T2	T3
1			LL(Y)
2			LL(Z)
3	LL(X)		
4	LE(X)		
5			LE(Y)
6			LE(Z)
7		LL(Z)	
8	LL(Y)		
9	LE(Y)		
10		LL(Y)	
11		LE(Y)	
12		LL(X)	

13		LE(X)	

EJERCICIO 5

Realice el Grafo de Precedencia completo e indique si la planificación es serializable. Justifique.

	T1	T2	T3	T4
1				LL(A)
2	LL(A)			
3	LL(C)			
4	LE(A)			
5		LL(B)		
6		LL(C)		
7		LE(D)		
8				LL(F)
9				LE(F)
10			LL(A)	
11			LE(B)	
12			LL(C)	
13			LE(F)	
14		LL(C)		
15		LL(F)		
16		LE(C)		
17				LL(D)
18				LL(F)
19				LE(A)

EJERCICIO 6

Realice el Grafo de Precedencia completo e indique si la planificación es serializable. Justifique.

	T1	T2	T3	T4
1				LL(A)
2				LI(A)
3	LL(A)			
4	LL(C)			

5	LE(A)			
6	LI(A)			
7	LI(C)			
8		LL(B)		
9		LL(C)		
10		LE(D)		
11		LI(D)		
12		LI(C)		
13		LI(B)		
14				LL(F)
15				LE(F)
16				LI(F)
17			LL(A)	
18			LE(B)	
19			LL(C)	
20			LE(F)	
21			LI(F)	
22			LI(C)	
23			LI(B)	
24			LI(A)	
25		LL(C)		
26		LL(F)		
27		LE(C)		
28		LI(C)		
29		LI(F)		
30				LL(D)
31				LL(F)
32				LE(A)
33				LI(A)
34				LI(F)
35				LI(D)

Ejercicio 7

Dadas las siguientes dos transacciones concurrentes, indique con cuál de las tres opciones de niveles de aislamiento se produciría Deadlock.

Transacción 1	Transacción 2
SET TRANSACTION ISOLATION LEVEL ……;	
	SET TRANSACTION ISOLATION LEVEL ……;
BEGIN TRANSACTION;	
	BEGIN TRANSACTION;

```
SELECT          *
FROM  Producto
WHERE           precio > 30;
```

```
                                SELECT *
                                FROM          Empleado
                                WHERE  salario > 20000;
```

```
UPDATE  Empleado
SET   salario = salario * 1.15 ;
```

```
                                INSERT INTO Producto
                                (codigo, descripcion, precio)
                                VALUES (223, 'Pepsi 3 litros', 42);

                                COMMIT TRANSACTION;
```

```
COMMIT TRANSACTION;
```

Opciones de aislamiento:

	T1	T2
Opcion 1	Repeatable read	Serializable
Opcion 2	Serializable	Repeatable read
Opcion 3	Repeatable read	Repeatable read

Ejercicio 8

Dadas las siguientes transacciones concurrentes, explique qué problema se presenta y en cuál de las transacciones. Luego indique qué cambio haría (solo uno) para solucionarlo.

Transacción 1 | Transacción 2 | Transacción 3

```
SET TRANSACTION ISOLATION
LEVEL READ COMMITTED;
```

```
                                SET TRANSACTION ISOLATION
                                LEVEL READ COMMITTED;
```

```
                                                        SET TRANSACTION
                                                        ISOLATION
                                                        LEVEL READ
                                                UNCOMMITTED;
BEGIN TRANSACTION;
```

```
                                BEGIN TRANSACTION;
```

```
                                                        BEGIN TRANSACTION;
SELECT  *
FROM     Producto
WHERE  precio_venta > precio_costo;
```

```
                                SELECT *
                                FROM          Producto
                                WHERE  precio_venta < precio_costo;
```

```
                                                        INSERT INTO Producto
```

```
                                                    (cod_prod, nom_prod,
precio_costo, precio_venta)
                                                    VALUES (101,'Pringles', 35,
48);
                                                    COMMIT TRANSACTION;
                          SELECT *
                          FROM        Producto
                          WHERE  precio_venta < precio_costo;
                          COMMIT TRANSACTION;
SELECT  *
FROM    Producto
WHERE  precio_venta > precio_costo;
COMMIT TRANSACTION;
```

Ejercicio 9

Dadas las siguientes transacciones concurrentes, explique qué problema se presenta y en cuál de las transacciones. Luego indique qué cambio haría (solo uno) para solucionarlo.

```
      Transacción 1              Transacción 2              Transacción 3
SET TRANSACTION ISOLATION
LEVEL READ COMMITTED;
                                 SET TRANSACTION ISOLATION
                                 LEVEL READ UNCOMMITTED;
                                                            SET TRANSACTION
                                                            ISOLATION
                                                            LEVEL READ COMMITTED;
BEGIN TRANSACTION;
                                 BEGIN TRANSACTION;
                                                            BEGIN TRANSACTION;
SELECT  count(*)
FROM    Cliente
WHERE  categoria = 2;
                                                            SELECT *
                                                            FROM      Cliente
                                                            WHERE  categoría = 5;
                                 UPDATE Cliente SET categoria = 5
                                 WHERE categoria = 4;
                          COMMIT TRANSACTION;
                                                            SELECT *
                                                            FROM      Cliente
                                                            WHERE  categoría = 5;
                                                            COMMIT TRANSACTION;
SELECT  count(*)
FROM    Cliente
WHERE  categoria = 2;
COMMIT TRANSACTION;
```

Ejercicio 10

1) Dadas las siguientes transacciones:

```
       T1:
       begin tran
T1S1          update estudiante set carrera='INGE' where carrera='ING'
       commit tran
       T2:
       begin tran
T2S1          insert into estudiante values(1,'Juan','ING')
```

```
T2S2          insert into estudiante values(2,'Pedro','ING')
       commit tran
```

Indicar si las siguientes afirmaciones son V ó F y justifique su respuesta:

a) Luego del commit, Juan y Pedro pueden tener diferentes valores en carrera, relativo al momento en que se hayan ejecutado las sentencias.
b) El proceso provoca Deadlock, ya que no se puede insertar y actualizar en la misma tabla en el mismo momento.
c) La T1 siempre se queda esperando a que finalice la T2, ya que sin insertar no tendría ninguna fila para poder actualizar.
d) Esto sólo funcionaria si se utilizara el nivel de Isolation Read Uncommitted.