

# EPL646 – Advanced Topics in Databases

## Concurrency control with Locking

<http://www.cs.ucy.ac.cy/~dzeina/courses/epl646/labs/lab.html>



University  
of Cyprus



# Exercise 1

**(Exercise 1)** Prove the following:

- a) Strict two-phase locking ensures strict schedules
- b) Cautious waiting avoids deadlocks

# Exercise 2

- (Exercise 2)** For each of the following locking protocols determine the type of schedules they allow (conflict-serializable, recoverable, avoids-cascading-aborts, strict).
- Always get an exclusive lock before writing. Keep all exclusive locks till the end of the transaction. There are no shared locks.
  - In addition to (a), get a shared lock before reading. Shared locks are released at any time.
  - In addition to (b) we have two-phase locking.
  - In addition to (b) all locks are held till the end of the transaction.

# Exercise 2

	Conflict-serializable	Recoverable	Avoid cascading aborts
<b>a</b>			
<b>b</b>			
<b>c</b>			
<b>d</b>			

# Exercise 3

**(Exercise 3)** Use the timestamp ordering algorithm with the following schedules to determine whether their execution would be allowed.

# Exercise 3

## (Exercise 3 a) Schedule E

Transaction T <sub>1</sub>	Transaction T <sub>2</sub>	Transaction T <sub>3</sub>
	read_item(Z);	
	read_item(Y);	
	write_item(Y);	
		read_item(Y);
		read_item(Z);
read_item(X);		
write_item(X);		
		write_item(Y);
		write_item(Z);
	read_item(X);	
read_item(Y);		
write_item(Y);		
	write_item(X);	

# Exercise 3

## (Exercise 3 b) Schedule F

Transaction T <sub>1</sub>	Transaction T <sub>2</sub>	Transaction T <sub>3</sub>
		read_item(Y);
		read_item(Z);
read_item(X);		
write_item(X);		
		write_item(Y);
		write_item(Z);
	read_item(Z);	
read_item(Y);		
write_item(Y);		
	read_item(Y);	
	write_item(Y);	
	read_item(X);	
	write_item(X);	

# Questions?

<http://www.cs.ucy.ac.cy/~dzeina/courses/epl646/labs/lab.html>



University  
of Cyprus

