1. Consider the concurrent transactions T_1, T_2, T_3 and T_4 under the schedule S in Figure 2, which access relational tables *Student*(*stuID*, *stuName*, *age*, *height*) and *Grade*(*stuID*, *cosID*, *grade*) concurrently. The tuples/rows in the tables are viewed as the data items to be locked by DBMS and independently accessed by the transactions.

(1) Construct the precedence graph for S.

(2) Is *S* a serializable schedule? If not, give the reason. If it is, give a serial schedule that is equivalent to S.

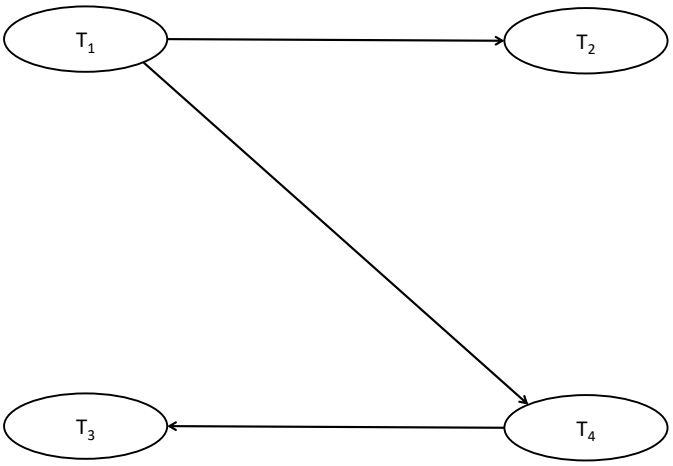
(3) Is S a recoverable schedule, and why?

(4) Is S a cascadeless schedule, and why?

T_1	T_2	T_3	T_4
begin_transaction			
	begin_transaction		
update *Student* set *stuName='Li'* where *stuID*=10			
		begin_transaction	
	update *Student* set *age=age*+1 where *stuID*=20		
		select *age* from *Student* where *stuID*=50	
			begin_transaction
			update *Student* set *height=height*+5 where *stuID*=30
update *Grade* set *grade=grade*+1 where *stuID*=40 AND *CosID*=10			
	select *stuName* from *Student* where *stuID*=10		
	commit		
commit			
			select *stuID*, *grade* from *Grade* where *stuID*=40 AND *CosID*=10
			commit
		Select *height* from *Student* where *stuID*=30	
		commit	

Figure 2 Schedule S

(1) 前驱图



说明：

T_1 修改了 student 表中 *stuID*=10 的元组，T_2 随后又读取该元组，因此存在一条从到的冲突边；对其它各条冲突边可做类似分析。

(2) 前驱图中无回路，因此是冲突可串行的。

(3) 不是可恢复调度。

T2 读取 student 表中 *stuID*=10 的元组时，该元组内容已由 T1 修改过，**但 T2 提交操作 commit 早于 T1 提交 commit**。一旦 T1 在 T2 的 commit 操作之后回滚其 update *student* 操作，将 *stuName* 回滚为旧值，则 T2 的 select 操作无法随着回滚，T2 读取的仍然是 *stuName* 修改后的值（1 分）。

（4）不是无级联回滚调度

因为不是可恢复调度，因此也不是无级联回滚调度。

或者：无级联回滚调度中，一个事务只能读取其它事务已经提交的数据。

但 T2 读取 student 表中 *stuID*=10 的元组时，该元组被被 T1 修改，但未提交。

2. The schedule *S* in Fig. 3 controls the transaction T_1, T_2, T_3 and T_4 to concurrently access the database *University*.

T_1	T_2	T_3	T_4
begin trans			
update *Instructor* set *salary*=2200 where *ID*=2019			
	begin trans		
	update *Student* set *age*=*age*+2 where *sid*=211		
checkpoint			
commit			
		begin trans	
		update *Course* set *credit*=3 where *title*=DBS	
			begin trans
			update *Department* set *budget*=20000 where *dname*=CompSci
checkpoint			
	rollback		
			update *Instructor* set *salary*=salary+200 where *ID*=2019
			commit
		select *credit* from *Course* where *ID*=211	
system crash			

Figure 3 Schedule *S*

These transactions modify the following data items:

1) the value of the attribute *salary* of the *instructor* whose *ID* is 2019 in *Instructor*(*ID*, *name*, *department*, *salary*),

2) the value of the attribute *age* of the *student* whose *s_id* is 211 in *Student*(*sid*, *name*, *department*, *age*),

3) the value of the attribute *credit* of the course *Database System Concepts* (i.e. DBS) in *Course*(*course_id*, *title*, *department*, *credit*), and

4) the value of the attribute *budget* of the department *Computer Science* (i.e. CompSci) in

Department(*dname*, *building*, *budget*).

It is assumed that

i) the initial values of the attribute *salary*, *age*, *credit* and *budget* to be updated are 2000, 18, 0, and 10000, respectively; and

ii) immediate database modification and checkpoint techniques are employed.

Q&A:

(1) Give the log records of the schedule *S*.

<T1, start>
<T1, salary, 2000, 2200>
<T2, start>
<T2, age, 18, 20>
Checkpoint{T1, T2}
<T1, commit>
<T3, start>
<T3, credit, 0, 3>
<T4, start>
<T4, budget, 10000, 20000>
Checkpoint{T2,T3,T4}
<T2, age,18> /rollback 产生的日志
<T2, abort> /rollback 产生的日志
<T4, salary, 2200, 2400>
<T4, commt>
crash

(2) After the second checkpoint, what are the values of the data items *salary*, *age*, *credit* and *budget* in the database?

***salary*=2200, *age*=20, *credit*=3, and *budget*=20000**

(3) After the system crash occurs, what recovery actions (i.e. redo, undo, ignore) should be conducted respectively for T_1, T_2, T_3, and T_4?

T_1: ignore T_2: ignore or redo
T_3: undo, T_4: redo

(4) After recovery operations on T_1, T_2, T_3, and T_4 are completed, what are the values of the data items *salary*, *age*, *credit* and *budget* in the database?

***salary*=2400, *age*=18, *credit*=0, and *budget*=20000**