

Australian National University
Department of Computer Science

COMP2400/COMP6240

Tutorial Exercise – Week 12
 Transactions / Serializability

You should try to complete this tutorial exercise on transactions and serialisability before your session in Week 12. Expect Part A to be discussed only if students have prepared answers ahead of time.

Part A : Transactions

Consider the relational schema from earlier laboratory exercises augmented with a Prereq table and a view. Also assume Course Supervisors have Oracle accounts with username the same as the CourseCode (e.g. COMP2400). Course Supervisors have read access on all the base tables, and may change enrolment data only for their own courses, but not for other courses.

Student (StudentId, Surname, Initials, Degree)

Course (CourseCode, CourseName, ScienceGroup, Points, Dept)

Enrolment (StudentId, CourseCode, Year, Mark, Grade)

Prereq (CourseCode, PrereqCode)

View **CourseEnrolments** (StudentId, CourseCode, Year, Mark, Grade) defined by

```
CREATE OR REPLACE VIEW CourseEnrolments AS
SELECT StudentId, CourseCode, Year, Mark, Grade
FROM Enrolment
WHERE TO_CHAR(Year) = TO_CHAR(SYSDATE, 'YYYY')
      AND UPPER(CourseCode) = USER
WITH CHECK OPTION
```

For the following transactions (carried out by course supervisors) identify which base tables will be read-accessed and which base tables will be write-accessed. Identify what input values the user will need to provide for the transaction. Also identify the SQL statements required for the database access.

1. Amendment of Mark and Grade for a particular student by a Course Supervisor
2. Withdrawal of a student from enrolment in a course if the student does not satisfy the course prerequisites.

Part B : Serializability

E&N Algorithm 17.1 gives instructions for constructing a **precedence graph** to determine serializability. Lecture 24 also discusses serializability.

A precedence graph consists of :

- A node for each transaction,
- A directed edge $T_i \rightarrow T_j$ if T_j reads the value of an item written by T_i ,
- A directed edge $T_i \rightarrow T_j$ if T_j writes a value into an item after it has been read by T_i ,
- A directed edge $T_i \rightarrow T_j$ if T_j writes a value into an item after it has been written by T_i .

Consider schedules **S1** and **S2** of four transactions T_1, T_2, T_3, T_4 running in inter-leaved mode. The transactions update database items A, B, C, D, E, reading the item before writing a new value (the ‘constrained write’ assumption). The schedules show the order of database reads and writes.

1. Are the schedules S1 and S2 serializable? If a schedule is serializable, give an equivalent serial order of T_1, T_2, T_3 and T_4 . If it is not serializable, indicate which transactions have a non-serializable interaction. Show how you arrived at your answers.
2. Suppose **strict two-phase locking** is used as the method of concurrency control. Redraw the schedules showing the (exclusive) **locks**, **unlocks** and **waits** (when a transaction has to wait for a lock on a db item). See Lecture 25 for an example. Assume that if more than one transaction is waiting for an item, they are put in a queue. Redraw the precedence graph and comment on any significant changes to the schedule caused by application of the two-phase locking protocol. In particular, report on the equivalent serial order of transactions that complete successfully, and which transactions (if any) are involved in deadlock. If there is deadlock, show what will happen when a transaction is rolled back to break the deadlock.

Note: If you have not yet successfully tested any of your views from the Week 10 exercises, ask your tutor to help you.

Schedule S1

time	T1	T2	T3	T4
1			Read(B)	
2		Read(C)		
3		Read(D)		
4			Write(B)	
5	Read(A)			
6		Write(D)		
7				Read(D)
8				Read(E)
9				Write(D)
10		Write(C)		
11			Read(C)	
12			Write(C)	
13	Read(B)			
14	Write(B)			
15				Write(E)
16			Read(D)	
17	Write(A)			
18		Read(A)		
19		Write(A)		
20				Read(A)
21			Write(D)	
22				Write(A)

Schedule S2

time	T1	T2	T3	T4
1				Read(D)
2	Read(A)			
3		Read(C)		
4	Read(B)			
5	Write(B)			
6				Write(D)
7			Read(B)	
8		Read(D)		
9		Write(D)		
10		Write(C)		
11			Read(C)	
12			Write(B)	
13	Write(A)			
14				Read(A)
15				Read(E)
16				Write(A)
17			Write(C)	
18			Read(D)	
19		Read(A)		
20		Write(A)		
21			Write(D)	
22				Write(E)