University of California, Los Angeles
Department of Computer Science

Computer Science 143 **Prof. Ryan Rosario**

Homework 5

Due Wednesday, June 5, 2019 at 11:59pm on CCLE

Please remember the following:

1. Homework is mostly graded on completion. We may grade a few parts, but it will never be the majority of the grade on the assignment. So try your best, and focus on solving the problems. Consider homework (and studying the solutions) as practice for the final exam.
2. Homework must be submitted digitally, on CCLE. We will not do any paper grading. You can use a text file, but if you use Word, a PDF is preferred rather than a DOC file.
3. Solutions will be posted.
4. **Students are bound by the signed Academic Integrity Agreement. Students copying off of each other (or from other sources) or having unusually similar responses (without citing who they worked with), are easy to identify and will receive a grade of 0.**

1. We will use the following transaction schedule S for this problem.

T_1	T_2
`read(A)`	
`write(A)`	
	`read(A)`
	`read(B)`
	`commit`
`read(B)`	
`write(B)`	
`commit`	

   (a) Is S serial?

   (b) Is S conflict serializable? If so, what are the equivalent serial schedules? Show your work using the swap method or the graph method.

2. Look carefully at these three transactions T_1, T_2 and T_3.

T_1	T_2	T_3
`write(A)`		
	`read(A)`	
		`write(B)`
`write(B)`		
		`write(B)`
	`write(A)`	
		`read(B)`
	`read(B)`	

   (a) Construct the precedence graph for this schedule S.

   (b) Is S conflict serializable? Justify the answer.

3. Consider the relation `BankAccount(acctno, money)` where we store the amount of money in a, well, bank account, and `acctno` is the key. Suppose we execute the following three transactions.

   T_1:

   ```
   SELECT SUM(money) FROM BankAccount;
   COMMIT;
   ```

   T_2: In this transaction, the bank has lost its mind and gives everyone a \$100 bonus to stimulate spending. But your professor (`acctno = 7`) gets an additional \$1000 because he is cheap and the bank wants him to perform some transactions.

   ```
   UPDATE BankAccount SET money = money + 100;
   UPDATE BankAccount SET money = money + 1000 WHERE acctno = 7;
   COMMIT;
   ```

   T_3: We some more money to account 7 in this transaction. We will also set `acctno = 42` to \$0 because he died and had some liens on his account.

   ```
   UPDATE BankAccount SET money = money + 1000 WHERE acctno = 7;
   UPDATE BankAccount SET money = 0 WHERE acctno = 42;
   COMMIT;
   ```

   The `BankAccount` table originally has two tuples `(7, 15000)` and `(42, 25000)`. Assume that *individual* SQL statements execute atomically.

   (a) If all three transactions execute under the `SERIALIZABLE` isolation level, list all possible values that can be returned by T_1. Explain your answer.

   (b) If T_1 executes under the `READ UNCOMMITTED` isolation level and T_2 under `REPEATABLE READ` access level, and T_3 under the `SERIALIZABLE` isolation level, list all possible values that can be returned by T_1. Explain your answer.