CS145 Written Assignment #6

Due Thursday May 20*

1. The personnel database you worked on in Written Assignment #5 for the Springfield Nuclear Power Plant (SNPP) has been in operation for some time now. Its schema is repeated below:

   ```
   Dept(DNo, name, budget)
   Emp(SSN, name, salary, DNo)
   Mgr(SSN, assistantSSN)
   ```

   Both Mr. Burns and Smithers are active users of the database. One day, Smithers was gathering some salary statistics for SNPP's biannual report. He wrote a query to compute the lowest and the highest salaries, and inserted them into an initially empty table, `SalaryStat`. Just to be sure, he repeated the same `INSERT` statement again. The two `INSERT` statements ran as one transaction which then committed:

   ```
   INSERT INTO SalaryStat           -- S1
   (SELECT MIN(salary), MAX(salary) FROM Emp);
   INSERT INTO SalaryStat           -- S2
   (SELECT MIN(salary), MAX(salary) FROM Emp);
   COMMIT;                          -- S3
   ```

   At the same time, Mr. Burns was making haphazard executive decisions from his laptop connected to the same database. At first he wanted to give Smithers a raise, but rejected the idea later. Then, he hired Prof. Frink as a consultant for SNPP. Finally, he decided to cut down Barney Gumble's salary.

   ```
   UPDATE Emp SET salary = 85000   -- B1
   WHERE name = 'Smithers';
   ROLLBACK;                        -- B2

   INSERT INTO Emp                  -- B3
   VALUES('142857142', 'Prof. Frink',
          80000, 7);
   COMMIT;                          -- B4

   UPDATE Emp SET salary = 20000   -- B5
   WHERE name = 'Barney Gumble';
   COMMIT;                          -- B6
   ```

   The database resides on a Megatron 2000 DBMS. Megatron 2000 executes one SQL statement at a time; it cannot execute two statements in parallel. On the other hand, Megatron 2000 supports concurrent transactions as specified by the SQL standard. The server may choose to interleave statements from concurrent transactions, as long as the execution schedule is permissible at the current isolation levels.

*Please refer to CS145 Course Information Page (`http://www.stanford.class/cs145/info.html`) for submission instructions and late policy.

Let us assume that all three of Mr. Burns' transactions ran at isolation level `SERIALIZABLE`. Before any transaction began, the lowest salary of SNPP employees was $25,000, the highest was $75,000, and the `SalaryStat` table is empty.

What should the final state of `SalaryStat` be (i.e., what data should be in `SalaryStat`) after both Smithers' and Burns' transactions had completed? There are many possibilities. If Smithers' transaction also ran at isolation level `SERIALIZABLE`, then one possibility is that `SalaryStat` would contain two tuples (25000, 80000) and (25000, 80000) after the following execution sequence: B1, B2, B3, B4, S1, S2, S3, B5, B6, which is permissible at the given isolation levels.

(a) Suppose that Smithers' transaction ran at isolation level `SERIALIZABLE`. What are all the other possible final states of `SalaryStat` besides the one described above? For each possible final state, show one permissible execution sequence that would produce this final state (however, you do not have to list all possible execution sequences that would produce this state).

(b) Suppose that Smithers' transaction ran at isolation level `REPEATABLE READ`. What are all the possible final states of `SalaryStat` that are *not* possible at a stronger isolation level? Again, for each possible final state, show one permissible execution sequence that would produce this final state.

(c) Suppose that Smithers' transaction ran at isolation level `READ COMMITTED`. What are all the possible final states of `SalaryStat` that are *not* possible at a stronger isolation level? Again, for each possible final state, show one permissible execution sequence that would produce this final state.

(d) Suppose that Smithers' transaction ran at isolation level `READ UNCOMMITTED`. Give one possible final state of `SalaryStat` that is *not* possible at a stronger isolation level. Again, show one permissible execution sequence that would produce this final state.

2. Consider the following self-explanatory schema that uses Oracle's object-relational features:

```
CREATE TYPE AddressType AS OBJECT(street CHAR(20), city CHAR(20));
/
CREATE TYPE StudentType AS OBJECT(name CHAR(20), address AddressType);
/
CREATE TYPE CollegeType AS OBJECT(name CHAR(20), city CHAR(20));
/
CREATE TABLE Student OF StudentType;
CREATE TABLE College OF CollegeType;
CREATE TABLE Attends(student REF StudentType, college REF CollegeType);
```

You may assume that student and college names are unique, that all students have exactly one address and attend exactly one college, and that all colleges are located in exactly one city. Write the following queries using Oracle 8 syntax. For (a) and (b), please do not use joins or subqueries.

(a) Find the names of all students who live in Palo Alto and attend Stanford.

(b) Find the names of all students who live in the same city as the college they attend.

(c) Find the names of all pairs of students who live at the same street address in the same city, and go to the same college in their hometown.