**Introduction to Databases, Fall 2019**

**Homework #4 (70 Pts, Dec 22, 2019)**

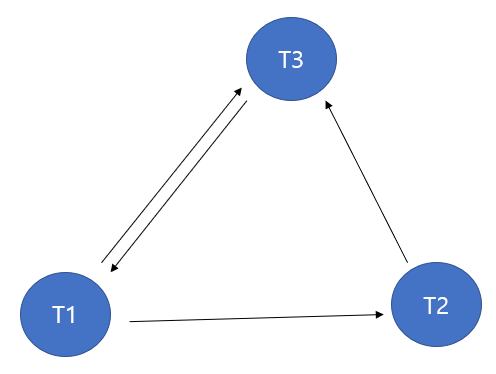
**Student ID 2016311821**

**Name 한승하**

**(1) [20 pts]** Check whether the following schedule is serializable. For a serializable schedule, determine the equivalent serial schedule(s). Note that indicates *read* operation on for . Explain why (**5pts each**).

(a)

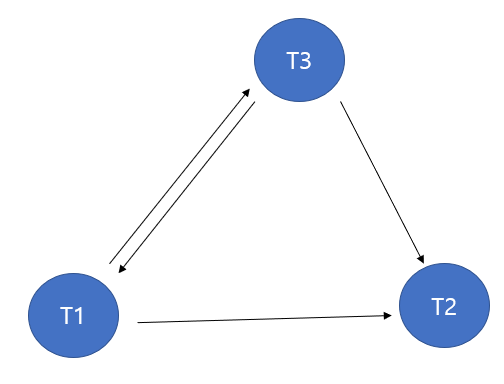
**Answer:**



다음과 같이 구성되어 Serializable 하지 않다. (T1 과 T2 사이에 Loop 이 있다.)

(b)

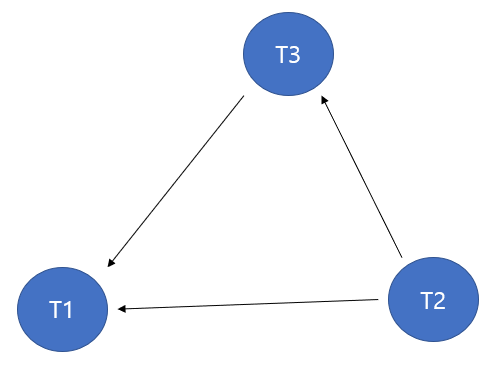
**Answer:**



다음과 같이 구성되어 Serializable 하지 않다. (T1 과 T2 사이에 Loop 이 있다.)

(c)

**Answer:**

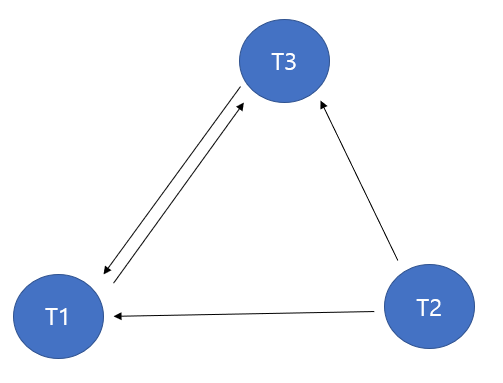


다음과 같이 구성되어 Serializable 하다.

Serial schedule: r3(X); w3(X); r2(X); r1(X); w1(X);

(d)

**Answer:**

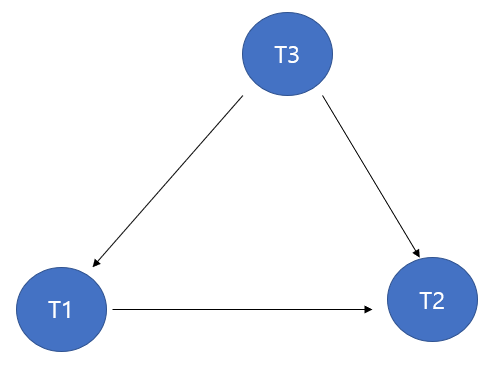


다음과 같이 구성되어 Serializable 하지 않다. (T1 과 T2 사이에 Loop 이 있다.)

**(2) [20 pts]** Consider the three transactions T1, T2, and T3. Draw the serializability (precedence) graphs for two schedules S1 and S2, and check whether each schedule is serializable or not. If the schedule is serializable, write down the equivalent serial schedule(s) (**5pts each**).

(a)

**Answer:**

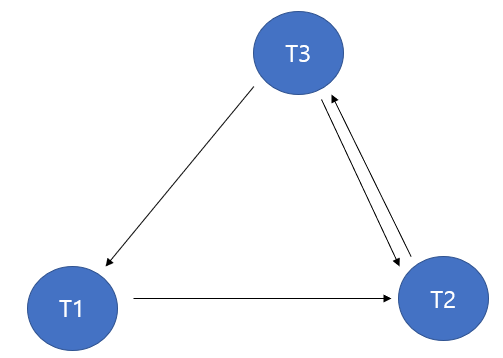


위와 같은 그래프를 가지며 Serializable 하다

Serializable schedule:

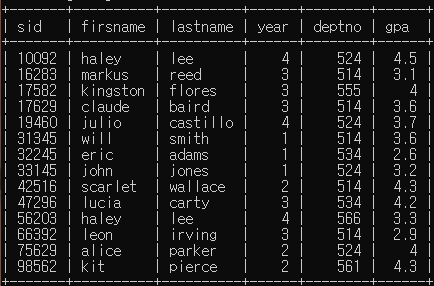
(b)

**Answer:**

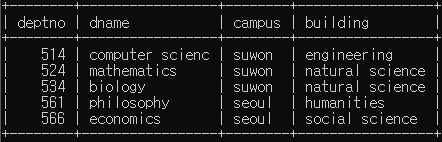


위와 같이 T3과 T2사이 그리고 전체적인 Loop이 생겨 Serializable 하지 않다.

**(3) [30 pts]** Implement the following SQL statement using MapReduce framework. Assume that there are two tables: student(sid, firstname, lastname, year, deptno, gpa) and dept(deptno, dname, campus, building) as follows. **Implement and explain your map and reduce function with execution snapshots.**



**TABLE “student”**

****

**TABLE “dept”**

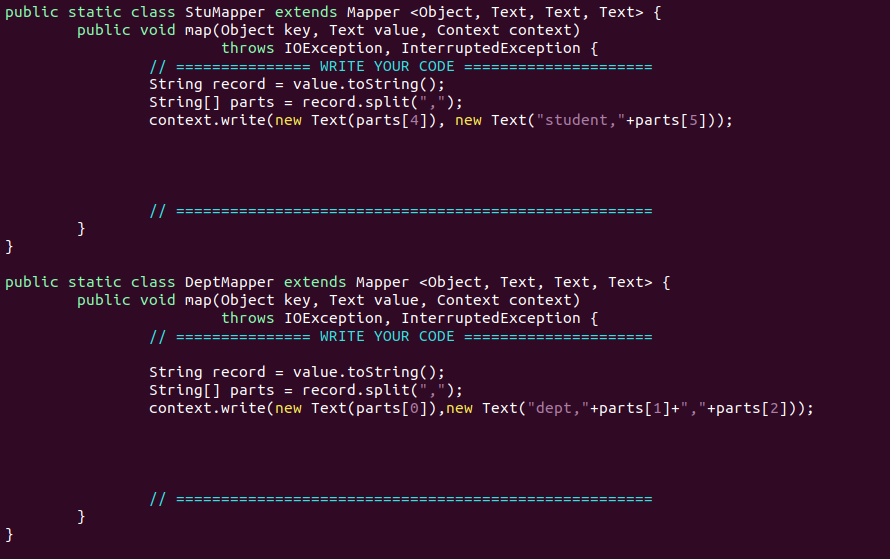
|  |
| --- |
| **SELECT d.dname, max(s.gpa), d.campus**  **FROM student s**  **JOIN dept d ON s.deptno = d.deptno**  **GROUP BY d.dname**  **HAVING avg(s.gpa) > 3.5;** |

**Instruction:** Please write your code to complete MapReduce function in “Answer.java”. (See W14 How to Install Hadoop for information.) Compress ‘Answer.java’ and ‘your report’ (this current document file) and submit with the filename ‘HW4\_STUDENT ID.zip’.

**NOTE**: You should write your codes in ‘WRITE YOUR CODE’ signs. It is not recommended to edit other parts, but you can add/modify functions if you need.

**NOTE**: You can run your code with the provided scripts, for example by simply command ‘sh ./script.txt’. If you have problem compiling your code, delete files and directories **except** ‘input’ and ‘Answer.java’ and re-run the scripts.

**Answer: (Submit your code to i-campus. Don’t write your code here.)**



Map 파트는 위와 같이 작성되었습니다. Join을 할 값인 student의 5번째 tuple, dept의 첫번째 tuple인 deptno를 넘겨주었으며, 그 이후엔 table name과 함께 출력해야 하는 값 (gpa, campus,dname)을 넘겨주었습니다.



Reduce는 위와 같이 작성하였습니다. for문을 돌며 들어오는 input당 student일 경우 gpa값을 체크하여 갱신해 주는 식으로 각 dname당 gpa의 최댓값을 구해주었습니다. 또한, 해당되는 entry의 개수와 gpa총 합을 저장해 후에 평균을 구해주는데 사용하였습니다, 이후 3.5이상인 avg\_gpa값을 가지고 있는 d name에 대해 campus와 max\_gpa를 출력해주었으며 결과창은 아래와 같습니다.

