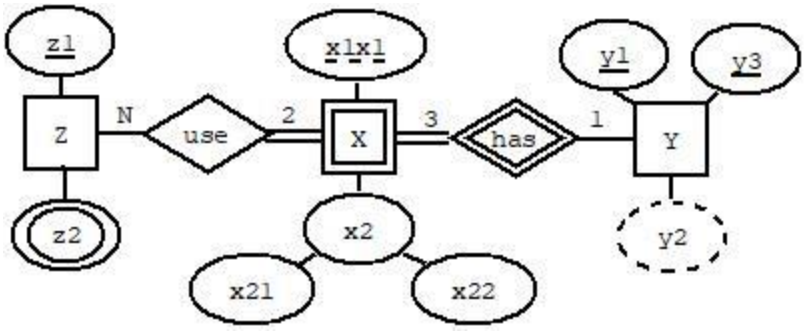
SJSU CMPE 226 HW3 SPRING 2020

REMINDER: Each homework is individual. "Every single byte must come from you." Cut&paste from others is not allowed. Keep your answer and source code to yourself only - never post or share them to any site in any way.

1. (10 pts) Map the following ERD to the schema diagram. But specify the schema diagram in text (for online exam).



2. (30 pts) Consider primary key based normalization only. Given a relation R(A, B, C, D, E, F, G, H, I, J) with primary key AB, and the set of functional dependencies (FDs)

AB → C, A → DE, B → F, F → GH, D → IJ

a. (15 pts, 5 pts each) Is this relation compliant with primary key based 1NF, 2NF, or 3NF? Justify your answer.

1NF: Yes/No, justify:

2NF: Yes/No, justify with all applicable FDs:

3NF: Yes/No, justify with all applicable FDs:

b. (15 pts) Apply primary key based normalization successively (e.g., n NF then n+1 NF then …) until you reach 3NF or you cannot decompose the relations further. For each normalization step, list all the resulting relations, underline the primary key of each relation, and state the reasons behind each decomposition (e.g., remove … problem). Type your answer or use software to draw a diagram; hand-drawing is **not** accepted.

3. (25 pts) Given the following three transactions

T1 = r1(x); w1(y);

T2 = r2(z); r2(y); w2(y); w2(x);

T3 = r3(z); w3(x); r3(y);

Consider the schedule

S = r1(x); r3(z); r2(z); w3(x); r2(y); r3(y); w2(y); w1(y); w2(x);

a. (10 pts) Use any software to draw the precedence graph of schedule *S*, and label each edge with data item(s). Hand-drawing is **not** accepted.

b. (5 pts) Specify the precedence graph in text (for online exam).

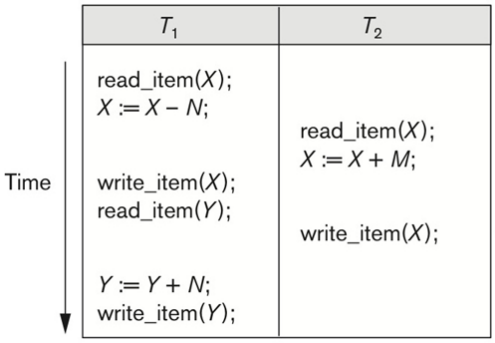
c. (10 pts) Based on the precedence graph, determine whether *S* is conflict serializable and justify your answer. If it is serializable, specify all possible equivalent serial schedule(s), else indicate all cycle(s).

4. (15 pts) Given

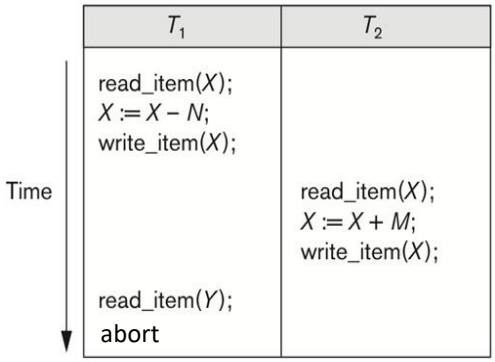
- operations ***read_lock(?)***, ***write_lock(?)***, and ***unlock(?)***, that acquires read lock, acquires write lock (or upgrades from read to write lock), and releases any lock, respectively (replace **?** with the proper data item).
- it is not allowed to issue ***write_lock*** in order to perform ***read_item*** (to facilitate more concurrency for read)
- For any data item, a transaction can own either read or write lock, but not both, at any time

Given two transactions in schedules S1 and S2 as illustrated in the following two figures.

Schedule S1: Lost update



Schedule S2: Dirty read



a. (5 pts) Schedule S1 (the lost update problem): apply the basic *two phase locking* (2PL) protocol to the entire T1 and entire T2 separately and they become T1' and T2'. Now interleave T1' and T2' according to S1 to justify why S1 is not allowed and therefore such problem does not happen. (Hint: examine each ***read_lock*** and ***write_lock*** operations in T1' and T2' to determine if each operation is allowed in S1.)

b. (5 pts) Schedule S2 (the dirty read problem): apply the strict *two phase locking* (2PL) protocol to the entire T1 and entire T2 separately and they become T1' and T2'. Now interleave T1' and T2' according to S2 to justify why S2 is not allowed and therefore such problem does not happen. (Hint: examine each ***read_lock*** and ***write_lock*** operations in T1' and T2' to determine if each operation is allowed in S2.)

c. (5 pts) Why is the basic 2PL not good enough for schedule S2?

5. (10 pts) Given two data items X and Y. Use the *basic timestamping* protocol for concurrency control to trace the following two transactions T1 and T2, and at each step indicate

- the time stamps of T1 and T2, i.e., TS(T1) = …, TS(T2) = …
- the read and write timestamps of data items X, Y, i.e., R_TS(X) = …, W_TS(X) = …, R_TS(Y) = …, W_TS(Y) = …
- the reason if the operation is not successfully.

Assumptions:

- Before T1 and T2 are executed, R_TS(X) = W_TS(X) = R_TS(Y) = W_TS(Y) = 0.
- The statements are executed in a strictly alternating way (first statement of T1 followed by the first statement of T2 followed by the second statement of T1, and so on) and there are no other transactions. Transaction timestamp assignment starts at 1, with increment 1, i.e., 1, 2, 3, 4, etc.

Transaction T1	Transaction T2
Write(X)	Read(Y)
Write(Y)	Write(X)

6. (10 pts, 5 pts per query) Use "standard" SQL (the ones we discussed in class); any DB engine specific functions (e.g., PARTITION, rowid, rownum, LIMIT, TOP, etc.) are not allowed.

Given the following table where primary key is underlined:

Products (ProductID, ProductName, Price)

Without using ***GROUP BY***, formulate **two** SQL queries, one is nested and the other non-nested, to list duplicated ProductName (where each such ProductName is listed only once). For example, if product names are

A
B
C
C
A
D
A

then the result set of the query should be

A
C

since A and C have duplicates while B and D do not (the result set order does not matter). Include (1) the SQL query (in text), (2) screenshot of table content, and (3) screenshot of both the SQL query and the result set (query output). No credit if no screenshot or screenshot is unreadable.

Submit the following *single* file

- `CMPE226_HW3_YourName` (.pdf, .doc, or .docx)

The ISA and/or instructor leave feedback to your homework as comments and/or annotated comment. To access annotated comment, click "view feedback" button. For details, see the following URL:

https://guides.instructure.com/m/4212/l/352349-how-do-i-view-annotation-feedback-comments-from-my-instructor-directly-in-my-assignment-submission