CS3223: Database Management Systems
Tutorial 10
(Week of 4th April 2022)

1. (This is the same question as Tutorial 9 Q3 which we did not manage to do in the last tutorial) Consider the following schedule:

$$X_1(B)\ X_4(A)\ S_3(C)\ S_1(A)\ X_2(D)\ X_2(C)\ X_3(B)\ S_4(D)$$

   (A) Give the wait-for-graph for the lock requests for the sequence of actions in the schedule under 2PL (with S/X locks). Is there a deadlock?

   (B) To prevent deadlock, we use a lock manager (LM) that adopts the Wait-Die policy. We assume the four transactions have priority: T1 < T2 < T3 < T4. Determine which lock request will be granted (`g'), blocked (`b') or aborted (`a'); for `abort', specify which transaction is aborted - e.g., `a' (T1 is aborted):
   - $X_1(B)$: granted/abort/blocked?
   - $X_4(A)$:
   - $S_3(C)$:
   - $S_1(A)$:
   - $X_2(D)$:
   - $X_2(C)$:
   - $X_3(B)$:
   - $S_4(D)$:

   (C) Now, we use a lock manager (LM) that adopts the Wound-Wait policy. We assume the four transactions have priority: T1 < T2 < T3 < T4. As in (B), determine which lock request will be granted (`g'), blocked (`b') or aborted (`a').

2. (Taken from main text) Consider a database organized in terms of the following hierarchy of objects: The database itself is an object (D), and it contains two files (F1 andF2), each of which contains 1000 pages (P1...P1000 and P1001...P2000, respectively). Each page contains 100 records, and records are identified as p : i, where p is the page identifier and i is the slot of the record on that page. Multiple-granularity locking is used, with S, X, IS, IX and SIX locks, and database level, file-level, page-level and record-level locking. For each of the following operations, indicate the sequence of lock requests that must be generated by a transaction that wants to carry out (just) these operations:
   1. Read record P1200 : 5.
   2. Read records P1200 : 98 through P1205 : 2.
   3. Read all (records on all) pages in file F1.
   4. Read pages P500 through P520.
   5. Read pages P10 through P980.
   6. Read all pages in F1 and (based on the values read) modify 10 pages.
   7. Delete record P1200 : 98. (This is a blind write.)
   8. Delete all records.

3. Consider a database with six elements A, B, C, D, E and F. There are five transactions T1 to T5 that read and write to these database elements. The read and write sets of transactions T, U, W and V are given in the table below.

Transaction	Read-Set	Write-Set
T1	{D}	{B, C}
T2	{A, C}	{D}
T3	{E}	{A, F}
T4	{A}	{A, D}
T5	{B, C}	{A, B, C}

The times at which the transactions start, try to validate and finish are as follows:

- T1, T2 starts
- T1 validates
- T3 starts
- T2 validates
- T2 finishes
- T4 starts
- T1 finishes
- T4 validates
- T5 starts
- T3 validates
- T4 finishes
- T5 validates
- T5 finishes
- T3 finishes

Which of these transactions validate successfully?

4. (Past-year question) Consider the concurrency control mechanism for B^+-tree (as taught in class). Suppose we have a 3 level tree – root, internal node, leaves. Which of the following sequence(s) is(are) possible sequences of actions on the tree to insert a new entry to a leaf page? Here, L denote Lock, U denote Unlock, M denote modification. L(X) denote locking a node at level X (X = R for root; I for internal node; and L for leaf node). U(X) is similarly defined. Explain your answer.

   A. L(R); L(I); L(L); M(L); U(R); U(I); U(L)
   B. L(R); L(I); U(R); L(L); U(I); M(L); U(L)
   C. L(R); L(I); L(L); U(R); U(I); M(L); U(L)
   D. L(R); L(I); U(R); L(L); M(L); U(I); U(L)