IMPORTANT

- Deposit hard copy of your answers in class at 1:30pm on Thu, 4/23/2015.
- Separate answers, as usually, i.e., please each question on a separate page, with the usual info (andrewID, etc)

Reminders

- Plagiarism: Homework may be discussed with other students, but all homework is to be completed individually.
- Typeset all of your answers whenever possible. Illegible handwriting may get no points, at the discretion of the graders.
- Late homeworks: please email late homeworks – to all TAs
  - with the subject line exactly 15-415 Homework Submission (HW 8)
  - and the count of slip-days you are using.

For your information:

- Graded out of 100 points; 4 questions total
- Rough time estimate: ≈4 hours (~1 hour for each question)

### Question Points Score

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serializability and 2PL</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Deadlock Detection and Prevention</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Hierarchical Locking - Oscar Nominations</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>B+ tree Locking</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Total:</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
Question 1: Serializability and 2PL ................. [20 points]
Submit on separate page
Course: 15-415/615; HW: ; Q: 
Name: ______________________; andrew-id: ____________________; late days: 

(a) Yes/No questions: 
   i. [2 points] Conflict Serializability always permits as many schedules as View Serializability does.  
      yes □ no □
   ii. [2 points] A schedule is conflict serializable if its dependency graph is acyclic. 
      yes □ no □
   iii. [2 points] In the Shrinking phase of 2PL, the transaction can be granted lock requests. 
      yes □ no □
   iv. [2 points] All serializable schedules are allowed by 2PL. 
      yes □ no □
   v. [2 points] Schedules under strict 2PL could have cascading aborts. 
      yes □ no □

(b) Serializability:  
Consider the schedule given below in Table 1. R(·) and W(·) stand for ‘Read’ and ‘Write’, respectively.

<table>
<thead>
<tr>
<th>time</th>
<th>t₁</th>
<th>t₂</th>
<th>t₃</th>
<th>t₄</th>
<th>t₅</th>
<th>t₆</th>
<th>t₇</th>
<th>t₈</th>
<th>t₉</th>
<th>t₁₀</th>
<th>t₁₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td></td>
<td>R(A)</td>
<td></td>
<td>W(A)</td>
<td></td>
<td>R(C)</td>
<td></td>
<td>W(C)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T₂</td>
<td></td>
<td>R(B)</td>
<td></td>
<td>W(B)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T₃</td>
<td>R(A)</td>
<td></td>
<td>W(A)</td>
<td></td>
<td></td>
<td></td>
<td>R(B)</td>
<td>W(B)</td>
<td>R(C)</td>
<td>W(C)</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: A schedule with 3 transactions

i. [1 point] Is this schedule serial? 
   □ Yes □ No □

ii. [3 points] Give the dependency graph of this schedule.

Solution: 
   • T₃ → T₁ because of A 
   • T₁ → T₃ because of C 
   • T₂ → T₃ because of B 

iii. [1 point] Is this schedule conflict serializable? 
    □ Yes □ No □

Question 1 continues...
iv. [3 points] If you answer “yes” to (iii), provide the equivalent serial schedule. If you answer “no”, briefly explain why.

**Solution:** This schedule is not conflict serializable because there exists a cycle \( (T_3 \rightarrow T_1 \rightarrow T_3) \) in the dependency graph.

v. [2 points] Could this schedule have been produced by 2PL?

☐ Yes  ■ No
Question 2: Deadlock Detection and Prevention . . . . [30 points]
Submit on separate page
Course: 15-415/615; HW: ; Q: 
Name: __________; andrew-id: _______________; late days: 

(a) Deadlock Detection:
Consider the following lock requests in Table 2. And note that
• S(·) and X(·) stand for ‘shared lock’ and ‘exclusive lock’, respectively.
• T_1, T_2, and T_3 represent three transactions.
• LM stands for ‘lock manager’.

<table>
<thead>
<tr>
<th>time</th>
<th>t_1</th>
<th>t_2</th>
<th>t_3</th>
<th>t_4</th>
<th>t_5</th>
<th>t_6</th>
<th>t_7</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_1</td>
<td>S(D)</td>
<td></td>
<td></td>
<td>X(C)</td>
<td></td>
<td>S(B)</td>
<td></td>
</tr>
<tr>
<td>T_2</td>
<td></td>
<td>S(A)</td>
<td></td>
<td></td>
<td>X(D)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T_3</td>
<td></td>
<td></td>
<td>S(A)</td>
<td></td>
<td></td>
<td></td>
<td>S(C)</td>
</tr>
<tr>
<td>LM</td>
<td>g</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Lock requests of 3 transactions

i. [6 points] For the lock requests in Table 2, determine which lock will be
granted or blocked by the lock manager. Please write ‘g’ in the LM row to
indicate the lock is granted and ‘b’ to indicate the lock is blocked. For example,
in the table, the first lock (S(D) at time t_1) is marked as granted.

Solution:
• S(A) at t_2: g
• S(A) at t_3: g
• X(D) at t_4: b
• X(C) at t_5: g
• S(C) at t_6: b
• S(B) at t_7: g

ii. [4 points] Give the wait-for graph for the lock requests in Table 2.

Solution: T_3 → T_1 ← T_2

iii. [3 points] Determine whether there exists a deadlock in the lock requests in
Table 2, and briefly explain why.

Question 2 continues...
Solution: There will not be a deadlock because the dependency graph is acyclic.

(b) Deadlock Prevention:
Consider the following lock requests in Table 3. Again,

• S(·) and X(·) stand for ‘shared lock’ and ‘exclusive lock’, respectively.
• $T_1$, $T_2$, $T_3$, and $T_4$ represent four transactions.
• $LM$ represents a ‘lock manager’.

<table>
<thead>
<tr>
<th>Time</th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
<th>$t_5$</th>
<th>$t_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>S(D)</td>
<td></td>
<td>S(B)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$T_2$</td>
<td></td>
<td></td>
<td>S(C)</td>
<td>X(D)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$T_3$</td>
<td></td>
<td>X(B)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$T_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X(C)</td>
<td></td>
</tr>
<tr>
<td>$LM$</td>
<td>g</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Lock requests of 4 transactions

i. [6 points] For the lock requests in Table 3, determine which lock request will be granted, blocked or aborted by the lock manager ($LM$), if it has no deadlock prevention policy. Please write ‘g’ for grant, ‘b’ for block and ‘a’ for abort. Again, example is given in the first column.

Solution:

• X(B) at $t_2$: g
• S(B) at $t_3$: b
• S(C) at $t_4$: g
• X(C) at $t_5$: b
• X(D) at $t_6$: b

ii. [5 points] Give the wait-for graph for the lock requests in Table 3. Determine whether there exists a deadlock in the lock requests in Table 3 under $LM$, and briefly explain why.

Solution:

• $T_1 \rightarrow T_3$
• $T_2 \rightarrow T_1$
• $T_4 \rightarrow T_2$

Question 2 continues...
The lock requests don’t have a deadlock because there is no cycle in the dependency graph.

iii. [3 points] To prevent deadlock, we use the lock manager \((LM)\) that adopts the Wait-Die policy. We assume that in terms of priority: \(T_1 > T_2 > T_3 > T_4\). Determine which lock request will be granted (‘g’), blocked (‘b’) or aborted (‘a’). Follow the same format as the previous question.

**Solution:**

- X(B) at \(t_2\): g
- S(B) at \(t_3\): b
- S(C) at \(t_4\): g
- X(C) at \(t_5\): a
- X(D) at \(t_6\): a

iv. [3 points] Now we use the lock manager \((LM)\) that adopts the Wound-Wait policy. We assume that in terms of priority: \(T_1 > T_2 > T_3 > T_4\). Determine which lock request will be granted (‘g’), blocked (‘b’) or aborted (‘a’). Follow the same format as the previous question.

**Solution:**

- X(B) at \(t_2\): g
- S(B) at \(t_3\): g (\(T_3\) aborts)
- S(C) at \(t_4\): g
- X(C) at \(t_5\): b
- X(D) at \(t_6\): b

Homework 8 continues...
Question 3: Hierarchical Locking - Oscar Nominations[30 points]

Submit on separate page
Course: 15-415/615; HW: ; Q:
Name: __________________; andrew-id: __________________; late days:
Graded by: Hong Bin Shim

Consider a Database (D) consisting of two tables, Actors (A) and Nominations (N). Specifically,

- Actors(aid, first_name, last_name), spans 300 pages, namely \( A_1 \) to \( A_{300} \)
- Nominations(nid, aid, year, movie, character, won), spans 600 pages, namely \( N_1 \) to \( N_{600} \)

Further, each page contains 100 records, and we use the notation \( A_3:20 \) to represent the 20th record on the third page of the Actors table. Similarly, \( N_5:10 \) represents the 10th record on the fifth page of the Nominations table.

We use Multiple-granularity locking, with S, X, IS, IX and SIX locks, and four levels of granularity: (1) database-level (D), (2) table-level (A, N), (3) page-level (\( A_1 - A_{300}, N_1 - N_{600} \)), (4) record-level (\( A_1:1 - A_{300}:100, N_1:1 - N_{600}:100 \)).

For each of the following operations on the database, please determine the sequence of lock requests that should be generated by a transaction that want to carry out these operations efficiently.

Please follow the format of the examples listed below:

- write “IS(D)” for a request of database-level IS lock
- write “X(\( N_2:30 \))” for a request of record-level X lock for the 30th record on the second page of the Nominations table
- write “S(\( N_2:30 - N_3:100 \))” for a request of record-level S lock from the 30th record on the second page of the Nominations table to the 100th record on the third page of the Nominations table.

(a) [6 points] Read 100th record on page \( A_1 \).

Solution: IS(D), IS(A), IS(\( A_1 \)), S(\( A_1:100 \))

(b) [6 points] Read ALL records on page \( A_1 \) through \( A_{15} \), and modify the records \( A_{10}:10 \) through \( A_{10}:100 \).

Solution: IX(D), SIX(A), IX(\( A_{10} \)), X(\( A_{10}:10 - A_{10}:100 \)); also acceptable: IX(D), IX(A), S(\( A_{11} - A_9 \)), S(\( A_{11} - A_{15} \)), SIX(\( A_{10} \)), X(\( A_{10}:10 - A_{10}:100 \))

(c) [6 points] Modify the first record on EACH and EVERY page of the Nominations table (these are blind writes that do not depend on the original contents in the pages).

Question 3 continues...
Solution: IX(D), X(N)

(d) [6 points] For EACH record in the Actors table, capitalize the English letters in the last_name if it is not capitalized. That is, “Redmayne” will be modified as “REDMAYNE” but “MOORE” will be left unchanged.

Solution: IX(D), X(A)

(e) [6 points] Delete ALL the records from ALL tables.

Solution: X(D)
Question 4: B+ tree Locking .......................... [20 points]
Submit on separate page
Course: 15-415/615; HW: ; Q: 
Name: __________________; andrew-id: ______________; late days: 
Consider the following B+ tree:

![B+ tree diagram]

Figure 1: B+ tree locking

To lock this B+ tree, we would like to use the Bayer-Schkolnick algorithm (described in lecture notes #22[1] slide 32 - 35). Important: we use the version as presented in the lecture, which does not use lock upgrade.

For each of the following transactions, give the sequence of lock/unlock requests. For example, please write S(A) for a request of shared lock on node A, X(B) for a request of exclusive lock on node B and U(C) for a request of unlock node C.

Important notes:

- Each of the following transactions is applied on the original tree, i.e., please ignore any change to the tree from earlier problems.
- For simplicity, ignore the changes on the pointers between leaves.

(a) [5 points] Search for data entry “37*”

Solution: S(A), S(C), U(A), S(G), U(C), S(N), U(G), U(N)

(b) [5 points] Delete data entry “7*”

Solution: S(A), S(B), U(A), S(D), U(B), X(I), note that the basic algorithm wins because we don’t need to merge on deletion.


Question 4 continues...
U(D), U(I)

Final answer: S(A), S(B), U(A), S(D), U(B), X(I), U(D), U(I)

(c) [5 points] Insert data entry “30*”

Solution: S(A), S(C), U(A), S(F), U(C), X(M), note that leaf is not safe because we need to split it, U(M), U(F), we need to restart X(A), X(C), U(A), X(F), U(C), X(M), U(F), U(M)

Final answer: S(A), S(C), U(A), S(F), U(C), X(M), U(M), U(F), X(A), X(C), U(A), X(F), U(C), X(M), U(F), U(M)

(d) [5 points] Insert data entry “47*”

Solution: S(A), S(C), U(A), S(G), note that we cannot unlock C here because G is full, meaning that it is not safe, X(P), U(C), U(G), U(P), we can unlock G and C after we lock P because we know G is safe at this point

Final answer: S(A), S(C), U(A), S(G), X(P), U(C), U(G), U(P)