Carnegie Mellon University
15-415A Database Applications
C. Faloutsos - Fall 2001

Homework 6 - Due: Nov. 20, 2001

Reminders
Due: Nov. 20, 3pm (in class)
Please, TYPE your answers
Total weight: 3% of the course grade.
Estimated time: 40-60 minutes

1 Schedules [29 pts]
Consider the interleaved execution (‘schedule’) of Figure 1
1. Is it serial? (Yes/No) [3 pts]
2. Is it serializable? (Yes/No) [3 pts]
3. Give the precedence graph. [4 pts]
4. Very tricky: Could it occur under the 2PL protocol? (Yes/No) [7 pts]
5. Justify your answer above: if ’Yes’, give the timestamps of the lock requests; if ’No’,
   prove it rigorously. [12 pts]

2 Deadlocks [15 pts]
Consider the situation of Figure 2.
1. Show that we don’t have a deadlock at time ’t4’. [5 pts]
2. Give a lock-request that would cause a deadlock at ’t5’. [5 pts]
3. Give the ’wait-for’ graph, after your suggested lock-request [5 pts]
<table>
<thead>
<tr>
<th>time</th>
<th>T1</th>
<th>T2</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>read(A)</td>
<td></td>
</tr>
<tr>
<td>t2</td>
<td>write(A)</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t10</td>
<td></td>
<td>read(A)</td>
</tr>
<tr>
<td>t11</td>
<td></td>
<td>write(A)</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t20</td>
<td></td>
<td>read(B)</td>
</tr>
<tr>
<td>t21</td>
<td></td>
<td>write(B)</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t31</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: An interleaved execution

<table>
<thead>
<tr>
<th>time</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>lock-mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>lock(A)</td>
<td>lock(B)</td>
<td></td>
<td>granted</td>
</tr>
<tr>
<td>t2</td>
<td></td>
<td>lock(B)</td>
<td>lock(C)</td>
<td>granted</td>
</tr>
<tr>
<td>t3</td>
<td>lock(B)</td>
<td></td>
<td>lock(C)</td>
<td>granted</td>
</tr>
<tr>
<td>t4</td>
<td>lock(B)</td>
<td></td>
<td></td>
<td>wait</td>
</tr>
<tr>
<td>...</td>
<td>....</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: Maybe we have a deadlock

3 Multiple granularity locks [16 pts]

Consider the following hierarchy of lock-able objects:

- Database
  - Employee-table(ssn, name, salary)
    - E1(123, Smith, 50K)
    - E2(234, Johnson, 30K)
  - Department-table(dnumber, dname, budget)
    - D1(10, Marketing, 1M)
    - D2(20, Engineering, 1.5M)

For each of the following transactions, list the objects and the locks they should request, in the proper sequence:

1. T1: wants to read the salary of 'Smith' [4 pts]
2. T2: wants to change the name of 'Johnson' to 'Jonson' [4 pts]
3. T3: wants to give a 5% raise to every employee [4 pts]
4. T4: wants a list of all the department-names [4 pts]
4 Recovery [32 pts]

Consider the log of Figure 3. Assume it operates with 'incremental updates' and checkpoints. The records are of the form: (t-id, object-id, old-value, new-value)

(T1 starts )
(T2 starts )
(T1, A, 10, 20)
(T1, B, 15, 25)
(T2, C, 100, 150)
(checkpoint { ????? } )
(T1 commits)
(T2, D, 200, 250)
(T3 starts)
(T3, E, 1000, 1500)
(checkpoint { ????? } )
(T2 commits)
- - - crash - - -

Figure 3: Write-ahead log.

1. Which transaction-ids should we have instead of the 'question marks' at the first checkpoint? [4 pts]
2. Repeat, for the second checkpoint [4 pts]
3. For each data item (A-E), determine its value on the disk, after the crash and before the recovery. Possible answers: 'old-value', 'new-value', 'can-not.tell' [12 pts]
4. Repeat, for after the recovery: For each data item A-E, determine its value on the disk ('old', 'new', 'can-not.tell') [12 pts]

5 Semijoins [8 pts]

Consider the relations $R(A, B)$ and $S(A, C, D)$, with the following extensions:

\[
R(A, B) = \{(a1, b1), (a2, b1), (a2, b3)\}
\]
\[
S(A, C, D) = \{(a1, c1, d1), (a3, c1, d5)\}
\]

1. Give the result of the semijoin: $R$ semijoin $S$ [4 pts]
2. Give the symmetric result: $S$ semijoin $R$ [4 pts]

End of questions