Chapter 13-15 Problems

(10) 1. Given relations \( t_r(V, W, X) \) and \( t_s(X, Y, Z) \), which have the following properties: \( t_r \) has 10,000 tuples, \( t_s \) has 90,000 tuples, 50 tuples of \( t_r \) fit on one block, and 15 tuples of \( t_s \) fit on one block. Estimate the number of block transfers and seeks required, using each of the following join strategies for \( t_r \bowtie t_s \):

   a. Nested-loop join
   b. Block nested-loop join
   c. Indexed nested-loop join (suppose there is a primary B+ tree index with height 5 on the join attribute (X).)
   d. Merge join
   e. Hash join

(30) 2. Using a sequence of transformations from the 12 equivalence rules for relational algebra expressions listed in Section 14.2.1 (5th edition), show how to derive the following equivalences:

   (A) \( \sigma_{\theta_1 \land \theta_2 \land \theta_3}(E) = \sigma_{\theta_1}(\sigma_{\theta_2}(\sigma_{\theta_3}(E))) \)
   (B) \( \sigma_{\theta_1 \land \theta_2}(E_1 \bowtie E_2) = \sigma_{\theta_1}(E_1 \bowtie \sigma_{\theta_2}(E_2)) \), where \( \theta_2 \) involves only attributes from \( E_2 \).

(20) 3. Why is conflict serializability emphasized rather than view serializability, even though every conflict-serializable schedule is view serializable?

(20) 4. Consider the two transactions \( T_1 \) and \( T_2 \):

   \( T_1 \): read(\( P \));
   read(\( Q \));
   if \( P = 0 \) then \( Q := Q + 1 \);
   write(\( Q \));

   \( T_2 \): read(\( Q \));
   read(\( P \));
   if \( Q = 0 \) then \( P := P + 1 \);
   write(\( P \));

   Let the consistency requirement be \( P = 0 \lor Q = 0 \), with \( P = Q = 0 \) the initial values.

   a. Show that every serial execution involving these two transactions preserves the consistency of the database.
   b. Show a concurrent execution of \( T_1 \) and \( T_2 \) that produces a nonserializable schedule.
   c. Is there a concurrent execution of \( T_1 \) and \( T_2 \) that produces a serializable schedule?
5. Prove that the following schedule is not under the timestamp protocol, but instead, is under the two-phase locking protocol.

<table>
<thead>
<tr>
<th>step</th>
<th>$T_0$</th>
<th>$T_1$</th>
<th>Precedence marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lock-S(A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>read(A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>lock-X(B)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>write(B)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>unlock(B)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>lock-S(B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>read(B)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>unlock(A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>unlock(B)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$T_1 \rightarrow T_0$