Exam 2 Part 2 Solutions

Name: 
UFID: 
Email Address: 

Pledge (Must be signed according to UF Honor Code)

On my honor, I have neither given nor received unauthorized aid in doing this assignment.

Signature

For scoring use only:

<table>
<thead>
<tr>
<th></th>
<th>Maximum</th>
<th>Received</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 4</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Question 5</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>
Question 3 (Left Outer Join)  [20 points]

1. Let \( \mathcal{R} \) be a relation schema and \( R \) be a relation with respect to \( \mathcal{R} \), Let \( \mathcal{S} \) be a relation schema and \( S \) be a relation with respect to \( \mathcal{S} \). We assume that \( \mathcal{R} \cap \mathcal{S} \neq \emptyset \). Show formally (without examples) that the outer-join operator \( R \bowtie S \) is a derived Relational Algebra operator. That is, it can be expressed by the 5 \((+1)\) basic Relational Algebra operators. You are allowed to make use of derived Relational Algebra operators that have already been shown to be derived in class. [10 points]

Let the relation \( T \) have the schema \( S - \mathcal{R} \). Let \( T = \{ (null, null, \ldots, null) \} \), that is, \( T \) is a relation with a single tuple and \( |S - \mathcal{R}| \) attributes, and all attribute values of this single tuple are \( null \). Then we can define:

\[
R \bowtie S = R \bowtie S \cup (R - \pi_{\mathcal{R}}(R \bowtie S)) \times T
\]

2. Let now \( \mathcal{R} = \{ A_1, A_2, \ldots, A_n, C_1, \ldots, C_k \} \) and \( \mathcal{S} = \{ B_1, B_2, \ldots, B_m, C_1, \ldots, C_k \} \) be table schemas that correspond to the relation schemas in part 1. The attributes \( C_1, \ldots, C_k \) are shared by both schemas. The \( A_i \)'s and \( B_j \)'s are supposed to be distinct from each other. Let \( R \) and \( S \) be tables with respect to the schemas \( \mathcal{R} \) and \( \mathcal{S} \) respectively. Provide a translation of your definition in part 1 into a corresponding SQL expression. The translation should be as near as possible to the Relational Algebra expression in part 1. [10 points]

```sql
CREATE TABLE T (B1 DT1, B2 DT2, ..., Bm DTm);
   -- “DTj” means “data type j”

INSERT INTO T (B1, B2, ..., Bm) VALUES (null, null, ..., null);

(SELECT * FROM R NATURAL JOIN S)
UNION
( SELECT * FROM
   ( SELECT * FROM R
   MINUS
   (SELECT A1, ..., An FROM R NATURAL JOIN S)
  )
)
CROSS JOIN T
);
```
Question 4 (SQL)  [30 points]

Consider the following database schema:

Product(maker, model, type)
PC(model1, speed, ram, hd, rd, price)
Laptop(model1, speed, ram, hd, screen, price)
Printer(model1, color, type, price)

The Product relation gives the manufacturer, model number and type (PC, laptop, or printer) of various products. We assume for convenience that model numbers are unique over all manufacturers and product types. The PC relation gives for each model number that is a PC the speed (of the processor, in gigahertz), the amount of RAM (in megabytes), the size of the hard disk (in gigabytes), the speed and type of the removable disk (CD or DVD), and the price. The Laptop relation is similar, except that the screen size (in inches) is recorded in place of information about the removable disk. The Printer relation records for each printer model whether the printer produces color output (true if so), the process type (laser, ink-jet, or bubble), and the price.

Write a SQL statement for the following questions:

1. Find the manufacturers that make at least five different models of laptops. [6 points]

   SELECT maker
   FROM Product
   WHERE type='laptop'
   GROUP BY maker
   HAVING COUNT(model) >= 5;

2. Find the average ram amount of a laptop for all those manufacturers that make printers. [6 points]

   SELECT AVG(L.ram) AS Avg_Ram_Amount
   FROM Product R, Laptop L
   WHERE R.model = L.model AND R.maker IN ( SELECT maker
                                                   FROM Product
                                                   WHERE type = 'printer'
                                                   );

3. Find the model number of the item (PC, laptop, or printer) with the highest price. [6 points]

   SELECT model, price FROM ( SELECT model, price FROM PC UNION
                              SELECT model, price FROM Laptop UNION
                              SELECT model, price FROM Printer ) M1
WHERE M1.price >= ALL ( SELECT price FROM PC UNION SELECT price FROM Laptop UNION SELECT price FROM Printer );

4. Delete all PCs from PC table which are made by any manufacturer that doesn't make laptops. [6 points]

DELETE FROM PC P WHERE NOT EXISTS ( SELECT * FROM Product P1, Product P2 WHERE P1.model = P.model AND P1.maker = P2.maker AND P2.type = 'laptop' );

5. For each maker, find the average prices of PCs sold by it. [6 points]

SELECT AVG(P.price) FROM PC P, Product P1 WHERE P.model = P1.model GROUP BY P1.maker;