Problem 1 (10 pts)
How many different ways (considering orders of tuples and attributes) are there to represent a relation instance if that instance has \( n \) attributes and \( m \) tuples?

Problem 2 (30 pts)
Do Exercise 4.2 on page 127 of the text, skipping part (7).

Problem 3 (15 pts)
(a) Suppose relations \( R(A, B, C) \) and \( S(A, B, C) \) have the tuples as shown:

\[
\begin{array}{ccc}
R: & & \\
A & B & C \\
--- & --- & --- \\
3 & 2 & 1 \\
4 & 2 & 3 \\
4 & 5 & 6 \\
2 & 5 & 3 \\
1 & 2 & 6
\end{array}
\quad
\begin{array}{ccc}
S: & & \\
A & B & C \\
--- & --- & --- \\
2 & 5 & 3 \\
2 & 5 & 4 \\
4 & 2 & 3 \\
3 & 2 & 1
\end{array}
\]

Compute \((R - S) \cup (S - R)\), known as the “symmetric difference” of \( R \) and \( S \). List all the tuples in the result relation.
(b) Suppose relations $R(A, B)$ and $S(B, C, D)$ have the tuples as shown:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>9</td>
</tr>
</tbody>
</table>

Compute $R \bowtie_{R.A < S.C \land R.B < S.D} S$ and list all the result tuples.

**Problem 4 (30 pts)**

Use the following database schema for this problem. The relations represent information on bank branches and their customers:

- **Customers** $(\text{cname}, \text{street}, \text{city})$
- **Branches** $(\text{bname}, \text{city})$
- **Accounts** $(\text{cname}, \text{bname}, \text{anumber})$

Here $\text{cname}$ means customer-name, $\text{bname}$ means branch-name, and $\text{anumber}$ means account-number.

The **Customers** relation has customer names and their addresses. The **Branches** relation has branch names and the city that the branch is located in. The **Accounts** relation represents at which branch a customer has his/her accounts. We assume that customer names and branch names are unique. We also assume that a customer may have multiple accounts in one branch and the customer may have accounts in multiple branches.

Complete a relational algebra expression for each of the following queries by filling the blanks in the partial answer provided. I am providing partial solutions to make grading a little easier. *(Hint: When a query is difficult to write, think of its complement.)*

(a) Find the names of all customers who have an account in the 'Claremont' branch.

$$\pi_{\ldots} (\sigma_{\ldots} (\ldots))$$

(b) Find the names of all customers who have an account in a branch not located in the same city that they live in.

$$\pi_{\ldots} (\sigma_{\ldots} (\rho_{\ldots} (\text{Branches}) \ldots \rho_{\ldots} (\text{Customers} \ldots \text{Accounts})))$$

(c) Find the customer names who do not have any account in the 'Claremont' branch.

$$\pi_{\ldots} (\text{Customers}) \ldots \pi_{\ldots} (\sigma_{\ldots} (\text{Accounts}))$$

**Problem 5 (15 pts)**

The relation **Students** $(\text{sid}, \text{gpa})$ captures the student-gpa information, where $\text{sid}$ is the id and $\text{gpa}$ is the GPA of a student. Complete a relational algebra expression I wrote to “find the ids of the students with the lowest GPA.” by filling in the blanks. We will not use any aggregate operator in answering this question.

$$\pi_{\ldots} (\text{Student}) \ldots \pi_{\ldots} \sigma_{\ldots} (\rho_{\ldots} (\text{Student}) \ldots \rho_{\ldots} (\text{Student})))$$
Problem 6 (15 pts)
Assume the following database schema for this problem:

- Employees\texttt{(pname, age, street, city)}
- Works\texttt{(pname, \textit{cname}, salary)}
- Companies\texttt{(\textit{cname}, city)}
- Manages\texttt{(pname, mname)}

Here \texttt{pname} means person-name, \texttt{cname} means company-name, and \texttt{mname} means manager-name. A person may work for more than one company. A company may be located in more than one city. Each person’s name is unique.

(a) Complete the query that I started in SQL to find the names of the companies that all of their employees have salaries higher than $100,000.

```
SELECT \textit{cname}
FROM Works W1
WHERE 100000 \textless{} (SELECT salary
FROM Works W2
WHERE W1.cname=W2.cname)
```

(b) Complete the same query that I started in Relational Algebra.

```
\pi_{\textit{cname}}(\textit{Works}) \cap \pi_{\textit{cname}}(\sigma_{\textless{}(\textit{Works})})
```

(c) Compare the results of (a) and (b). Are they the same? Why or why not?

Problem 7 (20 pts)
Use the following database schema for this problem.

- MovieStars\texttt{(name, address, gender)}
- MovieExecs\texttt{(name, address, company, netWorth)}

(a) We want to find the names and addresses of all female movie stars (\texttt{gender = ‘F’} in the \textit{MovieStars} relation) who are also movie executives with a net worth of over $1,000,000 (\texttt{netWorth > 1000000} in the \textit{MovieExecs} relation).

(1) Complete the query using \texttt{INTERSECT} operator.

```
SELECT name, address
FROM MovieStars
WHERE __________________
INTERSECT
SELECT name, address
FROM MovieExecs
WHERE __________________
```
(2) Complete the query without using \texttt{INTERSECT} operator, but using \texttt{IN} operator.

\begin{verbatim}
SELECT name, address 
FROM MovieStars 
WHERE _________________ IN (SELECT name, address 
                        FROM MovieExecs 
                        WHERE _________________)
\end{verbatim}

(b) We want to find the movie stars who are not movie executives.

(1) Write the query using \texttt{EXCEPT} operator.

\begin{verbatim}
SELECT name 
FROM ____________ 
EXCEPT 
SELECT name 
FROM ____________
\end{verbatim}

(2) Write the query without using \texttt{EXCEPT} operator, but using \texttt{IN} operator.

\begin{verbatim}
SELECT name 
FROM ____________ 
WHERE name ________ IN (SELECT name 
                        FROM ____________) 
\end{verbatim}

\textbf{Problem 8 (50 pts)}

Assume the following database schema for this problem.

\begin{verbatim}
Computers(maker, model, price) 
Desktops(model, speed, ram, hdd) 
Laptops(model, speed, ram, hdd, weight) 
\end{verbatim}

Assume that a computer is either a desktop or a laptop.

(a) Using a \texttt{CHECK} constraint on the \texttt{Laptops} table, express the constraint that a laptop cannot have weight greater than 5kg. You do not need to show the entire \texttt{CREATE TABLE} statement. Show only the \texttt{CHECK} constraint part in the \texttt{CREATE TABLE} statement.

(b) Complete the trigger that I started to replace the \texttt{CHECK} constraint in part (a) above, so that when we try to add a laptop with weight greater than 5kg, the tuple is still inserted, but the value of the "weight" attribute is set to NULL.

\begin{verbatim}
CREATE TRIGGER T 
AFTER INSERT ____ Laptops 
REFERENCING __________ nrow 
FOR EACH __________ 
WHEN (nrow.__________) 
BEGIN ____ Laptops 
    SET weight = NULL 
    WHERE __________ 
END
\end{verbatim}
Using the database given above, answer the following questions. Complete the SQL statements that I started.

(c) Find the average speed of all desktop computers.

```
SELECT ____________ FROM Desktops
```

(d) Find the average price of all laptops with weight below 2kg.

```
SELECT ________________
FROM Computers C, Laptops L
WHERE _________________________
```

(e) Find the average price of desktops and laptops made by “Apple.”

```
SELECT _______________
FROM Computers
WHERE ________________
```

(f) For each different CPU speed, find the average price of a laptop.

```
SELECT ________________
FROM Laptops L, __________________
_______________________
____________ speed
```

(g) Find the manufacturers that make at least three different computer models.

```
SELECT maker
FROM Computers
__________ maker
__________ COUNT(model)_____________
```

Also write the following database modifications by completing what I started.

(h) Using two INSERT statements, insert a desktop computer manufactured by HP, with model number 1200, price $1000, speed 2.5GHz, 2GB RAM, and a 120GB hard drive.

```
INSERT INTO __________ VALUES ('HP', 1200, 1000)
INSERT INTO __________ VALUES _______________________
```

(i) Using two DELETE statements, delete all desktops manufactured by IBM with price below $1000. (Note: Be careful with the order of your two DELETE statements.)

```
DELETE FROM Desktops
WHERE model _____ (SELECT model
FROM Computers
WHERE maker='IBM' AND price<1000)
DELETE FROM Computers
WHERE _____________________________
```
(j) For each laptop made by Sony, add 2kg to the weight. *(Hint: The *WHERE* clause in an *UPDATE* statement may contain complex conditions, including subqueries.)*

```
UPDATE Laptops
SET ______________
WHERE model _____ (SELECT model
             FROM Computers
             WHERE ________________)
```

**Problem 9 (10 pts)**

In this problem you will express a referential integrity constraint using a general SQL assertion. Assume that there are two tables \( R(A) \) and \( S(A) \). Using a general SQL assertion, express that \( S.A \) is a foreign key referencing \( R.A \). That is, there should not be any \( S.A \) value that does not appear in \( R.A \).

Note that a general SQL assertion is not attached to a particular table. Therefore, whenever a SQL modification statement is executed that may potentially violate the assertion, the DBMS checks the result of the statement and rejects the statement if it causes violation. This behavior is the same as the default semantics of a foreign-key constraint.

Complete what I started here:

```
CREATE ASSERTION FKey ______ (NOT _______ (SELECT *
             FROM S
             WHERE A _____ (SELECT A
                     FROM R)))
```

**Problem 10 (15 pts)**

Consider the table \( R(A,B) \), which currently has only one tuple \((1,0)\). Assume that the following trigger has already been created for the database.

```
CREATE TRIGGER Times2
AFTER UPDATE ON R
REFERENCING
    NEW ROW AS n
FOR EACH ROW
WHEN (n.B < 5)
BEGIN
    UPDATE R SET B=B*2 WHERE A=n.A;
    INSERT INTO R VALUES(100, 0);
END
```

List all tuples in the table \( R \) after the following update statement is executed:

```
UPDATE R SET B=2 WHERE A=1
```