This problem set is due at **11:55pm on Wednesday, April 22, 2020**. Don’t go by the due date that you see on Blackboard because it is in EST. Go by the one given in this handout.

**Be sure to include** a comment at the top of each file submitted that gives **your name** and **email address**.

Submit all the files that make up your solutions on **Blackboard**. Multiple submissions are allowed before the due date.

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

**Problem 2 (30 pots)**
Given two relations \( R_1 \) and \( R_2 \), where \( R_1 \) contains \( n_1 \) tuples, \( R_2 \) contains \( n_2 \) tuples, and \( n_2 > n_1 > 0 \), give the minimum and maximum possible sizes (in number of tuples) for the resulting relation produced by each of the following relational algebra expressions. In each case, state any assumptions about the schemas for \( R_1 \) and \( R_2 \) needed to make the expression meaningful:

1. \( R_1 \cup R_2 \)
2. \( R_1 \cap R_2 \)
3. \( R_1 - R_2 \)
4. \( R_1 \times R_2 \)
5. \( \sigma_{a=5}(R_1) \)
6. \( \pi_a(R_1) \)

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

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>
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:

\[
\begin{array}{ccc}
R: & S: \\
A & B & B & C & D \\
----- & ----- & --- & --- & --- \\
1 & 2 & 2 & 4 & 6 \\
3 & 4 & 8 & 6 & 8 \\
5 & 6 & 7 & 5 & 9 \\
\end{array}
\]

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(cname, street, city)`
- `Branches(bname, city)`
- `Accounts(cname, bname, aid)`

Here `cname` means customer-name, `bname` means branch-name, and `aid` means account-id.

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 in 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 ‘Songdo’ 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.

\[\pi_{\ldots}(\sigma_{\ldots}(\rho_{\ldots}(\ldots)))\]

(c) Find the customer names who do not have any account in the ‘Songdo’ branch.

\[\pi_{\ldots}(\ldots)\]

Problem 5 (15 pts)

The relation `Students(sid, gpa)` captures the student-gpa information, where `sid` is the student id and `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.

\[\pi_{\ldots}(\ldots)\]
Problem 6 (15 pts)

Assume the following database schema for this problem:

- Employees(ename, age, street, city)
- Works(ename, cname, salary)
- Companies(cname, city)
- Manages(ename, mname)

Here ename means employee-name, cname means company-name, and mname means manager-name.

A person may work for more than one company as an employee. 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 $110,000.

```
SELECT cname
FROM Works W1
WHERE _______ ______ (SELECT salary
FROM Works W2
WHERE W1.cname=W2.cname)
```

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

```
πcname(Works)____πcname(σ____(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(name, address, gender)
- MovieExecs(name, address, company, netWorth)

(a) We want to find the names and addresses of all female movie stars (gender = ‘F’ in the MovieStars relation) who are also movie executives with a net worth of over $1,500,000 (netWorth > 1500000 in the MovieExecs relation).

(1) Complete the query using INTERSECT operator.

```
SELECT name, address
FROM MovieStars
WHERE _____________
INTERSECT
SELECT name, address
FROM MovieExecs
WHERE _____________
```
(2) Complete the query without using INTERSECT operator, but using IN operator.

```sql
SELECT name, address
FROM MovieStars
WHERE ___________________ IN (SELECT name, address
FROM MovieExecs
WHERE ___________________)
```

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

(1) Write the query using EXCEPT operator.

```sql
SELECT name
FROM ___________________
EXCEPT
SELECT name
FROM ___________________
```

(2) Write the query without using EXCEPT operator, but using IN operator.

```sql
SELECT name
FROM ___________________
WHERE name __________ IN (SELECT name
FROM ___________________
```

Problem 8 (50 pts)

Assume the following database schema for this problem.

- Computers(maker, model, price)
- Desktops(model, speed, ram, hdd)
- Laptops(model, speed, ram, hdd, weight)

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

(a) Using a CHECK constraint on the Laptops table, express the constraint that a laptop cannot have weight greater than 3kg. You do not need to show the entire CREATE TABLE statement. Show only the CHECK constraint part in the CREATE TABLE statement.

```sql
CREATE TRIGGER T
AFTER INSERT ____ Laptops
REFERENCING __________ nrow
FOR EACH __________
WHEN (nrow._____________)
BEGIN _______ Laptops
SET weight = NULL
WHERE _______________
END
```

(b) Complete the trigger that I started to replace the CHECK constraint in part (a) above, so that when we try to add a laptop with weight greater than 3kg, the tuple is still inserted, but the value of the “weight” attribute is set to NULL.
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.

\[
\text{SELECT } \text{__________ FROM Desktops}
\]

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

\[
\text{SELECT } \text{__________ FROM Computers C, Laptops L WHERE } \text{__________}
\]

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

\[
\text{SELECT } \text{__________ FROM Computers WHERE } \text{__________}
\]

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

\[
\text{SELECT } \text{__________ FROM Laptops L, } \text{__________ speed}
\]

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

\[
\text{SELECT maker FROM Computers WHERE } \text{__________ COUNT(model) } \text{__________}
\]

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, 8GB RAM, and a 512GB hard drive.

\[
\text{INSERT INTO } \text{__________ VALUES ('HP', 1200, 1000)}
\]
\[
\text{INSERT INTO } \text{__________ VALUES ________}
\]

(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.*)

\[
\text{DELETE FROM Desktops WHERE model } \text{__________ (SELECT model FROM Computers WHERE maker='IBM' AND price<1000)}
\]
\[
\text{DELETE FROM Computers WHERE } \text{__________}
\]
(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 an 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 nr
FOR EACH ROW
WHEN (nr.B < 5)
BEGIN
    UPDATE R SET B=B*2 WHERE A=nr.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
```